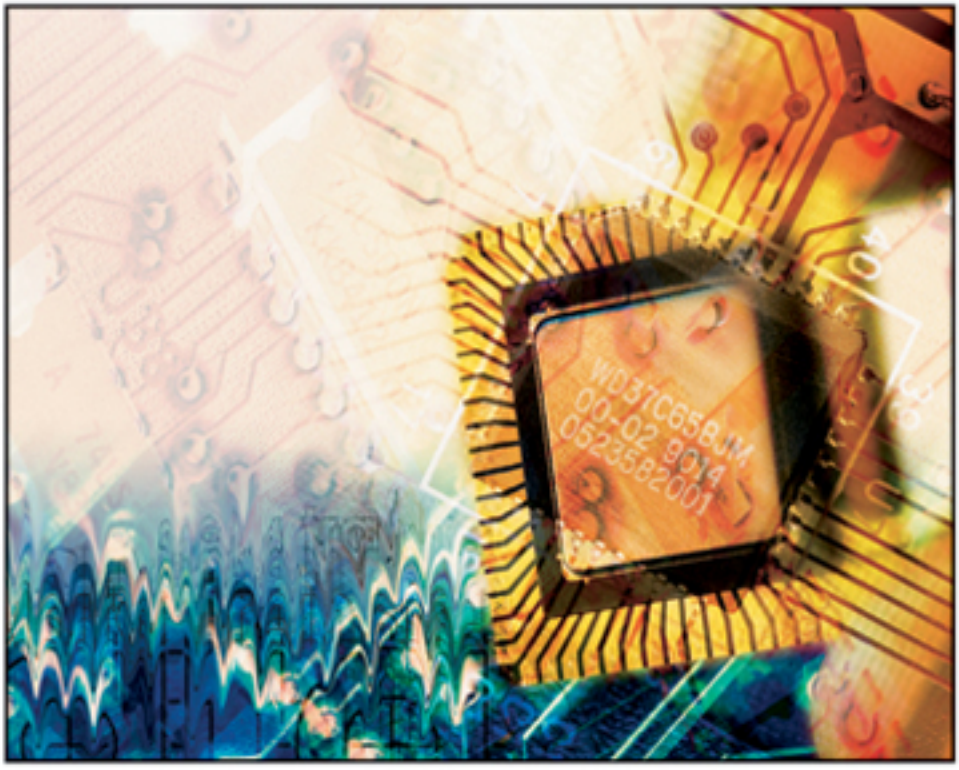
HANDBOOK OF RESEARCH ON

Innovations in Database Technologies and Applications

Current and Future Trends



Viviana E. Ferraggine, Jorge H. Doorn, & Laura C. Rivero

Handbook of Research on Innovations in Database Technologies and Applications:
Current and Future Trends

Viviana E. Ferraggine
Universidad Nacional del Centro de la Provincia de Buenos Aires, Argentina

Jorge H. Doorn
Universidad Nacional del Centro de la Provincia de Buenos Aires, Argentina

Laura C. Rivero
Universidad Nacional del Centro de la Provincia de Buenos Aires, Argentina

Information Science REFERENCE

INFORMATION SCIENCE REFERENCE

Hershey · New York

Director of Editorial Content: Kristin Klinger
Managing Editor: Jamie Snavely
Assistant Managing Editor: Carole Coulson
Cover Design: Lisa Tosheff
Printed at: Yurchak Printing Inc.

Published in the United States of America by
Information Science Reference (an imprint of IGI Global)
701 E. Chocolate Avenue, Suite 200
Hershey PA 17033
Tel: 717-533-8845
Fax: 717-533-8661
E-mail: cust@igi-global.com
Web site: http://www.igi-global.com/reference

and in the United Kingdom by
Information Science Reference (an imprint of IGI Global)
3 Henrietta Street
Covent Garden
London WC2E 8LU
Tel: 44 20 7240 0856
Fax: 44 20 7379 0609
Web site: http://www.eurospanbookstore.com

Library of Congress Cataloging-in-Publication Data

Handbook of research on innovations in database technologies and applications: current and future trends / Viviana E. Ferraggine, Jorge H. Doorn, and Laura C. Rivero, editors.

p. cm.

Includes bibliographical references and indexes.

Summary: "This book provides a wide compendium of references to topics in the field of the databases systems and applications"--Provided by publisher.

ISBN 978-1-60566-242-8 (hardcover) -- ISBN 978-1-60566-243-5 (ebook) 1. Database management. 2. Database design--Economic aspects. 3. Technological innovations. I. Ferraggine, Viviana E., 1961- II. Doorn, Jorge H., 1946- III. Rivero, Laura C., 1956-

QA76.9.D3H347326 2009

005.74--dc22

2008050170

British Cataloguing in Publication Data
A Cataloguing in Publication record for this book is available from the British Library.

List of Contributors

Table of Contents

Section V
Ontologies

Volume II

Section VI
Data Mining

Section VII
Physical Issues

Detailed Table of Contents

An entity relationship (ER) model that includes all the concepts of the original ER model and the additional concepts of generalizations/specializations and categories is often referred to as the extended ER (EER) model (Elmasri & Navathe, 2007). With the rising complexity of database applications, and in also light of today's web data applications (Necasky, 2006), the basic concepts of the ER model, as originally developed by Chen(1976), were no longer sufficient. Hence the basic ER model was extended to include generalizations and specializations (Bagui & Earp, 2003; Elmasri & Navathe, 2007), and the concept of categories (Elmasri, et al., 1985). This chapter sheds some light on these relationship concepts, concepts that database designers often find difficult to directly model (Engels et al., 1992/93). It also discuss the mapping rules for generalizations/specializations and categories. Important contributions in this area are also reported in (Elmasri et al., 1985; Gogolla & Hohenstein, 1991; Markowitz & Shoshani, 1992; Dey, et. al., 1999). Dullea, et. al. (2003) discusses the structural validity of modeling structures with ER models.

Bounded cardinality occurs when the cardinality of a relationship is within a specified range. Bounded cardinality is closely linked to symmetric relationships. This chapter describes these two notions, notes some of the problems they present, and discusses their implementation in a relational database.

Chapter VII

SDWs combine SDB and DW technologies for managing significant amounts of historical data that include spatial location. To better represent users' requirements for SDW applications, a conceptual model should be used. The advantages of using conceptual models are well known in database design. Nevertheless, the lack of a conceptual approach for DW and OLAP system modeling in addition to the absence of a commonly accepted conceptual model for spatial applications make the modeling task difficult. Existing conceptual models for SDBs are not adequate for DWs since they do not include the concepts of dimensions, hierarchies, and measures. Therefore, there is a need for extending multidimensional models by including spatial data to help users have a better understanding of the data to be analyzed.

Chapter VIII

This chapter refers to requirements specification and conceptual modeling phases for DW design. The author's proposal unifies the already existing approaches by giving an overall perspective of different alternatives available to designers when developing a DW.

Chapter IX

In data analysis process or data mining, it is necessary to know the nature of null values—the cases are by absence value, null value or default value -, being also possible and valid to have some imprecision, due to differential semantic in a concept, diverse sources, linguistic imprecision, element resumed in database, human errors, etc (Chavent, 1997). So, we need a conceptual support to manipulate these types of situations. This chapter describes symbolic data analysis (SDA), a new issue based on a strong conceptual model called symbolic object (SO).

Chapter X

Researchers in several areas (sociology, philosophy and psychology), among them Herbert Spencer and Abraham Maslow, attribute human actions resulting in continual environmental changes to the search for the satisfaction of individual and collective needs. Specifically in computer science, software engineering is a critical sub-area for these researches and their application (Lehman & Stenning, 1997), since it

involves the construction of models and orientation for their use in the development of resources, such as software, to support the user's needs. Databases should be included in this context as a component for data storage. Considering the premise of continuous changes and the human needs involved (Khan & Khang, 2004), the consequences for software and for the required database are obvious. In the field of computational science, these changes in the modern world are reflected in evolutionary features for software and databases, based on Database concepts, structures and processes that allow for rapid, albeit not traumatic, shifts to new industrial, commercial or scientific systems (Mcfadden et al., 1999) in new contexts (temporal scenarios) (Camolesi, 2004).

Chapter XI

Schema evolution is an important research topic with an extensive literature built up over the years. However, databases are still reluctant to change and thus their evolution is difficult to achieve because the evolution of the schema involves several issues at different levels of the database schema such as the change management at the logical level. Several approaches have been proposed to achieve the evolution of a schema of a wide range of types of databases. Versioning, modification and views are examples of these chosen approaches. This chapter presents and discusses one of these approaches, which is the versioning approach for database evolution. The future trends of the versioning are presented as well.

Chapter XII

With the introduction of evolutionary databases methodologies, the research area of schema evolution and versioning has been naturally broadening to embody new and more challenging research problems. In this chapter, borrowing the term from Ambler et al. (2006), we call this new research area evolutionary database. This chapter discusses the main issues concerning evolutionary database and then we survey several models and tools proposed for their solution.

Chapter XIII

This chapter deals with the problem of developing tools supporting the evolutionary data modeling process. First of all, the authors observe that the characteristics of the problem can be naturally framed in the agent paradigm, because the evolutionary data modeling can be seen as a process in active databases able to change their beliefs and structure. Moreover, the evolutionary data modeling can be compared to the design of an agent acting in an open environment: the environment can be represented by the

user needs and requirements (which change in an unforeseeable way), while the database development process is represented by the evolution of a reactive agent. Then, by following the AOSE (agent-oriented software engineering) view, we show that the use of tools and techniques from AI (artificial intelligence) can help facing the problem of developing supporting tools to automate evolutionary data modeling. To this end, after a brief introduction to the basic concepts in agent theory, and the highlighting of relationships among agents, software engineering, and databases, the authors point out the correspondence between agents and data modeling by showing a suitable architecture based on the logic of interrogation (Hintikka et al., 2002).

Chapter XIV

The objective of this entry is to introduce the problem of DW schema evolution, explaining current solutions and identifying open problems. It is organized as follows. First, background information, covering traditional solutions to the schema evolution problem, will be introduced. Then, research on conceptual evolution models and languages will be presented, comparing reference works to others. Open issues will be introduced. Finally, this entry will conclude with a summary

Chapter XV

This chapter provides an overall view of the state of the art in data warehouse model evolution. The authors present a series of comparison criteria and compare the different works. Moreover, the authors discuss the future trends in data warehouse model evolution.

Chapter XVI

In recent years, the spread of XML as the metalanguage for document modelling has been accompanied by a strong interest in XML document versioning. The interesting issue is that XML documents are no longer considered as atomic items that can be substituted or not, but composed of document nodes (elements) that can themselves be versioned. Besides, there have been several initiatives that propose using XML as the ideal format to represent metadata related with changes. Next, we revise the issues related with document versioning, the main approaches proposed and the issues that each approach favours. Issues related with XML will receive special attention in this updated chapterı. Versioning a document impacts not only the document itself but also other items, such as references from and to the versioned document, or the indexes created for information retrieval operations.

In this work, the model-driven development (MDD) approach has been considered to enhance the transformation rules of the conceptual schema into the relational schema. The relational model was considered in this work because most database methodologies are agreeing with it to transform the conceptual schema into a logical schema. A tool was plugged into rational rose to ensure this task. This tool can automatically generate maintaining mechanisms for these constraints to a target DBMS. Triggers system as maintaining mechanisms is used. These mechanisms can provide a fundamental base to obtain a very high level of knowledge independence (Paton, 1999). The triggers system is specified according to the recent SQL:2003 standard that revises all parts of SQL99 and adds new features (ISO Standard 2003).

While several research projects have contributed to different aspects of collaboration among software development environments during the past decade, little has been done on explicitly defining and modeling processes and environment artifacts involved in such partnerships. That is what this chapter is about. In the context of this study, environments work together by assigning tasks and sharing working methods. Tasks and working methods can be explicitly defined using process models. Process models, already the main focus in monolithic software development, will still be an important factor in our approach of collaborative software development. Because they are process-based, all software development environments considered here will be qualified in the continuation of process-sensitive software engineering environments (PSEEs).

Section II
Logical Modeling

The objective of this chapter is to introduce UML profiles for object-relational modeling. All such profiles exploit the extensibility mechanism of the UML. The author has chosen the profile used by Rational Rose Oracle8 tool as a representative one and has described it. Such profiles can be useful not only for manual object-relational database schema modeling but also for automated object to object-relational transformations in the MDA approach.

Chapter XXIV

A WfMS implemented on top of a database system is a special database application. Therefore, it helps to understand how advances on databases as a supporting technology may be applied to build more useful workflow applications and, on the other hand, how workflow application needs may drive the improvement of database technologies. Nevertheless, workflow applications are highly human-centered and should not be limited by technology. Organizational requirements have to be taken into consideration when developing workflow applications and in choosing an appropriate WfMS (Silva & Pinheiro, 2003). These requirements may impose restrictions that will never be completely solved by any supporting technology. For example, the most challenging kind of diversity is cultural diversity. It may be ameliorated using techniques to deal with semantic heterogeneity, but there is more to culture than differences in the interpretation of terms.

Chapter XXV

The shrinking of political party's membership and the role of mass media in shaping political communication and, to a large extent, the political agenda (Callaghan & Schnell, 2001), placing us in a era of perpetual election campaigning, may arguably be viewed as a drawback. Nevertheless, the role of technology is unavoidable and it may also facilitate life, being a motif of change and evolution (Dunleavy et. al., 2002). A first step is to acknowledge the importance of politically oriented applications and to design them taking politics as a prime aspect. By its nature, based on negotiation of positions, ideas and principles, politics is carried out through the exchange of information between the involved parties (Peled, 2001). In this sense databases are a key element for incorporating political aspects into system design. Recognizing the role of politics would help in designing databases that more appropriately capture the political needs of stakeholders. Issues like privacy, sensitive information, rule-making processes, and ownership could be shifted from a mere technical discussion to a proper political consideration.

Chapter XXVI

This chapter first presents a normative semantic model for enterprise information systems that has its roots in transaction processing information systems. The authors use this model because the majority of information processed and tracked by information systems is transactional in nature. The authors review empirical research on semantically modeled information systems and then provide an example company's semantic model as a proof of concept. The authors next discuss how this model can be applied to ERP systems and to inter-organizational systems and present future trends and research directions, and provide concluding comments.

Chapter XXX

Spatial query processing refers to the sequence of steps that a SDBMS will initiate to execute a given spatial query. The main target of query processing in the database field is to process the query accurately and quickly, by using both efficient representations and efficient search algorithms. Query processing in a spatial environment focuses on the design of efficient algorithms for spatial operators (e.g. selection operations, nearest neighbor search, spatial joins, etc.). Spatial query operations can be classified into five groups: point, range, spatial join, spatial aggregate and feature-based. For spatial query processing, the filter-refine paradigm is used over spatial access methods to minimize both the CPU and I/O cost. Future research trends include the study of new spatial queries (especially on spatial networks), the study of issues related to Web-Based Spatial Database Systems and work on cost models for estimating the selectivity of spatial queries.

Chapter XXXI

With the increasing demand for the geographical information expressed by the GIS users, the data enrichment processes play a key role to extend the dataset already stored within the system. In this context, the authors propose an approach to provide complementary data that enrich the descriptive aspect of the GDB. This chapter details the authors' semantic data enrichment approach. Furthermore, a refinement process was presented. It is provided to GIS users as a mean to describe with more accuracy the geographic entities and by the way to increase the chance to reach the pertinent documents.

Chapter XXXII

In many application domains, data are represented as a series of values in different time instances (time series). Examples include stocks, seismic signals, audio and many more. Similarity search in time series databases is an important research direction. Several methods have been proposed to provide efficient query processing in the case of static time series of fixed length. Research in this field has focused on the development of effective transformation techniques, the application of dimensionality reduction methods and the design of efficient indexing schemes. These tools enable the process of similarity queries in time series databases. In the case where time series are continuously updated with new values (streaming time series), the similarity problem becomes even more difficult to solve, since we must take into consideration the new values of the series. The dynamic nature of streaming time series precludes the use of methods proposed for the static case. To attack the problem, significant research has been per-

formed towards the development of effective and efficient methods for streaming time series processing. This chapter introduces the most important issues concerning similarity search in static and streaming time series databases, presenting fundamental concepts and techniques that have been proposed by the research community.

Internet map services (IMS) are redefining the ways in which people interact with geospatial information system (GIS) data. Driving forces behind this trend are the pervasiveness of GIS software and the emerging popularity of mobile devices and navigation systems utilizing GPS (global positioning system), as well as the ever-increasing availability of geospatial data on the Internet. These forces are also influencing the increasing need for temporal or real-time data. One trend that has become particularly promising in addressing this need is the development of IMS. IMS is changing the face of data access and creating an environment in which users can view, download, and query geospatial and real-time data into their own desktop software programs via the Internet. In this section, the authors will provide a brief description of the evolution and system architecture of an IMS, identify some common challenges related to implementing an IMS, and provide an example of how IMS have been developed using real-time weather data from the National Digital Forecast Database (NDFD). Finally, the authors will briefly touch on some emerging trends in IMS, as well as discussing the future direction of IMS and their role in providing access to real-time data.

Spatial networks databases are emerging from spatial and spatio-temporal databases as a distinct data management technology. This chapter reviews the motivation for developing techniques for the management of spatial networks, their fundamental concepts, and reports representative and recent research efforts and discusses possible future research directions. Although numerous research efforts have recently been devoted to spatial networks, significant research challenges still remain.

This chapter summarizes existing efforts from the database community to support LBSs in spatial networks. The focus of discussion is on the data models, data structures, and query processing techniques in SNDBs. An example application framework is presented to illustrate the relationship of these topics. Challenges and future trends are also addressed.

The demand for interoperability has boosted the development of standards and tools to facilitate data transformation and integration. Furthermore, this chapter focuses on interface standards as key to spatial data syntactical integration over the Web. Nevertheless, there are still many challenges to be met, especially those concerned with data semantics and harmonization of interoperating systems.

Section IV
Database Integrity

In a distributed database, the cost of accessing remote data for verifying the consistency of the database is the most critical factor that influences the performance of the system (Ibrahim, Gray & Fiddian, 2001). This chapter shows and proves through a simple analysis that selecting the suitable type of test can benefit the distributed database system, in which the amount of data transferred across the network is significantly reduced and the number of sites involved is always 1.

The authors of this chapter argue that inconsistency is far less harmful for database integrity than as suggested by commonly established results. They substantiate our claim by showing that, informally speaking, the consistent part of a possibly inconsistent database can be preserved across updates. More precisely, tjeu show that, if the simplified form of an integrity theory is satisfied, then each instance of each constraint that has been satisfied in the old state continues to be satisfied in the "new", i.e., updated state, even if the old database is not fully consistent. Therefore, such an approach can rightfully be called "inconsistency-tolerant". Yet, they also note that the use of inconsistency-tolerant integrity checking methods prevents an increase of inconsistency, and may even help to decrease it.

ing it (Redman, 1996; Eckerson, 2002). Today's quality measurements are typically driven by physical characteristics of the data (e.g., item counts, time tags, or failure rates) and assume an objective quality standard, disregarding the context in which the data is used. The alternative is to derive quality metrics from data content and evaluate them within specific usage contexts. The former approach is termed as structure-based (or structural), and the latter, content-based (Ballou and Pazer, 2003). This chapter proposes a novel framework to assess data quality within specific usage contexts and link it to data utility (or utility of data) - a measure of the value contribution associated with data within specific usage contexts. This utility-driven framework addresses the limitations of structural measurements and offers alternative measurements for evaluating completeness, validity, accuracy, and currency, as well as a single measure that aggregates these data quality dimensions.

A typical way to build surface numerical models or digital elevation models (DEM) for geographical information systems (GIS) is by processing the stereo images obtained from, for example, aerial photography or SPOT satellite data. These GIS can perform many computations involving their geographic databases. The quality control of a geographic database and in particular the topological and geometric integrity are, therefore, important topics (Guptill & Morrison, 1995; Harvey, 1997; Ubeda & Servigne, 1996; Laurini & Milleret-Raffort). The geometric quality control of the stored DEM is what we are concerned with here. «Quality» means the geometric accuracy measured in terms of the difference between a DEM and a reference DEM (R-DEM). We assume the R-DEM is a faithful model of the actual surface. Its point density may be greater than the DEM point density.

Studies performed on digital elevation models (DEM), obtained by means of photogrammetry, SPOT satellite images or other methods show that the precision in the z coordinate is different from the horizontal precision. In the case of the Interferometry SAR (IFSAR), the precision in the azimuth axis may be different from the precision in the range-axis. Moreover, the error in elevation is correlated with the error in the range axis. The method employed in the authors' other submission "*Geometric Quality in Geographic Information*" allows the evaluation of the DEM accuracy –vertical and horizontal- under some conditions of topographic unevenness. A reference DEM is required. In recent (Zelasco et al, 2001; Zelasco, 2002a) works it has been shown, by simulation, that the vertical error estimation is good and the horizontal error estimation is reasonably good depending on the surface roughness. In this chapter we show how to employ the quality control method when a DEM is obtained by IFSAR technology taking into account the corresponding hypotheses.

The motivation of this work stems from the observation that previously proposed approaches result not to be sound with respect to query answering when some peer is inconsistent. The idea proposed in this chapter consists in importing in each peer maximal sets of atoms not violating integrity constraints.

Section V
Ontologies

The current state of Web technology – the 'first generation' or 'syntactic' Web – gives rise to well known, serious problems when trying to accomplish in a non-trivial way essential tasks like indexing, searching, extracting, maintaining and generating information. These tasks would, in fact, require some sort of 'deep understanding' of the information dealt with: in a 'syntactic' Web context, on the contrary, computers are only used as tools for posting and rendering information by brute force. Faced with this situation, Tim Berners-Lee first proposed a sort of 'Semantic Web' where the access to information is based mainly on the processing of the semantic properties of this information: "… the Semantic Web is an extension of the current Web in which information is given well-defined meaning (emphasis added), better enabling computers and people to work in co-operation" (Berners-Lee et al., 2001: 35). From a technical point of view, the Semantic Web vision is deeply rooted into an 'ontological' approach, with some proper characteristics that differentiate it from the 'classical' approach to the construction of ontologies based on a methodology of the 'frame' type (Chaudhri et al., 1998) and on the use of tools in the 'standard' Protégé style (Noy, Fergerson and Musen, 2000). This chapter focuses on these characteristics.

Data migration relies on a schema matching process between the relational schema and the target ontology. Schema matching is considered as a task based on the fact that both schemata (relational and ontological) differ in structure, expressiveness and reasoning capability. This chapter proposes a methodology

for schema matching and present a tool, called RONTO (Relational to ONTOlogy), which deals with the semantic mapping between the elements of a relational schema to the elements of an ontological schema.

Chapter XLVIII

The ontology is used to define concepts, relationships and other distinctions that are relevant for modeling a domain where the specification takes the form of the definitions of representational vocabulary which provides meaning and constraints on the usage. Ontology is getting more and more popular as the description formalism for knowledge representation. There are many benefits of an ontology-based semantic data modeling over the traditional models. Ontology languages provide a universal, general description in standardized formalism. It has a logic foundation that supports reasoning and validation of the model. The ontology can be used as a general and extended semantic model in database design. On the other hand, it supports model validation, schema integration and extends the functionality of information retrieval.

Chapter XLIX

Inconsistency handling has been a prevailing task in important fields as the semantic web, data integration, and data cleaning. Several techniques are proposed, but the need of working with very large databases makes some of them unfeasible, especially those that are applied on the full KDB. In any case, the inconsistency is a persistent matter in the semantic web field. Due to that, several research teams are studying how to manage and control it.

Chapter L

This chapter presents the basic characteristics of the data integration systems, describing them in a simple way in order to enable an in depth study in this topic. The authors review the most important systems in this area, and divide them into two groups: traditional and ontology-based systems. The authors present

a comparative table in order to clarify the differences between the systems presented. Finally, they introduce some interesting issues being studied in this area, and which represent the future trends in it.

In this chapter we will focus on the use of ontologies because of their advantages when using for data integration. For example, an ontology may provide a rich, predefined vocabulary that serves as a stable conceptual interface to the databases and is independent of the database schemas; knowledge represented by the ontology may be sufficiently comprehensive to support translation of all relevant information sources; an ontology may support consistency management and recognition of inconsistent data; etc. Then, the next section will analyze several systems using ontologies as a tool to solve data integration problems.

In this chapter, we analyze several proposals that consider geographic information as sources to be integrated. First, we briefly describe basic concepts and conventions that will be used throughout this chapter. Following, an analysis is performed according to the use of ontologies in an integration process of geographic sources. Finally, future trends and conclusions are revealed as a consequence of our analysis.

Volume II

Domain experts use the concepts and terminology specific for their respective field of expertise, and different parameters and different languages to express their model of a concept. Therefore, very often different data sets can use different terms for the same kind of information. On the other hand, different data sets can use the same term for a completely different piece of information. Humans use their common sense, that is, their knowledge about the world, to translate the meaning of a foreign set of concepts and terms into their own terminology. Software systems usually do not have any knowledge about the world and have to explicitly be told how to translate one term into another. These problems can lead to serious conflicts during discovering and interpreting geographic data.

The main goal of this paper is to present the issue of the ontologies application in KDD. As a result of our research, we will propose a general ontology-based model, which includes all discovery steps. This paper is presented as follows: First, Background: main works in the field are introduced. Second, Main focus section is divided into: KDD Using Ontologies cycle in which we explain the knowledge process and propose a model, Domain Ontologies, Metadata Ontologies and Ontologies for Data Mining Process. Third: Future Trends, Conclusions, References and Key Terms.

Section VI
Data Mining

This chapter explains the problems involved in the design of an IQL and its associated evaluation techniques, and presents some solutions to those problems. Our proposal (Nieva-García & Benítez-Guerrero, 2006) of an extension to SQL for extracting decision rules of the form if <conditions> then <class> to classify uncategorized data and associated relational-like operator will be presented as a case study, and similar existing works will be overviewed. Future trends will then be introduced, and finally, the chapter will be concluded.

Privacy-preserving data mining emerged in response to two equally important (and seemingly disparate) needs: data analysis in order to deliver better services and ensuring the privacy rights of the data owners. Difficult as the task of addressing these needs may seem, several tangible efforts have been accomplished. In this article, an overview of the popular approaches for doing PPDM was presented, namely, suppression, randomization, cryptography, and summarization. The privacy guarantees, advantages, and disadvantages of each approach were stated in order to provide a balanced view of the state of the art. Finally, the scenarios where PPDM may be used and some directions for future work were outlined.

We have presented a survey of popular data clustering approaches, including both classic methods and recent advanced algorithms. The basic ideas of the approaches have been introduced and their characteristics analyzed. The techniques are designed for different applications and for different types of data, such as numerical data, categorical data, spatial data, text data and microarray data. The definitions of clusters in the algorithms are not always the same, and most of them favor certain types of clusters, such as sphere-shaped clusters, convex clusters and axis-parallel clusters. New definitions of clusters and novel techniques for clustering keep emerging as data mining is applied in new applications and in new fields.

Chapter LXI

Microarray data are often large and cumbersome to handle manually. Several algorithms have been developed to harness these data sets. We provided a brief description of hierarchical and nonhierarchical clustering methods, in addition to an implemented program C-MICRA. The authors are of the view that microarray data can be handled more efficiently using distance-based methods as opposed to parametric ones since the assumptions of parametric approaches have currently weak support. Finally, clustering techniques are data reduction methods and can be effective in detecting relevant genetic markers in microarray data.

Chapter LXII

The following section provides background information on the non-indexable Web and web databases. Though much research has emerged in recent years on querying web databases, there is still a great deal of work to be done. The deep Web will require more effective access techniques since the traditional crawl-and-index techniques, which have been quite successful for unstructured web pages in the publicly indexable Web, may not be appropriate for mostly structured data in the deep Web. Thus, a new field of research combining methods and research efforts of database and information retrieval communities may be created.

Chapter LXIII

In this entry, we have precisely formulated the problem of learning classifiers from distributed data and described a general strategy for transforming standard machine learning algorithms that assume centralized access to data in a single location into algorithms for learning from distributed data. The resulting algorithms are provably exact in that the hypothesis constructed from distributed data is identical to that obtained by the corresponding algorithm when it is used in the centralized setting. This ensures that the

This chapter focuses on sequential interestingness, which is an evaluation criterion of sequential patterns (Sakurai et al., 2008c). Also, this chapter focuses on 7 types of time constraints that are the background knowledge corresponding to the interests of analysts (Sakurai et al., 2008a). Lastly, this chapter introduces a discovery method based on the sequential interestingness and the time constraints.

Chapter LXVIII

In this paper, we will aim to find out a couple of authors who have never published together and who bear similar academic interests or study similar subjects. We will also show how the concepts of Yuan (Interdependent arising), Kong (Emptiness), Shi (Energy) and Guanxi (Relationship) in Chinese philosophy contribute to understand 'latent associations'. These four Chinese concepts are the theoretical basis of this paper. By explaining one by one what each concept is about we hope to tackle the two following questions: What do those four concepts exactly tell us? And how are they linked together? Finally, we will look at the empirical case study in scientometrics. We hope to show that this application of Chinese concepts can unravel latent associations between researchers in Database.

Section VII
Physical Issues

Chapter LXIX

In this chapter, four methods for constructing signature files are described. They are the sequential signature file, the bit-slice signature file, the S-tree and the signature tree. Among these methods, the signature file has the simplest structure and is easy to maintain, but slow for information retrieval. In contrast, the bit-sliced file and the S-tree are efficient for searching, but need more time for maintenance. In addition, an S-tree needs much more space than a sequential signature file or a bit-slice file. The last method, i.e., the signature tree structure, improves the S-tree by using less space for storage and less time for searching. Finally, as an important application, the signatures can be integrated into the top-down tree inclusion strategy to speed up the evaluation of containment queries. This can also be considered as a quite different way to organize a signature file.

Chapter LXX

In this chapter, a new algorithm is proposed for a kind of tree matching, the so-called twig pattern matching. This is a core operation for XML query processing. The main idea of the algorithm is to explore both T and Q bottom-up, by which each node q in Q is associated with a value (denoted $\delta(q)$) to indicate a node v in T, which has a child node v' such that $T[v']$ contains $Q[q]$. In this way, the tree embedding can be checked very efficiently. In addition, by using the tree encoding, as well as the subsumption checking mechanism, we are able to minimize the size of the lists of the matching query nodes associated with the nodes in T to reduce the space overhead. The algorithm runs in $O(|T|Q_{leaf})$ time and $O(T_{leaf}Q_{leaf})$ space, where T_{leaf} and Q_{leaf} represent the numbers of the leaf nodes in T and in Q, respectively. More importantly, no costly join operation is necessary.

Chapter LXXI

We reviewed the problem of clustering XML documents and related research work from different perspectives, namely, the representation models, the kind of information used to extract document features, and the tasks. We then motivated the role of semantic relatedness in the context of the classification of XML collections and presented our approach to the problem of clustering semantically related XML documents. Semantic integration and classification of XML data will be one of the most challenging problems for database researchers in semistructured data management to tackle in the near future.

Chapter LXXII

The aim of this chapter is to present an overview of state-of-the-art XML indices and to discuss the main issues, trade-offs, and future trends in XML indexing. Furthermore, since XML is gaining importance for representing business data for analytics (Beyer, Chamberlin, Colby, Özcan, Pirahesh & Xu, 2005), we also present an index we developed specifically for XML data warehouses.

Chapter LXXIII

This chapter discusses the benefits of managing business documents and Web content within the context of an integrative information systems architecture. This architecture incorporates database management, document and Web content management, integrated scanning/imaging, workflow and capabilities for integration with other technologies.

Many applications such as military tracking, medical monitoring, stock arbitrage system, network management, aircraft control, factory automation etc. that depend heavily on database technology for the proper storage and retrieval of data located at different remote sites have certain timing constraints associated with them. Such applications introduce the need for distributed real time database systems (DRTDBS) [Ramamritham, 1993]. The implementation of DRTDBS is difficult due to the conflicting requirements of maintaining data consistency and meeting distributed transaction's deadlines. The difficulty comes from the unpredictability of the transactions' response times [Huang, 1991]. Due to distributed nature of the transactions and in presence of other sources of unpredictability such as data access conflicts, uneven distribution of transactions over the sites, variable local CPU scheduling time, communication delay, failure of coordinator and cohort's sites etc., it is not easy to meet the deadline of all transactions in DRTDBS [Kao & Garcia – Monila, 1995]. The unpredictability in the commitment phase makes it more serious because the blocking time of the waiting cohorts due to execute-commit conflict may become longer. Hence, due to unique characteristics of the committing transactions and unpredictability in the commitment process, design of an efficient commit protocol is an important issue that affects the performance of DRTDBS [Shanker, Misra & Sarje, 2006d].

Chapter LXXIX

Here, a new distributed real time commit protocol (MECP)has been presented that uses a new locking scheme. The new locking scheme ensures that a borrower can't be a lender simultaneously at the same site and the same data can not be used by another borrower simultaneously as compared to PROMPT and 2SC, where there is a need for checking this. It not only optimizes the storage cost but also considers blind and update type writes collectively for DRTDBS. The simulation results show that the protocol performs better than PROMPT and 2SC commit protocols. It is well suited to data intensive applications where transaction arrival rate is high and the sizes of transactions are large.

Chapter LXXX

This chapter addresses the issue of self-tuning DBMS. In the remainder of the chapter we present a background on this topic, followed by a discussion focusing on performance, indexing and memory issues. Then, we highlight future trends and conclude the chapter.

Chapter LXXXI

Database replication had commonly used lazy protocols in commercial DBMSs, ensuring thus good performance, but without guaranteeing full replica consistency. This may lead to some transaction losses in case of failure. To overcome these problems, eager replication protocols with group communication support have been proposed in the last decade. The use of total order broadcast protocols has allowed the development of new kinds of eager replication: weak-voting and certification-based techniques. Such solutions are able to ensure one-copy serializability with a performance similar to lazy protocols, with better (although still limited) scalability and lower abort rates. However, these solutions are not applied to commercial database management systems, yet.

Chapter LXXXII

This chapter presents a real-time dynamic crash recovery scheme (RTDCRS) suitable for a DRTM-MDBS on the basis of considering carefully the timing constraint characteristics of data and transaction. The rest of the chapter is organized as follows: Section 3 first analyzes the recovery requirements of a DRTMMDBS and then gives the recovery correctness criteria suitable for a DRTMMDBS. In Section 4, we propose an RTDCRS and prove its correctness. Section 5 gives performance evaluation of a RTDCRS. In Section 6, we conclude the proposed RTDCRS. Finally, in Section 7, we discuss future and emerging trends.

Chapter LXXXIII

This chapter is organized in two parts. In the first part, we provide an overview, where we (1) define and characterize QDNs as a new family of data networks with common characteristics and applications, and (2) review possible database-like architectures for QDNs as query processing systems and enumerate the most important QDN design principles. In the second part of the article, as the first step toward realizing the vision of QDNs as complex distributed query-processing systems, we focus on a specific problem, namely the problem of effective data location (or search) for efficient query processing in QDNs. We briefly explain two parallel approaches, both based on techniques/models borrowed from the complex system theory, to address this problem.

The information is stored on the web and on computers mostly in full-text format. The current databases are able to store and manage huge document collection. Full-text data sources require specific search operations. Database management systems usually contain a separate full-text search engine to perform full-text search primitives. In general, the current FTS engines support the following functionalities: stemming, synonym and thesaurus based matching, fuzzy matching and Boolean operators. It has been shown that the average user requires additional help to exploit the benefits of these extra operators. Current research focuses on solving the problem of covering new document formats, adapting the query to the user's behavior, and providing an efficient FTS engine implementation.

Most Web developers underestimate the risk and the level of damage that might be caused when Web applications are vulnerable to SQL (structured query language) injections. Unfortunately, Web applications with such vulnerability constitute a large part of today's Web application landscape. This article aims at highlighting the risk of SQL injection attacks and provides an efficient solution.

Preface

Database technologies have a rich history of relevant developments immersed in a continuous evolution and consolidation process. Even more, during the last decades, they have evolved in such a way that almost all main software applications and modern information systems have a database as a core component. The information stored is usually accessed and manipulated by many application programs to perform business processes. In this sense, databases in any organization have provoked a profound impact and significant endeavors in their operability, and business assessments.

Moreover, data are one of the most valuable assets of any organization and the design of database applications is a factor of vital influence regarding the efficiency and manageability of their information systems. The extraordinary growth and widespread application of databases has reached a vast diversity of users with their own fields of development, their particular application requirements, and their own technological needs. In recent years, these facts have promoted the appearance of new interdisciplinary investigation areas. It is worthy of mention distributed real-time systems, data integration based on ontologies, collaborative software development, databases on the Web, spatio-temporal databases, multimedia databases, new scopes of database programming languages, and the appearance of new characteristics related to data quality, indexing and reengineering, among others.

A database management systems (DBMS) contributes to these objectives by providing data persistence, efficient access and data integrity. By isolating the conceptual schema from the implementation schema, database systems guarantee data independence from storage techniques and offer Standard Query Language (SQL) which is the query language per excellence. In addition, by means of the management of users and their privileges, the DBMS can provide safe control access to data. While the control of concurrent access to data is managed through different protocols of transaction scheduling and varied locking techniques, backups and database recovery strategies allow database recovering after hardware or software failures. These capabilities—among others—have opened wide research fields exciting challenges, and major technological and conceptual changes in many features through their evolution.

The "*Handbook of Research on Innovations in Database Technologies and Applications: Current and Future Trends*" provides a new and comprehensive knowledge compendium on databases. This handbook pulls together many relevant issues that researchers and practitioners have investigated, proposed or observed to solve diverse real-world problems with the help of databases, and provides a wide compilation of references to topics in the field of database systems and applications.

Since knowledge about databases and their entire environment has become an essential part of any education in computer science and the main subject of many research groups at universities and institutes all over the world, this handbook is an ideal source of knowledge for students, researchers, programmers, and database developers who may need speedy and reliable information, and authoritative references to current database areas, latest technologies and their practical applications. This handbook provides many articles offering coverage and definitions of the most important issues, basic concepts, trends, and

technologies in database field along with some papers presenting a theoretical foundation of relevant topics in such field.

This handbook is intended for a wide range of readers including computing students having basic knowledge on databases; teachers in charge of introductory and advanced courses on databases; researchers interested in specific areas related to their research, and practitioners facing database implementation choices. Curious and inexperienced readers will also find in this handbook many interesting articles, opening the gate to an invaluable knowledge about databases principles and novel applications. Experienced teachers will find a comprehensive compendium of teaching resources. The main endeavor of this handbook has been to grant access to essential core material for education, research and practice on database systems.

The handbook is composed by 93 articles from authoritative database researchers, focusing on object-oriented applications, multimedia data storing and management; also, on new fields of applications such as geospatial and temporal information systems, data warehousing and data mining, design methodologies, database languages and distributed databases, among other topics. Emerging areas that are becoming particularly mature are also faced. They include the integration of DBMSs into the World Wide Web; the effective support to the decision-making process in an organizational environment; the information visualization and the high performance database systems.

This "*Handbook of Research on Innovations in Database Technologies and Applications: Current and Future Trends*" is a collaborative effort addressing the endeavors that raise the increasing need of improving the storage of information, the adaptation or adherence of conceptual modeling and design to newer paradigms, and the development of advanced applications related to the Internet, e-commerce, data warehousing and data mining. Leading specialists in each area; researchers with a vast experience on the topics covered by this volume; experts in the development of database systems in organizational environments; and teachers with accumulated experience teaching graduate and undergraduate courses have contributed with valuable chapters on their fields of expertise.

This handbook has been built as a compilation of papers with a quasi-standardized structure. Many articles may be included into more than one group, but the arrangement was made taking into account their main area. Interested readers will be able to compose their own groups by gathering articles which share keywords or term definitions. It differs from the typical databases books in that it offers a quite balanced treatment of the most relevant characteristics, languages and definitions of the core terms in each subject. Many articles offer an analytical approach so that the concepts presented can serve as the basis for the specification of future systems. Many articles have plenty of examples to show readers how to apply that material.

On the other hand, each article offers a profuse set of recommended references to current and settled literature on each topic. The "*Handbook of Research on Innovations in Database Technologies and Applications: Current and Future Trends*" presents a sound grounding in the foundations of database technology and the state of the art; also, it covers other areas which are under exceptional development and spreading. Thus, the articles in this volume include a list with the key terms and concepts relevant to each topic along with their definitions. We have to thank our authors for the careful selection of terms they have made.

Section I: Conceptual Modeling

This first section groups a set of articles dealing with relevant topics related to conceptual modeling, current and traditional models, formal specifications, new paradigms and data warehousing. Among other subjects, this section includes original work on the entity relational model, completeness of the

information, capture of requirements for data warehousing, symbolic objects, temporary data, post-relational data models and data reengineering.

Sikha Bagui is the author of "*Mapping Generalizations and Specializations and Categories to Relational Databases*". This paper discusses the implementation of generalizations and specializations in relational databases, along with the application of the concept of inheritance, in the context of the extended entity relationship (EER) model.

In "*Bounded Cardinality and Symmetric Relationships*", Norman Pendegraft gives an overview on bounded cardinality and its links with symmetric relationships, highlighting some of the problems they present, and discussing their implementation in a relational database.

The article "*A Paraconsistent Relational Data Model*" by Navin Viswanath and Rajshekhar Sunderraman deals with the Closed World Assumption and the Open World Assumption. The first assumption is based on the completeness of the information stored in the database. Consequently, if a fact is not in the database, then its negation is true; under the second assumption, such negation should be explicitly stored to become true; otherwise nothing can be said about it. This article introduces a data model which is a generalization of the relational data model: "*The Paraconsistent Relational Data Model*".

The paper "*Managing Temporal Data*" by Abdullah Uz Tansel reviews the issues in modeling and designing temporal databases based on the relational data model. Also, it addresses attribute time stamping and tuple time stamping techniques.

Richard C. Millham contributed with an article entitled "*Data Reengineering of Legacy Systems*", which provides an overview on the transformation of legacy data from a sequential file system to a relational database, outlining the methods used in data reengineering to transform the program logic that access the database using wide spectrum language (WSL) as the intermediate representation of programs.

The three following articles have been authored by Elzbieta Malinowski. These contributions are tightly coupled as they deal with the MultiDim model which is a conceptual multidimensional model used for representing data requirements for data warehousing (DW) and on line analysis processing (OLAP) applications. The article "*Different Kinds of Hierarchies in Multidimensional Models*" describes the MultiDim model, showing its abilities to denote fact relationships, measures, dimensions, and hierarchies, for which novel classification is provided. Malinowski considers that DWs and OLAP can use relational storage, and presents how hierarchies can be mapped to the relational model. In "*Spatial Data in Multidimensional Conceptual Models*", additional characteristics of the MultiDim conceptual model are explored. It is extended providing spatial support for different elements such as levels and other measures. These characteristics are explored in the context of a platform-independent conceptual model. Finally in "*Requirement Specification and Conceptual Modeling for Data Warehouses*", Malinowski presents a proposal to cope with the lack of a methodological framework to guide developers through the different stages of the data warehouse design process. This proposal refers to the requirements specification and conceptual modeling phases for data warehouses design unifying already existing approaches by giving an overall perspective of the different alternatives available to designers.

In "*Principles on Symbolic Data Analysis*", Héctor Oscar Nigro and Sandra Elizabeth González Císaro revise the history, sources and fields of influence of Symbolic Data, providing formal definitions and semantics applied to such novel concept. They discuss how to handle null values, internal variations and rules using the symbolic data analysis approach which is based on the symbolic object model.

Six contributions written by different authors address the field of database evolution. Luiz Camolesi Júnior and Marina Teresa Pires Vieira coauthored the article "*Database Engineering Supporting the Data Evolution*". Their contribution surveys on database evolution as a vast subject under constant discussion and innovation, focusing on the evolutionary process, and the different ways in which it can be approached. Regarding that schema evolution is a key research topic with an extensive literature built

up over the years, the article summarizes why database evolution itself becomes hard to manage, and describes some proposed approaches to manage the evolution of a schema for a wide range of types of databases. Another approach for the evolution of databases can be found in "*Versioning Approach for Database Evolution*", written by Hassina Bounif, who analyzes versioning-based courses of action taking into account that the versioning principles can be applied universally to many different forms of data. The next four articles are framed into two main approaches: Schema Evolution approach and Schema Versioning approach. The following two are authored by Vincenzo Deufemia, Giuseppe Polese, and Mario Vacca. The first article, "*Evolutionary Database: State of the Art and Issues*" the authors focus on the recent introduction of evolutionary database methodologies which broaden the schema evolution and versioning problems to a wider vision highlighting new and more challenging research problems. The second one has been entitled "*Interrogative Agents for Data Modeling*" and examines the problem of evolutionary data modeling process. The authors present the characteristics of this subject in the frame of the agent paradigm, as the evolutionary data modeling can be seen as a process in active databases able to change their beliefs and structure. Moreover, following the agent-oriented software engineering (AOSE) view, the authors show that the use of tools and techniques from artificial intelligence (AI) can help to face the problem of developing supporting tools to automate evolutionary data modeling. The other two papers address schema evolution models in the context of data warehouses. "*Schema Evolution Models and Languages for Multidimensional Data Warehouses*" coauthored by Edgard Benítez-Guerrero and Ericka-Janet Rechy-Ramírez provides a deep analysis of both approaches reviewing recent research results on the subject, while the article "*A Survey of Data Warehouse Model Evolution*" written by Cécile Favre, Fadila Bentayeb and Omar Boussaid compares both approaches using three different criteria: functionality, deployment and performance.

"*Document Versioning and XML in Digital Libraries*" by M. Mercedes Martínez-González, is devoted to digital libraries. The author analyzes the issues related to document versioning and main existing approaches, together with their pros and cons. Also, she discusses how digital libraries mirror the traditional library and how they provide more services than those available in paper document libraries.

In the framework of the model-driven development (MDD) approach, Harith T. Al-Jumaily, Dolores Cuadra and Paloma Martínez contributed with the article "*MDD Approach for Maintaining Integrity Constraints in Databases*". The authors analize the semantic losses produced when logical elements are not coincident with conceptual elements –with especial emphasis on multiplicity constraints– and how to fix them, proposing a trigger system as the maintaining mechanism.

Pierre F. Tiako in his article entitled "*Artifacts for Collaborative Software Development*" provides an overview on collaborative software development analyzing modeling processes, and also, environment artifacts involved.

Other contributions dealing with ***Conceptual Modeling*** aspects can be found in the section ***Logical Modeling (Section II)***: "*Horizontal Data Partitioning: Past, Present and Future*" by Ladjel Bellatreche and Database Reverse Engineering by Jean-Luc Hainaut, Jean Henrard, Didier Roland, Jean-Marc Hick and Vincent Englebert and in the section ***Ontologies (Section V):*** "*Ontologies Application to Knowledge Discovery Process in Databases*" by Héctor Oscar Nigro and Sandra Elizabeth González Císaro and "*Ontology-Based Semantic Models for Databases*", by László Kovács, Péter Barabás and Tibor Répási.

Section II: Logical Modeling

For several decades, data modeling has been an aspect of the database world that has received many contributions from researchers and also important feedback from practitioners. Subjects as data modeling evolution, versioning, reverse engineering, and the impact of novel applications have driven research-

ers and practitioners to revisit well-established approaches to address the challenges such subjects are raising. This section contains valuable contributions focusing on such aspects.

In "*Object-Relational Modeling*", Jaroslav Zendulka shows how an object-relational database schema can be modeled in Unified Modeling Language (UML). Firstly, the author clarifies the fact that UML contains no direct support: neither for capturing important features of relational databases nor for specific features of object-relational databases. Regarding the fact that such features are necessary for modeling data stored in a relational database and objects stored in an object-relational database at design levels subsequent to the conceptual one, the author describes an extension of UML which adds the ability to model effectively and intelligibly such features in this kind of databases.

"*Concept-Oriented Model*" by Alexandr Savinov reviews concept-oriented model (CoM), an original approach to data modeling he has recently introduced. Its major goal consists in providing simple and effective means for the representation and manipulation of multidimensional and hierarchical data while retaining the possibility to model the way data is physically represented.

From a tutorial perspective, in the article "*Database Reverse Engineering*", Jean-Luc Hainaut, Jean Henrard, Didier Roland, Jean-Marc Hick and Vincent Englebert describe the problems that arise when trying to rebuild the documentation of a legacy database as long as the methods, techniques and tools that may be used to solve these problems.

"*Imprecise Functional Dependencies*" is a paper coauthored by Vincenzo Deufemia, Giuseppe Polese and Mario Vacca that overviews imprecise functional dependencies and provides a critical discussion of the dependencies applicable to fuzzy and multimedia data.

The article "*Horizontal Data Partitioning: Past, Present and Future*" by Ladjel Bellatreche is devoted to the analysis of the issues of horizontal data partition, the process of splitting access objects into sets of disjoint rows. This analysis ranges from the former utilizations –logically designing databases efficiently accessed– to the recent applications in the context of distributed environments and data warehouses.

Two contributions authored by Francisco A. C. Pinheiro are devoted to analyze the interaction of novel applications of database systems and the improvement of technologies and paradigms. The article "*Database Support for Workflow Management Systems*" provides an interesting overview on the relationships between database technologies and workflow issues. In this regard, the author addresses the discussion on how advances on databases as a supporting technology may be applied to build more useful workflow applications and how workflow application needs may drive the improvement of database technologies. On the other hand, the article "*Politically Oriented Database Applications*" deals with how technology pervades every aspect of modern life, having an impact on the democratic life of a nation and frequently, being an object of dispute and negotiation. These facts affect the way politics is done, by shaping new forms of planning and performing political actions. Applications used in or related to politics are information intensive, making databases a prime element in building politically oriented applications. In this article, Francisco A. C. Pinheiro discusses some aspects of database related technology necessary for this kind of applications.

Facing the need to integrate information efficiently, organizations have implemented enterprise resource planning (ERP) systems. Much of the value of these ERP systems resides in their integrated database and its associated data warehouse. Unfortunately, a significant portion of the value is lost if the database is not a semantic representation of the organization. Taking into account such negative aspect, Cheryl L. Dunn, Gregory J. Gerard, and Severin V. Grabski have coauthored the article "*Semantically Modeled Databases in Integrated Enterprise Information Systems*" focusing on the resources-events-agents (REA) ontology.

"*The Linkcell Construct and Location-Aware Query Processing for Location-Referent Transactions in Mobile Business*" contributed by James E. Wyse, describes location-qualified business information

for the provision of location-based mobile business. This information –contained in a locations repository– and its management –performed by a locations server– are the focal concerns of this article.

Hagen Höpfner is the author of "*Caching, Hoarding, and Replication in Client/Server Information Systems with Mobile Clients*". This paper presents a complete set of exact definitions of the caching, hoarding and replication techniques for handling redundant data in information systems with mobile clients in relation to the level of autonomy of mobile devices/users. Furthermore, the author explains the terms cache replacement, cache invalidation, cache maintenance, automated hoarding, and synchronization of replicated data.

Section III: Spatial and Temporal Databases

Databases handling temporal, spatial or both types of data are becoming more and more frequently used every day. Temporal data contains some references, attributes, or structures where time plays a role; the same happens with spatial data. The integration of factual with temporal and / or spatial data to be able to handle geographical information systems (GIS), location based services, all kind of mapping services or weather services require a profound understanding of the special needs such integration demands. This section contains several specialized contributions related to these matters.

Two contributions written by the same authors, deal with the processing of spatial temporal databases. The first one, "*Spatio-Temporal Indexing Techniques*" by Michael Vassilakopoulos and Antonio Corral, surveys the indexing of moving points and other spatio-temporal information, considering recent research results and possible research trends within this area of raising importance. The second one, "*Query Processing in Spatial Databases*" by Antonio Corral and Michael Vassilakopoulos specifically focuses on spatial query processing.

Khaoula Mahmoudi and Sami Faïz in "*Automatic Data Enrichment in GIS Through Condensate Textual Information*" propose a modular approach to enrich data stored in a geographic database (GDB), by extracting knowledge from on-line textual documents corpora. This is accomplished by using a distributed multi-document summarization. A refinement step to improve the results of the summarization process based on thematic delimitation, theme identification, delegation and text filtering is also proposed.

From a tutorial perspective, Maria Kontaki, Apostolos N. Papadopoulos and Yannis Manolopoulos in their article "*Similarity Search in Times Series*" introduce the most important issues concerning similarity search in static and streaming time series databases, presenting fundamental concepts and techniques.

"*Internet Map Services and Weather Data*", a contribution by Maurie Caitlin Kelly, Bernd J. Haupt and Ryan E. Baxter, provides a brief overview of the evolution and system architecture of internet map services (IMS), identifying some challenges related to the implementation of such service. The authors provide an example of how IMS have been developed using real-time weather data from the National Digital Forecast Database (NDFD).

The two following contributions address subjects related to spatial network databases. The first one, "*Spatial Network Databases*" by Michael Vassilakopoulos reviews the motivation behind the development of techniques for the management of spatial networks and their fundamental concepts. Additionally, the author reports the most representative and recent research efforts and discusses possible future research. The second one "*Supporting Location-Based Services in Spatial Network Databases*", contributed by Xuegang Huang, summarizes existing efforts from the database community to support location-based services (LBSs) in spatial networks, focusing the discussion on the data models, data structures, and query processing techniques.The author considers a prototype service that finds the *k* nearest neighbors to a mobile user in the network.

Laura Díaz, Carlos Granell, and Michael Gould focus on the interoperability problem from a syntactic point of view. In their article "*Spatial Data Integration Over the Web*", they propose the use of

interface standards as a key to spatial data integration over the Web. For that purpose, they report on the Geography Markup Language (GML) standard that provides spatial services with common data models for spatial data access and interchange of representation between spatial and non-spatial data with an XML-based format.

Other contributions dealing with ***Spatial and Temporal Databases*** aspects can be found in the section ***Conceptual Modeling (Section I)***: "*Managing Temporal Data*" by Abdullah Uz Tansel, in the section ***Ontologies (Section V)***: "*Mediation and Ontology-Based Framework for Interoperability*" by Leonid Stoimenov and in the section ***Physical Issues (Section VII)***: "*Querical Data Networks*" by Cyrus Shahabi and Farnoush Banaei-Kashani.

Section IV: Database Integrity

Database integrity is known to be important from the earliest days in the database history. At first glance, it could be said that database implementations have traded data redundancy or access flexibility to the data stored by new integrity requirements. When such flexibility reaches distributed databases or context aware applications, the integrity management needs to be increased again. It may be also true that future paradigms will raise new integrity issues. Several contributions related to integrity constraint checking, fault tolerant integrity control, and several points of views on data quality are included in this section.

In "*Improving Constraints Checking in Distributed Databases with Complete, Sufficient, and Support Tests*", Ali Amer Alwan, Hamidah Ibrahim and Nur Izura Udzir analyze the performance of the checking process in a distributed environment when various types of integrity tests are considered. Authors select the most suitable test for each situation in terms of the amount of data transferred across the network and the number of sites involved during the process of checking the constraints.

The paper "*Inconsistency-Tolerant Integrity Checking*" by Hendrik Decker and Davide Martinenghi highlights the fact that integrity checking is practically unfeasible for significant amounts of stored data without a dedicated approach to optimize the process. The authors give a fresh perspective by showing that if the simplified form of an integrity theory is satisfied then, each instance of each constraint that has been satisfied in the old state continues to be satisfied in the updated state even if the old database is not fully consistent. They rightfully call this approach "inconsistency-tolerant".

"*Merging, Repairing, and Querying Inconsistent Databases*", the article contributed by Luciano Caroprese and Ester Zumpano, introduces a framework for merging, repairing and querying inconsistent databases, investigating the problem of the satisfaction of integrity constraints implemented and maintained in commercial DBMS in the presence of null values. The authors also establish a new semantics for constraints satisfaction.

"*The Challenges of Checking Integrity Constraints in Centralized, Distributed and Parallel Databases*" by Hamidah Ibrahim surveys on the vital problem of guaranteeing database consistency, highlighting several factors and issues as regards preventing semantic errors made by the users due to their carelessness or lack of knowledge.

The following two contributions examine quality of data issues. The first one "*Data Quality Assessment*" by Juliusz L. Kulikowski, tackles the basic problems of data quality assessment, assuming that for high information processing systems' effectiveness high quality of data is a the main requirement. In "*Measuring Data Quality in Context*", the authors Gunesan Shankaranarayanan and Adir Even propose a framework to assess data quality within specific usage contexts linking it to data utility. The utility of data is conceived by these authors as a measure of the value associated with data within specific usage contexts.

Two related contributions sharing a couple of authors are devoted to data integrity in Geographical Information Systems. The first one, "*Geometric Quality in Geographic Information*" by José Francisco

Zelasco, Gaspar Porta and José Luís Fernandez Ausinaga proposes a method to evaluate the geometric integrity of digital elevation models (DEM) obtained by different techniques. In the second one, "*Geometric Quality in Geographic Information IFSAR DEM Control*", José Francisco Zelasco, Judith Donayo, Kevin Ennis and José Luis Fernandez Ausinaga consider Interferometry SAR (IFSAR) techniques and the stochastic hypotheses that are specific according to the particular geometry involved in this technique.

"*Querying and Integrating P2P Deductive Databases*" contributed by Luciano Caroprese, Sergio Greco and Ester Zumpano considers the integration of information and the computation of queries in an open-ended network of distributed peers. This proposal is based on a change in the perception of inconsistent peers, accepting data answering queries from those peers if it comes from the consistent part of the peer.

Other contributions dealing with ***Database Integrity*** aspects can be found in the section ***Conceptual Modeling (Section I)***: "*Bounded Cardinality and Symmetric Relationships*" by Norman Pendegraft, "*MDD Approach for Maintaining Integrity Constraints in Databases*" by Harith T. Al-Jumaily, Dolores Cuadra and Paloma Martínez and "*A Paraconsistent Relational Data Model*" by Navin Viswanath and Rajshekhar Sunderraman and in the section ***Ontologies (Section V)***: "*Inconsistency, Logic Databases and Ontologies*" co-authored by José A. Alonso-Jiménez, Joaquín Borrego-Díaz and Antonia M. Chávez-González.

Section V: Ontologies

Interaction between the fields of ontologies and databases may be produced in several ways. Ontologies may be used or required to understand the context of the future database applications being the foundation of the database schema. A database may be used as a repository for large ontologies. Ontologies may be also used to ease the integration of heterogeneous and distributed databases. This section holds articles dealing with different interactions between ontology and databases.

The article "*Using Semantic Web Tools for Ontologies Construction*" by Gian Piero Zarri describes the proper characteristics of the ontological approach that support the Semantic Web, differentiating it from the 'classical' approach of the construction of ontologies based on a methodology of the 'frame' type and on the use of tools in the 'standard' Protégé style.

In "*Matching Relational Schemata to Semantic Web Ontologies*", Polyxeni Katsiouli, Petros Papapanagiotou, Vassileios Tsetsos, Christos Anagnostopoulos and Stathes Hadjiefthymiades propose a methodology for schema matching and present a tool called RONTO (Relational to ONTOlogy). This tool deals with the semantic mapping between the elements of a relational schema to the elements of an ontological schema, in the context of data migration.

"*Ontology-Based Semantic Models for Databases*" by László Kovács, Péter Barabás and Tibor Répási shows and explains the importance and the role of ontologies in design processes, regarding the recognition that ontologies have achieved as the description formalism for knowledge representation.

"*Inconsistency, Logic Databases, and Ontologies*" is an article co-authored by José A. Alonso-Jiménez, Joaquín Borrego-Díaz, and Antonia M. Chávez-González. The authors base their contribution on the fact that to work with very large databases makes certain techniques for inconsistency handling not applicable. They discuss how in the semantic web future trends must study verification techniques based on a sound and limited testing, aided by a powerful automated theorem prover. These techniques need a deep analysis of the behavior of automated theorem provers having great autonomy because a slanted behavior may produce deficient reports about inconsistencies in the knowledge database (KDB). For these authors, the most promising research line in this field is the design and development of tools that allow explaining the source of anomalies detected in ontologies.

The following four articles focus on the fact that geographical information systems (GIS) are increasingly moving away from monolithic systems towards the integration of heterogeneous and distributed information systems. This interoperability problem forces to deal with a diversity of data sets, data modeling concepts, data encoding techniques and storage structures. Furthermore, a problem of semantic heterogeneity arises: different data sets usually have discrepancy in the terms they use. Software systems do not have "common sense" –as humans do- to deal with these discrepancies. Software systems usually do not have any knowledge about the world, leading to serious conflicts while discovering and interpretating data. "*Data Integration: Introducing Semantics*" is a tutorial article contributed by Ismael Navas-Delgado and José F. Aldana-Montes in which the reader will find a simple description of the basic characteristics of the data integration systems and a review of the most important systems in this area (traditional and ontology-based), along with a table highlighting the differences between them. Two contributions written by the same authors also address the issues related to data integration from a tutorial point of view. Both articles are devoted to the use of ontologies in the integration process, noting the advantages they bring to such process. The first article "*Overview of Ontology-Driven Data Integration*" by Agustina Buccella and Alejandra Cechich deals with a wider perspective considering general purpose Database Systems, while in the second article "*Current Approaches and Future Trends of Ontology-Driven Geographic Integration*" the focus is on geographic data. Two main problems are addressed in this case. The first one is how to combine the geographical information available in two or more geographical information systems with different structures, and the second one is related to the differences in the points of views and vocabularies used in each geographical information system.

Leonid Stoimenov, author of "*Mediation and Ontology-Based Framework for Interoperability*" considers the interoperability problem in the context of geographical information systems (GIS). In his article, Stoimenov introduces a common interoperability kernel called ORHIDEA (ontology-resolved hybrid integration of data in e-applications) consisting of three key components: semantic mediators, translators/wrappers, and a shared server.

In "*Ontologies Application to Knowledge Discovery Process in Databases*", the authors Héctor Oscar Nigro and Sandra Elizabeth González Císaro discuss the application of ontologies in KDD, and propose a general ontology-based model, which includes all discovery steps.

Other contribution dealing with ***Ontologies*** aspects can be found in ***Physical Issues (Section VII)***: "*Full-Text Manipulation in Databases*", by László Kovács and Domonkos Tikk.

Section VI: Data Mining

As data analysis techniques must process large amounts of data efficiently, special attention has been paid to new trends such as: evaluation techniques, safeguard of sensitive information and cluster techniques. Data mining is an interdisciplinary area of research, having its roots in databases, machine learning, and statistics. Several entries reporting many research efforts and main results in this field can be read in this section.

Edgard Benítez-Guerrero and Omar Nieva-García describe the problems involved in the design of an inductive query language and its associated evaluation techniques, and present some solutions to such problems in their article "*Expression and Processing of Inductive Queries*". They also present a case study based on their proposal of an extension to SQL for extracting decision rules of the form *if <conditions> then <class>* to classify uncategorized data, and associated relational-like operators.

"*Privacy Preserving Data Mining*" (PPDM), an article contributed by Alexandre Evfimievski and Tyrone Grandison, reviews PPDM as the area of data mining that seeks safeguarding sensitive information from unsolicited or unsanctioned disclosure.

In "*Mining Frequent Closed Itemsets for Association Rules*", Anamika Gupta, Shikha Gupta, and Naveen Kumar discuss the importance of mining frequent closed itemsets (FCI) instead of frequent itemsets (FI) in association rule discovery procedure, and explain different approaches and techniques for mining FCI in datasets.

"*Similarity Retrieval and Cluster Analysis Using R*-Trees*" contributed by Jiaxiong Pi, Yong Shi, and Zhengxin Chen examines time series data indexed through R*-Trees. The authors also study the issues of retrieval of data similar to a given query, and the clustering of the data based on similarity.

The paper entitled "*Outlying Subspace Detection for High-dimensional Data*" by Ji Zhang, Qigang Gao, and Hai Wang, gives an overview on the detection of objects that are considerably dissimilar, exceptional and inconsistent with respect to the majority of the records in an input database (outliers) and their outlying subspaces, i.e. subspaces in high-dimensional datasets in which they are embedded.

Two other contributions address data clustering issues. Clustering is one of the most important techniques in data mining. It is a tool to discover similar objects into different groups or non-overlapping clusters so that the data in each group shares commonality, often proximity, according to some defined distance measure. "*Data Clustering*" by Yanchang Zhao, Longbing Cao, Huaifeng Zhang, and Chengqi Zhang provides a wider view on the clustering problem presenting a survey of popular approaches for data clustering, including well-known clustering techniques, such as partitioning clustering, hierarchical clustering, density-based clustering and grid-based clustering; also recent advances, such as subspace clustering, text clustering and data stream clustering. The second contribution by Emmanuel Udoh and Salim Bhuiyan "*C-MICRA: A Tool for Clustering Microarray Data*", focuses on clustering as an important unsupervised method in the exploration of expression patterns in gene data arrays.

"*Deep Web: Databases on the Web*", authored by Denis Shestakov makes valuable background information on the non-indexable Web and web databases available, surveying on the recent concept of Deep Web.

Doina Caragea and Vasant Honavar have contributed the article "*Learning Classifiers from Distributed Data Sources*" whose purpose is to precisely define the problem of learning classifiers from distributed data and summarize recent advances that have led to a solution to this problem. They describe a general strategy to transform standard machine learning algorithms—that assume centralized access to data in a single location—into algorithms to learn from distributed data.

The article "*Differential Learning Expert System in Data Management*" by Manjunath R. is devoted to problems related to knowledge acquisition for expert systems, and the analysis of plausible solutions for some of them. In this sense, the author exposes that a system using a rule-based expert system with an integrated connectionist network could benefit from the advantages of connectionist systems, regarding that machine-learning helps towards knowledge acquisition. The article presents a system based on rule-based expert system with neural networks which are able to perform a "learning from example" approach to extract rules from large data sets.

"*Machine Learning as a Commonsense Reasoning Process*" written by Xenia Naidenova concentrates on one of the most important tasks in database technology which is to combine the activities of inferring knowledge from data (data mining) and reasoning on acquired knowledge (query processing). The article includes a proposal of a unified model of commonsense reasoning, and also a demonstration showing that a large class of inductive machine learning (ML) algorithms can be transformed into the commonsense reasoning processes based on well-known deduction and induction logical rules.

"*Machine Learning and Data Mining in Bioinformatics*" is a contribution coauthored by George Tzanis, Christos Berberidis, and Ioannis Vlahavas. In this article, the authors review the exponential growth of biological data and the new questions these data have originated, due to recent technological advances. In particular, they focus on the mission of bioinformatics as a new and critical research

domain, which must provide the tools and use them to extract accurate and reliable information in order to gain new biological insights.

The contribution "*Sequential Pattern Mining from Sequential Data*" overviews sequential pattern discovery methods from discrete sequential data. Its author, Shigeaki Sakurai, focuses on sequential interestingness, which is an evaluation criterion of sequential patterns, highlighting that there are 7 types of time constraints that are the background knowledge related to the interests of analysts.

The field of scientometrics has been looking at the identification of co-authorship through network mapping. In a similar context, the paper entitled "*From Chinese Philosophy to Knowledge Discovery in Databases: A Case Study: Scientometric Analysis*" by Pei Liu explores the latent association of two authors, i.e. the collaboration between two researchers which has not yet occurred but might take place in the future. The author also shows how the concepts of Yuan (Interdependent arising), Kong (Emptiness), Shi (Energy) and Guanxi (Relationship) in Chinese philosophy contribute to understand 'latent associations', bringing in this way an original approach which could be applicable to the database research community.

Other contributions dealing with ***Data Mining***, data warehousing and knowledge acquisition aspects can be found in the section ***Conceptual Modeling (Section I)***: "*Schema Evolution Models and Languages for Multidimensional Data Warehouses*" by Edgard Benítez-Guerrero, Ericka-Janet Rechy-Ramírez, "*A Survey of Data Warehouse Model Evolution*" by Cécile Favre, Fadila Bentayeb and Omar Boussaid, "*Principles on Symbolic Data Analysis*" by Héctor Oscar Nigro and Sandra Elizabeth González Císaro and three articles "*Different Kinds of Hierarchies in Multidimensional Models*", "*Spatial Data in Multidimensional Conceptual Models*" and "*Requirement Specification and Conceptual Modeling for Data Warehouses*" by Elzbieta Malinowski, in the section ***Spatial and Temporal Databases (Section III)***: "*Automatic Data Enrichment in GIS Through Condensate Textual Information*" by Khaoula Mahmoudi, and Sami Faïz, "*Similarity Search in Times Series*" by Maria Kontaki, Apostolos N. Papadopoulos and Yannis Manolopoulos and "*Current Approaches and Future Trends of Ontology-Driven Geographic Integration*" by Agustina Buccella and Alejandra Cechich, in the section ***Ontologies (Section V)***: "*Ontologies Application to Knowledge Discovery Process in Databases*" by Héctor Oscar Nigro and Sandra Elizabeth González Císaro and in the section ***Physical Issues (Section VII)***: "*Index and Materialized View Selection in Data Warehouses*" by Kamel Aouiche and Jérôme Darmont, "*Full-Text Manipulation in Databases*" by László Kovács and Domonkos Tikk "*Synopsis Data Structures for Representing, Querying, and Mining Data Streams*" and "*Innovative Access and Query Schemes for Mobile Databases and Data Warehouses*" both by Alfredo Cuzzocrea.

Section VII: Physical Issues

The increasing number of database paradigms, database applications, types of data stored and database storing techniques leads to several new physical issues regarding storage requirements, information retrieval and query processing. New indexing techniques, document clustering, materialized views, commit protocols, data replications and crash recovery issues are partial but important answers to these concerns, among many others. This section contains several research reports and tutorials on the state of art of physical issues.

In "*An Overview on Signature File Techniques*", Yangjun Chen presents an overview on recent relevant research results on information retrieval, mainly on the creation of database indexes which can be searched efficiently for the data under seeking. The focus of this article is on signature techniques.

The following contributions deal with efficiency on XML Databases. The article by Yangjun Chen, "*On the Query Evaluation in XML Databases*", presents a new and efficient algorithm for XML query

processing, reducing the time and space needed to satisfy queries. In the article "*XML Document Clustering*", Andrea Tagarelli provides a broad overview of the state-of-the-art and a guide to recent advances and emerging challenges in the research field of clustering XML documents. Besides basic similarities criteria based on structure of the document, the article focus is on the ability of clustering XML documents without assuming the availability of predefined XML schemas. Finally, "*Indices in XML Databases*", a contribution by Hadj Mahboubi and Jérôme Darmont presents an overview of state-of-the-art XML indexes, discusses the main issues, tradeoffs and future trends in XML indexing and, since XML is gaining importance for representing business data for analytics, it also presents an index that the authors specifically developed for XML data warehouses.

In the article "*XML Document Clustering*", Andrea Tagarelli provides a broad overview of the state-of-the-art and a guide to recent advances and emerging challenges in the research field of clustering XML documents. Besides basic similarities criteria based on structure of the document, the article focus is on the ability of clustering XML documents without assuming the availability of predefined XML schemas.

"*Integrative Information Systems Architecture: Document & Content Management*", an article submitted by Len Asprey, Rolf Green, and Michael Middleton, overviews benefits of managing business documents and Web content within the context of an integrative information systems architecture which incorporates database management, document and Web content management, integrated scanning/imaging, workflow and capabilities of integration with other technologies.

The contribution by Kamel Aouiche and Jérôme Darmont, "*Index and Materialized View Selection in Data Warehouses*", presents an overview of the major families of state-of-the-art index and materialized view selection methods; discusses the issues and future trends in data warehouse performance optimization, and focuses on data mining-based heuristics to reduce the selection problem complexity.

"*Synopsis Data Structures for Representing, Querying, and Mining Data Streams*", contributed by Alfredo Cuzzocrea, provides an overview of state-of-the-art of synopsis data structures for data streams, making evident the benefits and limitations of each of them in efficiently supporting representation, query, and mining tasks over data streams.

"*GR-OLAP: On Line Analytical Processing of GRid Monitoring Information*", the article contributed by Julien Gossa and Sandro Bimonte deals with the problem of management of Grid networks. The authors discuss recent advances in Grid monitoring, proposing the use of data warehousing and on line analytical processing to mine Grid monitoring information to get knowledge about the Grid networks characteristics.

"*A Pagination Method for Indexes in Metric Databases*", a paper by Ana Villegas, Carina Ruano, and Norma Herrera, proposes an original strategy for metric databases whose index and/or data do not fit completely in the main memory. This strategy adapts the metric database regarding the capacity of the main memory, instead of adapting the index to be efficiently handled in secondary memory.

"*SWIFT: A Distributed Real Time Commit Protocol*", an article submitted by Udai Shanker, Manoj Misra, and Anil K. Sarje, introduces a protocol to reduce the time to reach the commit in some specific situations, in the context of distributed databases.

"*MECP: A Memory Efficient Real Time Commit Protocol*", coauthored by Udai Shanker, Manoj Misra, and Anil K. Sarje presents the problem of handling huge databases in the context of real time applications. In both situations, any saving in main memory usage becomes very important. In this article, the design of a distributed commit protocol which optimizes memory usage is presented.

The article "*Self-Tuning Database Management Systems*" has been contributed by Camilo Porto Nunes, Cláudio de Souza Baptista, and Marcus Costa Sampaio and addresses the issue of self-tuning DBMS, presenting a background on this topic followed by a discussion centered on performance, indexing and memory issues.

The article "*Database Replication Approaches*" contributed by Francesc Muñoz-Escoí, Hendrik Decker, José Armendáriz, and José González de Mendívil revise different approaches tackling the problem of database replication management. The authors analyze new replication techniques that were introduced for databases—as an evolution of the process replication approaches found in distributed systems.

"*A Novel Crash Recovery Scheme for Distributed Real-Time Databases*", is a contribution by Yingyuan Xiao that reports research results into the crash recovery strategy area for distributed real-time main memory database systems (DRTMMDBS), including real-time logging scheme, local fuzzy checkpoint and dynamic recovery processing strategy.

The article "*Querical Data Networks*" (QDN) by Cyrus Shahabi and Farnoush Banaei-Kashani defines and characterizes QDNs as a new family of data networks with common characteristics and applications. It also reviews possible database-like architectures for QDNs as query processing systems and enumerates the most important QDN design principles. The authors also address the problem of effective data location for efficient query processing in QDNs, as the first step toward comprehending the vision of QDNs as complex distributed query-processing systems.

"*On the Implementation of a Logic Language for NP Search and Optimization Problems*" an article by Sergio Greco, Cristian Molinaro, Irina Trubitsyna, and Ester Zumpano, presents the logic language NP Datalog. It is a restricted version of DATALOG¸ to formulate NP search and optimization problems which admits only controlled forms of negation such as stratified negation, exclusive disjunction and constraints and enables a simpler and intuitive formulation for search and optimization problems. In this contribution, a solution based on the rewriting of logic programs into constraint programming is proposed.

The article by Alfredo Cuzzocrea, "*A Query-Strategy-Focused Taxonomy of P2P IR Techniques*", presents a taxonomy of the state-of-the-art of peer-to-peer (P2P) systems-information retrieval (IR) techniques, with emphasis on the query strategy used to retrieve information and knowledge from peers; and shows similarities and differences among the techniques.

In their contribution "*Pervasive and Ubiquitous Computing Databases: Critical Issues and Challenges*", the authors Michael Zoumboulakis and George Roussos offer a survey on the dual role that databases have to play in Pervasive and Ubiquitous Computing. In the short-term, they need to provide the mapping between physical and virtual entities and space in a highly distributed and heterogeneous environment while in the long term database management systems need to provide the infrastructure for the development of data-centric systems.

The following two contributions, written by the Christoph Bussler, deal with business integration. The former "*Business-to-Business (B2B) Integration*" surveys on how B2B integration is absolutely essential for business and organizations not only to stay competitive but also keep or even gain market share. The latter "*Enterprise Application Integration (EAI)*" by the same author surveys on current developments and critical issues of enterprise application integration (EAI) technologies, as they are essential for enterprises with more than one back end application system.

In a world in which globalization is increasingly integrating the economies and societies, products created in one nation are often marketed to a range of international consumers. Cross-border interactions on social and professional levels have been facilitated by the rapid diffusion of online media, however, different cultural expectations can cause miscommunication within this discourse paradigm. Localization has thus become an important aspect of today's global economy. "*The Role of Rhetoric in Localization and Offshoring*", a contribution by Kirk St.Amant, focuses on these issues, examining localization in offshoring practices that could affect database creation and maintenance.

In "*Adaptive XML-to-Relational Storage Strategies*", Irena Mlynkova provides an overview of existing XML-to-relational storage strategies. This paper examines their historical development and provides a more detailed discussion of the currently most promising ones—the adaptive methods.

"*Innovative Access and Query Schemes for Mobile Databases and Data Warehouses*" authored by Alfredo Cuzzocrea presents a critical discussion on several aspects of mobile databases and data warehouses, along with a survey on state-of-the-art data-intensive mobile applications and systems. The Hand-OLAP system, a relevant instance of mobile OLAP systems is also described.

The paper "*Full-Text Manipulation in Databases*", by László Kovács and Domonkos Tikk overviews issues and problems related to full-text search (FTS). The authors' aim is to elucidate about the needs of users which usually require additional help to exploit the benefits of the functionalities of the FTS engines, such as: stemming, synonym and thesaurus based matching, fuzzy matching and Boolean operators. They also point out that current research focuses on solving the problem of covering new document formats, adapting the query to the user's behavior, and providing an efficient FTS engine implementation.

"*Bind but Dynamic Technique: The Ultimate Protection Against SQL Injections*" a contribution by Ahmad Hammoud and Ramzi A. Haraty, explores on the risk and the level of damage that might be caused when web applications are vulnerable to SQL injections, and provides an efficient solution.

Other contributions dealing with ***Physical Issues*** can be found in the section ***Conceptual Modeling (Section I)***: "*Document Versioning and XML in digital libraries*" by M. Mercedes Martínez-González, in the section ***Spatial and Temporal Databases (Section III)***: "*Spatio-Temporal Indexing Techniques*" by Michael Vassilakopoulos and Antonio Corral and "*Query processing in spatial databases*" by Antonio Corral and Michael Vassilakopoulos, in the section ***Ontologies (Section V)***: "*Mediation and Ontology-Based Framework for Interoperability*" by Leonid Stoimenov and "*Similarity Retrieval and Cluster Analysis Using R*-Trees*" by Jiaxiong Pi, Yong Shi, and Zhengxin Chen and in the section ***Data Mining (Section VI)***: "*Managing Temporal Data*" by Abdullah Uz Tansel.

Summing up, this handbook offers an interesting set of articles about the state of the art of fundamental database concepts and a unique compilation of chapters about new technologies, current research trends, and challenging applications addressing the needs of present database and information systems.

Acknowledgment

We would like to thank Mehdi Khosrow-Pour for this invaluable opportunity. The edition of this handbook has been an exiting experience. We would also like to acknowledge the IGI Global staff, mainly to Jan Travers, Michelle Potter, Julia Mosemann, Kristin Roth, and Megan Childs, for their professional guidance and continuous support.

We would like to thank each of the authors for their insights and valuable contributions to this handbook.

We also want to thank all of the people who helped us in the reviewing process of this handbook. First of all, many thanks to the contributors who have done a significant job refereeing the articles from their colleagues providing constructive and comprehensive comments, and also as editors, we would like to specially acknowledge the help of all external reviewers involved in such process:

Liliana Favre, *Universidad Nacional del Centro, Argentina*
Mariana del Fresno, *Universidad Nacional del Centro, Argentina*
Laura Felice, *Universidad Nacional del Centro, Argentina*
Carmen Leonardi, *Universidad Nacional del Centro, Argentina*
Virginia Mauco, *Universidad Nacional del Centro, Argentina*
Marcela Ridao, *Universidad Nacional del Centro, Argentina*
Francesco Gullo, *Universita della Calabria, Italy*
Sergio Flesca, *Universita della Calabria, Italy*
Alexandros Nanopoulos, *Aristotle University of Thessaloniki, Greece*
Panos Vassiliadis, *University of Ioannina Greece*
Zaher Al Aghbari, *University of Sharjah, UAE*
Kanalugu Chandra Sekharaiah, *JNTU College of Engineering, Kakinada, Andhra Pradesh, India*
Yücel Saygin, *Sabanci University, Turkey*
Alban Gabillon, *Laboratoire GePaSud, Université de la Polynésie Française, France*
Karla A. V. Borges, *Prodabel Belo Horizonte, Brazil*
Faruk Polat, *Middle East Technical University Ankara, Turkey*
Mario Cannataro, *University "Magna Græcia" of Catanzaro, Italy*
Pierangelo Veltri, *University "Magna Græcia" of Catanzaro, Italy*
Woojong Suh, *College of Business Administration, Inha University, Korea*
Reda Alhajj, *University of Calgary, Canada*
Esko Marjomaa, *University of Joensuu, Finland*
Lu Yan, *Åbo Akademi, Finland*

Menzo Windhouwer, *Theoretische Taalwetenschap (UvA), The Netherlands*
Mathias Meixner, *Thales-e-Transactions GmbH, Bad Herfeld, Germany*
Antonio Badía, *University of Louisville, USA*
Richa Singh, *West Virginia University, USA*
Clodoveu Davis, *Universidade Federal de Minas Gerais, Brazil*
Nora Reyes, *Universidad de San Luis, Argentina*
José Hernández Orallo, *Universitat Politècnica de València, Spain*
María del Mar Roldán, *Universidad de Málaga, Spain*
Mohammad Dadashzadeh, *Oakland University, USA*
Hong Va Leong, *Hong Kong Polytechnic University, Hong Kong*
Andrea Rodríguez, *Universidad de Concepción, Chile*
Alkis Simitsis, *LABS HP, Intelligent Information Management Lab*
Prasad M. Deshpande, *University of Wisconsin, USA*
Kjetil Nørvåg, *Norwegian University of Science and Technology (NTNU), Norway*
Boanerges Aleman-Meza, *Polytechnic University of Victoria, Mexico*
Mariano Cilia, *Technische Universität Darmstadt, Germany*
Gladys Kaplan, *Universidad Nacional de La Matanza, Argentina*

Finally, the editors want to thank their colleagues, friends, and family for the encouragement and company received during this endeavor.

Viviana E.Ferraggine, Jorge H. Doorn, and Laura C. Rivero
Tandil, Buenos Aires, Argentina
October 2008

Section I
Conceptual Modeling

Chapter I
Mapping Generalizations and Specializations and Categories to Relational Databases

Sikha Bagui
University of West Florida, USA

INTRODUCTION

An Entity Relationship (ER) model that includes all the concepts of the original ER model and the additional concepts of generalizations/specializations and categories is often referred to as the Extended ER (EER) model (Elmasri & Navathe, 2007). With the rising complexity of database applications, and also in light of today's web data applications (Necasky, 2006), the basic concepts of the ER model, as originally developed by Chen(1976), are no longer sufficient. Hence the basic ER model has been extended to include generalizations and specializations (Bagui & Earp, 2003; Elmasri & Navathe, 2007), and the concept of categories (Elmasri, et al., 1985). In this short article we shed some light on these relationship concepts, concepts that database designers often find difficult to directly model (Engels et al., 1992/93). We also discuss the mapping rules for generalizations/specializations and categories. Important contributions in this area are also reported in (Elmasri et al., 1985; Gogolla & Hohenstein, 1991; Markowitz & Shoshani, 1992; Dey, et. al., 1999). Dullea, et. al. (2003) discusses the structural validity of modeling structures with ER models.

Due to the short nature of this paper, we will keep the discussion in this paper focused on implementing generalizations and specializations in relational databases; their parallel implementation in objects will not be covered. Also, the discussion of the concept of inheritance will center around generalizations/specializations and categories in EER diagrams, without getting into an almost equivalent notion in Object-oriented (OO) theory, ORM (Object-Role Modeling) and UML (Unified Modeling Language) class diagrams.

BACKGROUND

A generalization/specialization relationship models a superclass/subclass type of relationship. A generalization is an abstraction mechanism that allows for viewing of entity-sets as a single generic entity-set. The attributes and associations which are common to the entity-sets are associated with the generic (generalized) entity-set. The inverse of generalization is called specialization.

GENERALIZATION / SPECIALIZATION RELATIONSHIPS

If we are modeling a hospital database, for example, and we want to store information about the hospital's nurses, technicians, and physician assistants, we could create separate entities such as NURSE, TECHNICIAN and PHYSICIAN ASSISTANT. But, these three entities would also have a lot of fields in common, for example, name, social security number, address, phone, etc. may be common to all three entities. So, it would be a good idea to have an entity set called EMPLOYEE containing these common fields, and entity subsets, NURSE, TECHNICIAN and PHYSICIAN ASSISTANT, that could inherit this information from the EMPLOYEE entity set. In this case the EMPLOYEE entity set would be called the ***superclass***. This superclass is a ***generalization*** of the entity subsets, NURSE, TECHNICIAN and PHYSICIAN ASSISTANT. The NURSE, TECHNICIAN and PHYSICIAN ASSISTANT would be called the ***subclasses***. The subclasses are ***specializations*** of the superclass, EMPLOYEE, and inherit from the superclass. Several specializations can be defined for the same entity type (or superclass).

The subclass, denoted by a separate entity rectangle in the EER diagram, is considered to be a *part* of the superclass entity set, EMPLOYEE. Although it will have attributes that distinguish it from other subclasses, it is considered only a subset of the EMPLOYEE entity set. That is, all nurses are employees, but the reverse is not true - not all employees are nurses. Likewise, all technicians or physician assistants are employees, but all employees are not technicians or physician assistants.

Figure 1 shows this generalization/specialization example. We use Elmasri and Navathe's (2007) diagrammatic notations for the EER diagrams. The subset symbol, "$\subset$", indicates the direction of the superclass/subclass or parent-child, inheritance relationship. This superclass/subclass relationship is also often referred to as a ***IS-A*** or ***IS-PART-OF*** relationship (Sanders, 1995).

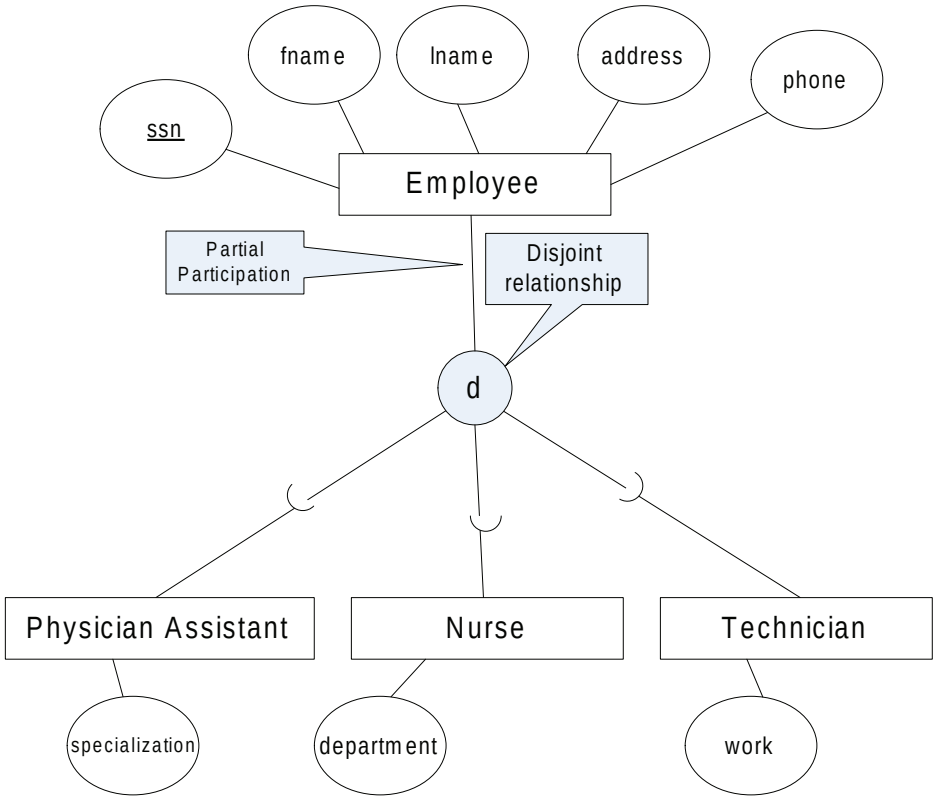
Constraints on Generalization/ Specialization Relationships

Generalizations and specializations can have two types of constraints: (i) the ***disjoint/overlap*** relationship constraint, and, (ii) participation constraints – total or partial. The combinations of these constraints can be: (i) disjoint and total participation; (ii) disjoint and partial participation; (iii) overlap and total participation; (iv) overlap and partial participation. First we will discuss disjoint/overlap relationship constraints and then we will discuss participation constraints, giving examples of combinations of the constraints along the way.

Disjoint/Overlap Relationship Constraints

Generalization/specialization relationships may be ***disjoint*** or they may ***overlap***. A disjoint relationship is shown by a "d" in the circle attaching the superclass to the subclass or subclasses (as shown in Figure 1). A disjoint relationship means that an entity from the superclass can belong to only one of the subclasses (can be of only one specialization). For example, according to figure 1, an EMPLOYEE can be at most a member of only one of the subclasses – PHYSICIAN ASSISTANT,

Figure 1. A generalization/specialization relationship



NURSE, or TECHNICIAN. An employee cannot be a physician assistant as well as a nurse, or, cannot be a nurse as well as a technician.

An overlap relationship is shown by an "o" in the circle attaching the superclass to the subclass or subclasses (as shown in Figure 4). Overlap means that an entity from the superclass can belong to more than one subclass (specialization). For example, according to Figure 4, a computer must be either a laptop or a desktop, or both a laptop and a desktop.

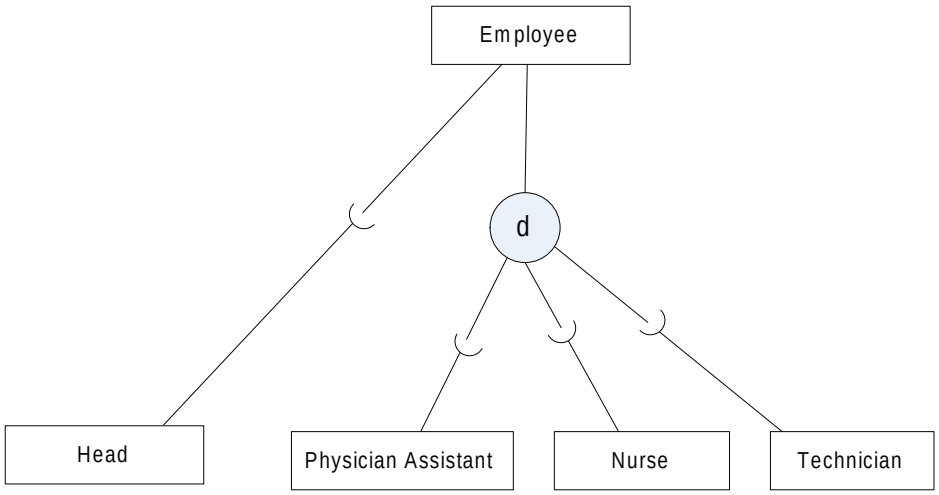
Participation Constraints

The second type of constraint on generalization/specialization relationships is participation constraints, which may be ***total*** (or full) or ***partial***. As in the ER model (Bagui & Earp, 2003; Elmasri & Navathe, 2007), we will show a full or total participation between the superclass and subclass entities by double lines, and a partial participation between the superclass and subclass entities by single lines. Partial participation is shown in Figure 1. Figure 1 can be read as:

*An EMPLOYEE **may** either be a PHYSICIAN ASSISTANT, NURSE or TECHNICIAN.*

Figure 1 shows a *disjoint, partial participation* relationship. The "may" means partial participation between the EMPLOYEE superclass and the respective subclasses entities. That is, not all employees of the EMPLOYEE entity set belong one of the subclasses, PHYSICIAN ASSISTANT, NURSE or TECHNICIAN. One may ask, why? Or how? To answer this, we will extend figure 1 to include another subclass, as shown in Figure 2. We now have an Employee from the EMPLOYEE

Figure 2. A disjoint



entity set who may also belong to the HEAD subclass. Once again, the "may" is inferred from the single line between the superclass (or generalization) entity, EMPLOYEE, and the subclass (or specialization) entity, HEAD. Figure 2 can be read as:

*An Employee **may** be a HEAD or PHYSICIAN ASSISTANT or NURSE or TECHNICIAN. Or, an Employee **may** be both a HEAD and a PHSYCIAN ASSISTANT, or both a HEAD and a NURSE, or both a HEAD and a TECHNICIAN.*

An example of total or full participation is shown in Figure 3. We can read Figure 3 as:

*An EMPLOYEE **must** either be a PHYSICIAN ASSISTANT, NURSE or TECHNICIAN.*

The "must" means total participation. So, an EMPLOYEE must belong to one of the subclasses. That is, ***all*** employees of the EMPLOYEE entity set must belong to one of the subclasses. But although there is total participation in Figure 3, the employee cannot belong to more than one subclass because of the "d" or disjoint relationship. Figure 3 shows a *disjoint, full participation* relationship and Figure 4 shows an *overlap, full participation* relationship.

Mapping Generalizations and Specializations to a Relational Database

Rules to map generalizations/specializations to a relational database depend on the constraints on the generalization/specialization relationships. One of the following four rules are generally used (Elmsari & Navathe, 2007) to map generalizations and specializations to a relational database:

Rule 1:

Rule 1 works well for total or partial generalization/specialization relationships as well as disjoint or overlap generalization/specialization relationships. Using this rule, we would create a separate relation for the superclass as well as for each of the subclasses.

For rule 1: (i) Create a relation for the superclass entity. Include all the attributes of this entity in this relation and underline the primary key attribute.

Figure 3. A disjoint and full participation with predicate defined subclasses

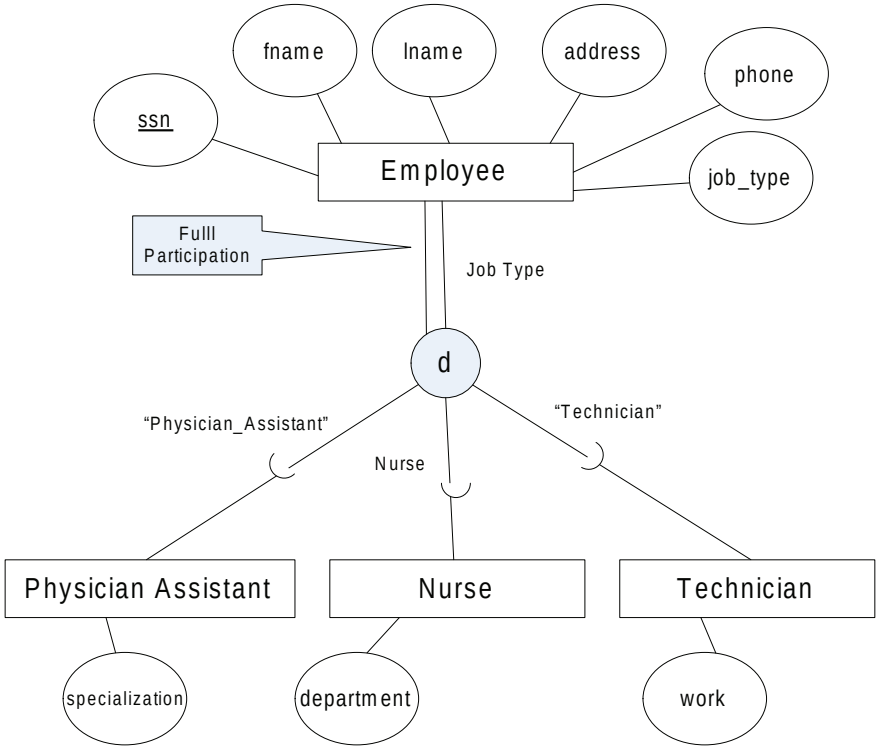
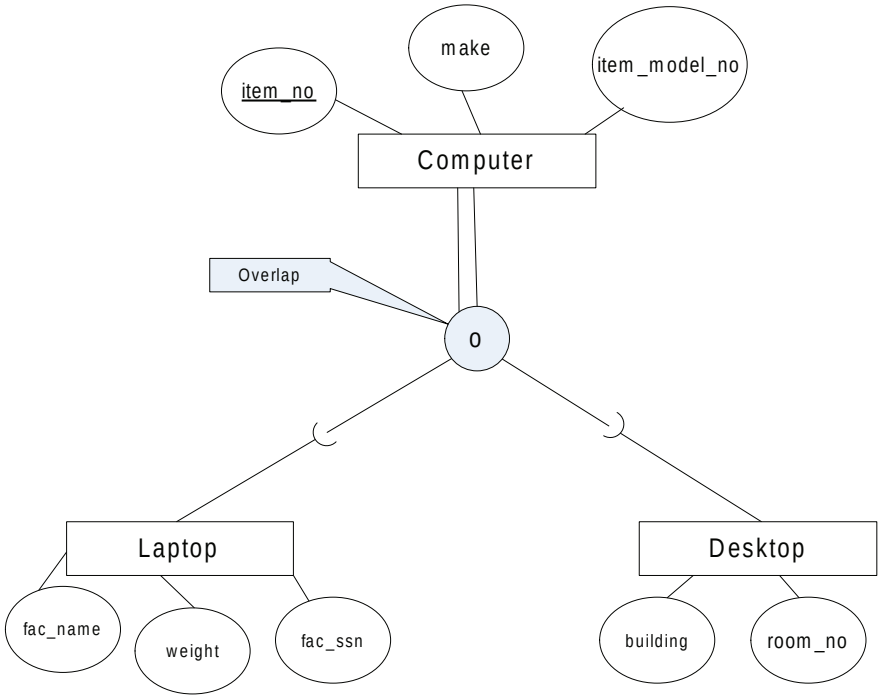


Figure 4. An overlap and full participation



(ii) Create a separate relation for each subclass (specialization) entity. Include the attributes of the respective subclasses in the respective subclass relations. Include the primary key from the superclass entity or relation in the subclass relation as the primary key of the subclass relation (and underline it).

To illustrate this rule we will map Figure 1 as shown below:

EMPLOYEE

ssn	fname	lname	address	phone

PHYSICIAN_ASSISTANT

ssn	specialization

NURSE

ssn	department

TECHNICIAN

ssn	work

Rule 2:

Rule 2 works well if: (a) there is total or full participation between the superclass and the subclasses. That is, if every member of the superclass entity set belongs to at least one of the subclasses; (b) if there is a disjoint relationship – otherwise there will be redundancy if a superclass entity belongs to more than one subclass; and, (c) when there is more emphasis on the subclass (specialization) and it is more important to know about the subclass and it's attributes. By this rule we create a separate relation for each subclass. In this rule you do not have a separate relation for the superclass entity.

For rule 2: Create a separate relation corresponding to each subclass entity. Include the attributes of the respective subclasses in the respective subclass relations. Also include the primary key and other attributes of the superclass entity in all the subclass relations. Underline the primary key brought from the superclass entity (to the subclass relation).

To illustrate this rule we will map figure 1 as shown below (but we are assuming that Figure 1 has full participation – double lines – between the EMPLOYEE superclass and the subclasses):

PHYSICIAN_ASSISTANT

ssn	specialization	fname	lname	address	phone

NURSE

ssn	dept.	fname	lname	address	phone

TECHNICIAN

ssn	work	fname	lname	address	phone

Rule 3:

Rule 3 works well if: (a) there is a disjoint relationship between the superclass and subclasses. It will create redundancy in the database if used with overlap scenarios. And, (b) if the subclasses are predicate defined (condition defined) or attribute defined. A predicate defined subclass is where a condition is placed on the value of some attribute of the superclass to determine the subclass. Figure 3 shows an example of a predicate defined subclass. If, as shown in figure 3, the EMPLOYEE entity has an additional attribute, JobType, and a condition is specified on the condition of membership in the PHYSICIAN ASSISTANT subclass by the condition (jobType="Physician_Assistant"), this is a defining predicate of the subclass, PHYSICIAN ASSISTANT. A predicate-defined subclass is shown by writing the predicate condition next to the arc that connects the subclass to the circle. Also, the defining attribute name is placed on the arc from the superclass to the circle. This rule is not recommended when subclasses have too many attributes (since this will cause too many null values).

For rule 3: Create a single relation that includes the superclass and its attributes as well as the

EMPLOYEE

ssn	fname	lname	address	phone	jobtype	specialization	department	work

subclasses and it's attributes. For this rule, we will map Figure 3 as shown below:

Rule 4:

Rule 4 works for both overlaps and disjoints, but it works better for overlaps. With disjoints, this rule will create null values when the entity is not a member of a particular subclass. This rule is also not recommended when subclasses have too many attributes (since this will also cause too many null values). If subclasses have few attributes, however, this rule may be preferred to rule 1 or rule 2 since it will eliminate the need for a relational join. In this rule, a flag is created for each superclass tuple that belongs to a subclass.

For rule 4: Create a single relation that includes the attributes of the superclass and the attributes of its subclasses. To illustrate this rule we will map Figure 4 to the COMPUTER relation, as shown at the bottom of this page.

CATEGORIES

The concept of categories extends the concepts of generalization entity types and specialization entity types even further. Categories are created by grouping entity types (generalization entity types or specialization entity types) by the roles they may play within relationships. So, categories can represent superclasses (generalization categories) or subclasses (specialization categories). Important contributions his this area have been made by Elmasri, et. al. (1985), Gogolla, et. al. (1991), Elmasri and Navathe (2007).

In Figure 5, the PAYOR could be inheriting from the PRIMARY INSURANCE, PATIENT, or OTHER RESPONSIBLE PARTY. The PRIMARY INSURANCE, PATIENT, and OTHER RESPONSIBLE PARTY represent a superclass (generalization category) of entity types. Each of the entity types in this generalization or superclass is a different entity type with different keys, but they play a common role – the role of a PAYOR. Here we would refer to the subclass (in this case, PAYOR) as a ***category*** or ***union type*** of two or more superclasses. Hence, a category is a subclass of a union of two or more superclasses that are *different* entity types (Elmasri et al., 1985; Elmasri & Navathe, 2007) playing a common role. A category is diagrammatically shown by a "$\cup$" in the circle that attaches the category to the superclasses, as shown in Figure 5. We can read Figure 5 as:

A PAYOR may be a PRIMARY INSURANCE or PATIENT or OTHER RESPONSIBLE PARTY.

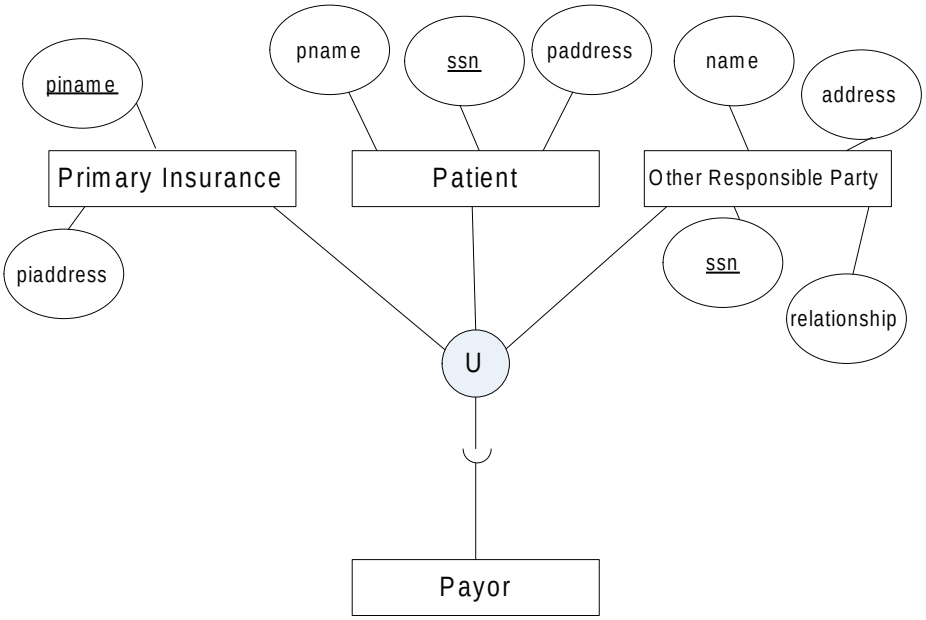
Participation Constraints in Categories

Just like other subclasses, categories can also have total (or full) participation or partial participation. Total participation means that the category holds the union of *all* entities in its superclasses. That is,

COMPUTER

item_no	make	item_model_no	lflag	fac_name	weight	fac_ssn	dflag	building	room_no

Figure 5. A Category



if there were full participation in figure 5 (double lines running from the category, PAYOR, to the circle with the "∪"), then every entity of PRIMARY INSURANCE would exist in PAYOR, and every entity of PATIENT would exist in PAYOR, and every entity of OTHER RESPONSIBLE PARTY would exist in PAYOR.

Partial participation means that a category holds only a subset of the union of all entities in its superclasses. That is, as shown in Figure 5, every entity of PRIMARY INSURANCE does not exist in PAYOR, and every entity of PATIENT does not exist in PAYOR, and every entity of OTHER RESPONSIBLE PARTY does not exist in PAYOR. Once again, partial participation would be shown by single lines from the category to the circle with the "∪".

Mapping Categories

There are two rules to map categories: (Elmasri & Navathe, 2007):

Rule 5:

Rule 5 should be used when the superclasses have different keys. For rule 5:

(i) Create a new relation to correspond to the category.
(ii) Since the keys of each of the superclasses are different, we cannot use any one of them as the key for this new relation. So, we have to specify a new key attribute (called a surrogate key) to correspond to the category in the new relation.
(iii) To be able to join the subclass with the superclass/superclasses, include the surrogate key attribute as the foreign key in each superclass relation.

To illustrate rule 5, we will map Figure 5 as shown below:

PRIMARY_INSURANCE

piname	piaddress	payorid

PATIENT

ssn	paddress	pname	payorid

OTHER_RESPONSIBLE_PARTY

ssn	relationship	name	address	payorid

PAYOR

payorid

Rule 6:

The second rule used to map categories is used when the superclass entities have the same key.

For example, we would map figure 6 as:

DORM

dormnu	dname

ON_CAMPUS

dormnu	bldg	supervisor

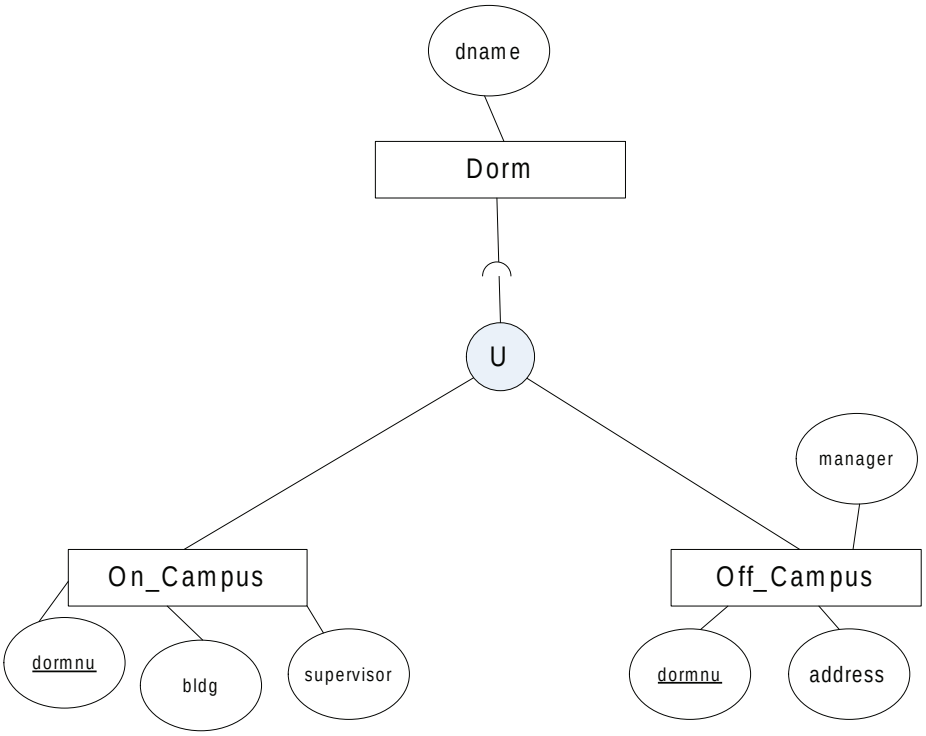
OFF_CAMPUS

dormnu	address	manager

Multiple Inheritance

In this paper we have given examples of how subclasses inherit from superclasses. In reality however, subclasses often inherit from more than one superclass. This concept is known as multiple inheritance. Categories are also not necessarily disjoint, so a given entity may be a member of several different categories. Due to the brief nature of this paper, we will not elaborate on the concept of multiple inheritance.

Figure 6. Category where superclasses have the same key



FUTURE TRENDS

Due to the intuitive nature of the ER and EER approach, and in the light of the next generation of Web applications (Shanmugasundaram, et al., 2000) which will depend on mature relational database technology with respect to storage, retrieval and update (Ceri, et al., 2000; Kappel, et al., 2001a; Kappel, et al., 2001b; Widom, 1999), the ER and EER approach will be heavily used in the future. Also, given the fact that most of the Web data have a hierarchical structure with classes and subclasses, the EER model can lend itself somewhat more naturally to a conceptual structure of web data. Hence we should see a rise the use of the EER model in the future.

CONCLUSION

Databases are becoming increasingly complex today. To deal with these complexities, it is becoming essential for database designers have to an understanding of the extended ER model, which incorporates generalizations/specializations and categories. In this paper we briefly presented generalizations/specializations and categories with the use of examples. We also presented rules to map generalizations/specializations and categories to a relational database.

REFERENCES

Bagui, S., & Earp, R. (2003). *Database Design Using Entity-Relationship Diagrams.* Auerbach Publications, Boca Raton, Florida: CRC Press.

Ceri, S., Fraternali, P., & Paraboschi, S. (2000). XML: Current Developments and Future Challenges for the Database Community. *Proceedings of the 7th International Conference on Extending Database Technology (EDBT),* Springer, LNCS 1777, Konstanz.

Chen, P. P. (1976). The entity-relationship model: Toward a unified view of data. *ACM Transactions of Database Systems, 1*(1), 9-36.

Chen, P., Thalheim, B., & Wong, L. Y. (1999). Future Directions of Conceptual Modeling. *LNCS*, 1565, 287.

Dey, D., Storey, V., & Barron, T., (1999). Improving Database Design through the Analysis of Relationships. *ACM Transactions on Database Systems, 24*(4), 453-486.

Dullea, J., Song, Li-Y., & Lamprou, I. (2003). An analysis of structural validity in entity-relationship modeling. *Data and Knowledge Engineering, 47*(2), 167-205.

Elmasri, R., & Navathe, S. B. (2007). *Fundamentals of Database Systems,* 5th ed., Addison Wesley.

Elmasri, R., Weeldreyer J., & Hevner, A. (1985). The category concept: An extension to the Entity-Relationship model. *Data and Knowledge Engineering, 1,* 75-116.

Engels, G., Gogolla, M., Hohenstein, U., Hulsmann, K., Lohr-Richter, P., Saake, G., & Ehrich, H-D. (1992/93). Conceptual Modeling of database applications using an extended ER model. *Data and Knowledge Engineering 9,* 157-204.

Gogolla, M., & Hohenstein, U. (1991). Towards a Semantic View of an Extended Entity Relationship Model. *ACM Transactions on Database Systems, 16*(3), 369-416.

Gogolla, M., Meyer, B., & Westerman, G. D. (1991). Drafting Extended EntityRelationship Schemas with QUEER. In T. J. Teorey, (Ed.), *Proc. 10th Int. Conf. on Entity-Relationship Approach,* (pp. 561-585).

Kappel, G., Kapsammer, E., & Retschitzegger, W. (2001a). Architectural Issues for Integrating XML and Relational Database Systems – The X-Ray Approach. *Proceedings of the Workshop*

on XML Technologies and Software Engineering, Toronto.

Kappel, G., Kapsammer, E., & Retschitzegger, W. (2001b). XML and Relational Database Systems – A Comparison of Concepts'. *Proceedings of the 2nd International Conference on Internet Computing (IC).* CSREA Press, Las Vegas, USA.

Markowitz, V., & Shoshani, A. (1992). Representing Extended Entity-Relationship Structures in Relational Databases: A Modular Approach. *ACM Transactions on Database Systems, 17*(3), 423-464.

Necasky, M. (2006). Conceptual Modeling for XML: A Survey. *Proceedings of the DATESO 2006 Annual International Workshop on Databases, Texts, Specifications, and Objects,* Desna-Cerna Ricka, Czech Republic.

Sanders, L. G. (1995). *Data Modeling.* International Thomson Publishing Company.

Shanmugasundaram, J., Shekita, E., Barr, R., Carey, M., Lindsay, B., Pirahesh, H., & Reinwald, B. (2001). Efficiently publishing relational data as XML document. *VLDB Journal 19*(2-3), 133-154.

Widom, J. (1999). Data Management for XML – Research Directions. *IEEE Data Engineering Bulletin*. *Special Issue on XML*, *22*(3), 44-52.

KEY TERMS

Category: A collection of objects or entities that is a subset of the union of different entity types; entities offering a similar role are grouped into a category.

Entity sets or entity types: Similar entities or objects with common properties are summarized into entity sets or entity types; graphically represented in rectangles.

Generalization: When entities have similar basic attributes, they can be classified into a generalized entity type. Generalization is the process of finding commonalities between entities to be able to abstract to a higher level entity set.

Inheritance: A subclass inherits all the attributes of a superclass and all the relationships that it (the superclass) participates in.

Specialization: A process of conceptual refinement to form specialized subclasses for entity sets. An entity type can have several specializations or categories of specializations defined on the basis of some distinguishing characteristics of the entities in the superclass.

Subclass: Same as specialization; a meaningful sub-grouping of entity-sets that needs to be represented explicitely. These sub-groups are a subset of the entities that belong to the entity set from which they are derived.

Superclass: Same a generalization. A superclass is the main set (entity) from which subsets (sub-classes) are defined based on meaningful criteria needed for a particular database.

Chapter II
Bounded Cardinality and Symmetric Relationships

Norman Pendegraft
University of Idaho, USA

INTRODUCTION

Bounded cardinality occurs when the cardinality of a relationship is within a specified range. Bounded cardinality is closely linked to symmetric relationships. This article describes these two notions, notes some of the problems they present, and discusses their implementation in a relational database.

BOUNDED CARDINALITY AND SYMMETRIC RELATIONSHIPS

Bounded Cardinality

An entity relationship diagram (ERD) shows the cardinality of each entity in a relationship. In an ERD, minimum cardinalities can be either 0 or 1, and maximum cardinalities can be 1 or infinity. Bounded cardinality occurs when a relationship between entities has cardinality within a specified range. Problems displaying bounded cardinality might include team rosters that must have exactly 5, 9, 11, or some other number of players. Figure 1 illustrates how UML (unified modeling language) provides for modeling specified-range relationships in a class diagram (Dennis, Wixom, & Tegarden, 2005). ERD, as described by Chen (1976), does not, although there are extensions to the ERD model that do (Webre, 1981). The SQL-92 standard provides for such constraints, but many relational database management systems (RDBMSs) do not support these features, and consequently do not allow for easy implementation of such a constraint (Lewis, Bernstein, & Kifer, 2002).

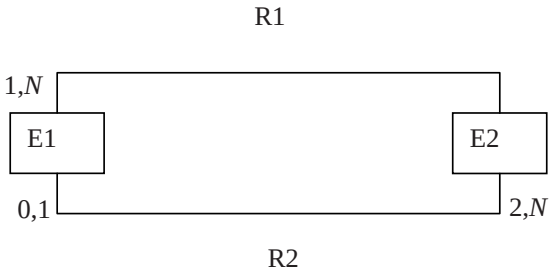
Bounded cardinality presents some interesting problems. For example, Boufares and Kraïem (2001) point out that cardinality constraints may result in conflicts. Figure 2 illustrates one of their examples. In Figure 2, if we let e_i be the number of instances of entity E_i and r_i be the number of instances of relationship R_i, then we get the following constraints.

$$r_1 = e_1 \qquad r_1 > e_2$$
$$r_2 < e_2 \qquad r_2 > 2\, e_1$$

These lead in turn to e1 > e2 and e2 > 2 e1. Clearly these allow only the solution e1 = e2 =0,

Figure 1. Specified range / bounded cardinality

Figure 2. Inconsistent cardinality constraints



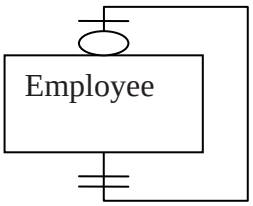
that is, an empty database. Boufares and Bennaceur (2004) offer a mathematical programming technique to detect inconsistent constraints.

Symmetric Relationships

Symmetric relationships require that if a relationship R(x,y) holds for x and y in S, then R(y,x) must also hold (Dean, 1966). For example, if George is married to Martha, then Martha must also be married to George. While these relations are common, they may be difficult to model and to implement in an RDBMS. In particular, there is no way in an ERD to easily show a symmetric relationship. The relational database model does not provide for any way to impose a constraint requiring that a relationship be symmetric.

Bounded cardinality also arises from symmetric relationships. As Ross and Stoyanovich (2004, p. 913) note, there is a "natural isomorphism between symmetric relationships among k entities and k-element multi-sets." Multisets may be represented using bounded cardinality. In other words, a group may also be viewed as a relationship amongst its members. For example, a marriage may be viewed as a group of size 2. SQL-2003 provides for a

Figure 3. A simple design for a marriage



multiset data type (Eisenberg, Melton, Kulkarni, & Zemke, 2004). Similarly, a team roster represents a symmetric relationship. Suppose an athletic league maintains a roster of players. Each player is on one team, and each team must have from 5 to 10 players. In Figure 1 we would have n = 5 and N = 10. Note that the business problem requires that the relationship be symmetric: If Able is on a team with Baker and Charlie, then Baker must be on a team with Able and Charlie as well.

Constraints

Business rules are generally imposed in a database design via constraints. These constraints may be implemented declaratively or procedurally. Declarative constraints are created in the database definition. An example is "NOT NULL" to require that an attribute have some value. Another is "REFERENCES," which imposes a referential integrity constraint on a tuple.

Procedural constraints are imposed in the logic of applications or in triggers. For example, the procedure that allows us to record a marriage could be designed so as to ensure that the data entered would be symmetric. In contrast, a declarative constraint would build that requirement into the database itself and simply forbid entering nonsymmetric data. Date (2000) points out that declarative constraints are superior because they relieve programmers of additional programming tasks. Since the code would be more difficult to write, test, and debug, there is a greater chance of an error that will further introduce errors into the data. Türker and Gertz (2001) point out that declarative constraints are less costly to execute.

Example

As a simple example of a symmetric relationship, consider a personnel database that records marriages, but only those between employees (see Figure 3; Teorey, Yang, & Fry, 1986). Note that the ERD cannot indicate that this is a symmetric relationship.

SQL-92 Solution

SQL-92 provides for this problem using a semantic constraint that may be implemented as a subselection in a check constraint or as an assertion (Lewis et al., 2002). Consider the table Person (personid [PK], groupid, other_data). The person table could include a constraint like the following.

```
CHECK (NOT EXISTS (
SELECT * FROM person WHERE groupid IS NOT NULL
GROUP BY groupid
HAVING COUNT <>2))[1]
```

This restricts a person to at most one group and a group to at most two persons. This constraint could also be implemented as an assertion. Unfortunately, these features may not be widely available. According to Türker and Gertz (2001), it was not available in any of the leading products at the time of their paper.

Approaches in a Noncompliant Environment

Now consider the problem faced by a designer using an RDBMS that does not support these features. In an environment that does not support these features, other imperfect designs may be attempted. Note that a marriage is a two-element multiset. Note also that the obvious solution of treating male and female employees differently cannot be generalized to all groups, nor will it work in all jurisdictions for marriages.

One solution is to record all of the data in each record. This consumes extra storage space and requires that updating logic impose the symmetry requirement. Table 1 illustrates this design. Note that it does not impose symmetry because it would permit George to be married to Martha, and Martha to be married to John, and John to be married to Abigail, and so forth. We could not correctly update the table by inserting (Abe, Mary) unless we also inserted (Mary, Abe). Likewise, updates and deletions must apply to two records.

Table 1.

Ename	SpouseEname	Data
John	Abigail	John's data
Abigail	John	Abigail's data
Martha	George	Martha's data
George	Martha	George's data

An alternative illustrated in Table 2 is to store the data in only one of the records. This reduces storage space and update processing time. Note that is does not prevent the sort of "chain" marriage noted above, so it still requires external logic to enforce symmetry.

Table 2.

Ename	SpouseEname	Data
Abigail	John	Abigail's data
George	Martha	George's data
Martha	<null>	Martha's data
John	<null>	John's data

Table 3.

Ename	Data
Abigail	Abigail's data
George	George's data
Martha	Martha's data
John	John's data

Table 4.

Marriage ID	Ename
Marriage 1	Abigail
Marriage 1	John
Marriage 2	George
Marriage 2	Martha

It also complicates searches since two columns must be searched to answer questions like "Who is John's spouse?" The SQL query

```
SELECT * FROM table2 WHERE ename = 'John' ;
```

will not find (Abigail, John) even though that marriage satisfies the intent of the query. Instead, we must search over the SpouseEname column as well as Ename.

The design could be modeled by creating a second table (Tables 3 and 4). Table 4 carries the information regarding marriages. Now we must ensure that updates on Table 4 are aware of the symmetry. We also incur the cost of a join when we want to find all of the data about a married couple. This design enforces the symmetry by creating a group of size 2, that is, a bounded cardinality problem. Note that it has $k = 2$ 1:1 relationships between the group entity and the individual entity. While this design imposes the constraint, it also imposes a computational cost because the data in the original table can only be recovered with a join. Other designs are possible, but they suffer from similar problems.

Array Data Types

Some object relational database management systems offer an array data type that permits bounded arrays to be stored in a singe column. However, this solution creates normalization problems and may also be difficult to search. The Oracle varray (*Oracle9i Application Developer's Guide*, n.d.; *PL/SQL User's Guide and Reference*, n.d.) is a data type that permits an element of a table to be an array. Varray does impose an upper bound on the array, but elements of the array may not be individually updated nor can they be individually indexed.

FUTURE TRENDS

Calero, Ruiz, Baroni, Brito e Abreu, and Piattini (2006) discuss the evolution and ontology of the SQL standard (International Organization for Standardization [ISO], 2003). Their discussion leads one to hope to see full implementation of the complete SQL features in commercial products. Ross and Stoyanovich (2004) suggested that SQL be extended to include a symmetric declarative constraint. They describe data structures and methods that would allow for the updating, querying, and indexing of such structures. Given the importance of such problems, it seems reasonable to hope that such extensions to SQL will be forthcoming.

CONCLUSION

It appears that there are many theoretical advances that are not readily available to practitioners. Bounded cardinality has been widely studied, yet is unmentioned in many popular database textbooks. Perhaps this is so because the semantic constraints are not available in some of the most popular database products. In any case, it is hoped that this work will contribute to greater awareness of these trends.

ACKNOWLEDGMENT

The author thanks two anonymous referees for their helpful comments.

REFERENCES

Boufares, F., & Bennaceur, H. (2004). Consistency problems in ER-schemas for database systems. *Information Sciences, 163*, 263-274.

Boufares, F., & Kraïem, N. (2001). A new tool to analyze ER-schemas. *Proceedings of the Second Asia-Pacific Conference on Quality Software* (pp. 302-307).

Calero, C., Ruiz, F., Baroni, A., Brito e Abreu, F., & Piattini, M. (2006). An ontological approach to describe the SQL:2003 object-relational features. *Computer Standards & Interfaces, 28*(6), 695-713

Chen, P. P.-S. (1976). The entity-relationship model: Toward a unified view of data. *ACM Transaction on Database Systems, 1*(1), 9-36.

Date, C. J. (2000). *An introduction to database systems* (7th ed.). Reading, MA: Addison Wesley.

Dean, R. (1966). *Elements of abstract algebra.* New York: Wiley.

Dennis, A., Wixom, B. H., & Tegarden, D. (2005). *Systems analysis and design with UML version 2.0* (2nd ed.). Wiley.

Eisenberg, A., Melton, J., Kulkarni, K., & Zemke, F. (2004). SQL:2003 has been published. *ACM SIGMOD Record, 33*(1), 119-126.

International Organization for Standardization (ISO). (2003). *Database language SQL, ISO/IEC 9075.*

Lewis, P. M., Bernstein, A., & Kifer, M. (2002). *Database transaction processing: An application oriented approach.* Boston: Addison Wesley.

Oracle9i application developer's guide: Object-relational features release 2 (9.2). (n.d.). Retrieved from http://download-east.oracle.com/docs/cd/B10501_01/appdev.920/a96594/adobjdes.htm#441636

PL/SQL user's guide and reference 10g release 1 (10.1). (n.d.). Retrieved from http://download-east.oracle.com/docs/cd/B14117_01/appdev.101/b10807/05_colls.htm#i20446

Ross, K. A., & Stoyanovich, J. (2004). *Symmetric relations and cardinality-bounded multisets in database systems.* Proceedings of the 20th International Conference on Very Large Databases, Toronto, Ontario, Canada.

Teorey, T. J., Yang, D., & Fry, J. P. (1986). A logical design methodology for relational databases using the extend entity-relationship model. *Computing Surveys, 18*(2).

Türker, C., & Gertz, M. (2001). Semantic integrity support in SQL:1999 and commercial (object-) relational database management systems. *The VLDB Journal, 10*, 241-269.

Webre, N. W. (1981). An extended entity-relationship model. *Proceedings of the Second International Conference on the Entity-Relationship Approach to Information Modeling and Analysis.*

KEY TERMS

Bounded Cardinality: A specific finite upper and/or lower bound on the cardinality of a relationship between entities.

Cardinality: The number of instances of an entity associated with a relationship.

Declarative Constraint Support: The ability of a database engine to implement constraints as part of the definition of the schema.

Multiset: A collection in which elements may occur more than once.

Procedural Constraint Support: Implementation of a constraint on a database via a procedure.

Reflexive Relation: If R is a relation on A, then a in A implies R(a,a). Sometimes it is erroneously used as a synonym for a symmetric relation.

Relation: For sets A and B, a relation R is a subset of the Cartesian product AXB, that is, $R \subseteq \{(a, b)/a \in A, b \in B\}$. The relation may be written R(a,b) indicating that (a,b) is in R.

Relationship: An association between entities.

Semantic Constraint: In contrast to a structural constraint, a constraint that does not place a limit on the database structure.

Specified Range: UML term used to describe bounded cardinality in class diagrams.

Symmetric Relation: If R is a relation on AXA, then R is symmetric if and only if R(a,b) implies R(b,a).

ENDNOTE

[1] The author is indebted to an anonymous referee for this solution.

Chapter III
A Paraconsistent Relational Data Model

Navin Viswanath
Georgia State University, USA

Rajshekhar Sunderraman
Georgia State University, USA

INTRODUCTION

Typically, relational databases operate under the Closed World Assumption (CWA) of Reiter (Reiter, 1987). The CWA is a meta-rule that says that given a knowledge base KB and a sentence P, if P is not a logical consequence of KB, assume ~P (the negation of P).

Thus, we explicitly represent only positive facts in a knowledge base. A negative fact is implicit if its positive counterpart is not present. Under the CWA we presume that our knowledge about the world is complete i.e. there are no "gaps" in our knowledge of the real world. The Open World Assumption (OWA) is the opposite point of view. Here, we "admit" that our knowledge of the real world is incomplete. Thus we store everything we know about the world – positive and negative. Consider a database which simply contains the information "Tweety is a bird". Assume that we want to ask this database the query "Does Tweety fly?". Under the CWA, since we assume that there are no gaps in our knowledge, every query returns a yes/no answer. In this case we get the answer "no" because there is no information in the database stating that Tweety can fly. However, under the OWA, the answer to the query is "unknown". i.e. the database does not know whether Tweety flies or not. We would obtain a "no" answer to this query under the OWA only if it was explicitly stated in the database that Tweety does not fly.

Current implementations of relational databases adopt the CWA; and for good reason. The negative facts generally turn out to be much larger than the positive facts and it may be unfeasible to store all of it in the database. A typical example is an airline database that records the flights between cities. If there is no entry in the database of a flight between city X and city Y, then it is reasonable to

conclude that there is no flight between the cities. Thus for many application domains the Closed World Assumption is appropriate. However, there are a number of domains where the CWA is not appropriate. A prime example is databases that require domain knowledge. For example, consider a biological database that stores pairs of neurons that are connected to each other. If we were to ask this database the query "Is neuron N1 connected to neuron N2?" and this information was not available in the database, is "no" an appropriate answer? What if we do not know yet whether N1 is connected to N2? Then surely the answer "no" is misleading. A more appropriate answer would be "unknown" which we would obtain under the OWA.

Inconsistent information may be present in a database in various forms. The most common form of inconsistency in relational databases is due to the violation of integrity constraints imposed on the database. Under the OWA, inconsistency may also be introduced directly by having both a fact and its negation stored explicitly in the database. Such a situation may arise when data is integrated from multiples sources.

The aim of this article is to introduce a data model that allows the user to store both positive and negative information. When the user poses a query to the database under this model, he obtains both positive and negative answers. The positive answers are those for which the answer to the query is "yes" and the negative answers are those for which the answer to the query is "no". We define the data model and a relational algebra for query processing.

Since the model allows the user to store both positive and negative information explicitly, it is possible for the database to become inconsistent. The data model we introduce allows the user to deal with inconsistent information by keeping the inconsistencies local so that whenever a query is posed, we obtain an inconsistent answer only when the database is itself inconsistent. However, the consistent portion of the database remains unaffected (Grant J. and Subrahmanian V.S., 2000).

BACKGROUND

Two important features of the relational data model (Codd, 1970) for databases is its value-oriented nature and rich set of simple, but powerful algebraic operators. Moreover, a strong theoretical foundation for the model is provided by the classical first order logic. A limitation of the relational model is that it cannot be applied in non-classical situations. Null values in relational databases have been studied extensively (Maier, 1983). Incomplete information has been studied in (Imielinski T. & Lipski Jr. W., 1984, Sarma et al 2006, Antova et al 2007, Benjelloun et al 2008). Disjunctive information has been studied in (Liu K.-C & Sunderraman R., 1990; Liu K.-C & Sunderraman R., 1991, Edward Chan P.F, 1993, Vadaparty K.V and Naqvi S.A 1995). The model that we present here, the paraconsistent relational data model, is a generalization of the relational data model (Bagai R. and Sunderraman R., 1995). This model is capable of manipulating inconsistent as well as incomplete information. Paraconsistent relations are the fundamental data structures underlying the model. A paraconsistent relation essentially consists of two kinds of tuples: ones that definitely belong to the relation and others that definitely do not belong to the relation. These structures are strictly more general than ordinary relations, in that for any ordinary relation there is a corresponding paraconsistent relation with the same information content, but not vice versa.

PARACONSISTENT RELATIONS

In this section, we construct a set-theoretic formulation of paraconsistent relations. Unlike ordinary relations that can model worlds in which every tuple is known to hold an underlying predicate or not to hold it, paraconsistent relations provide a framework for incomplete or inconsistent information about tuples. The data model is *skeptical* in that the tuples that are included as answers to

queries are those that are the beliefs of *any* rational reasoner. The reason that we adopt the skeptical approach is that it is tractable as compared to the credulous approach where the number of possible answers could be exponential in the size of the database.

Let a relation scheme Σ be a finite set of attribute names, where for any attribute name $A \in \Sigma$, $dom(A)$ is a non-empty domain of values for A. A tuple on Σ is any map $t : \Sigma \rightarrow \bigcup_{A \in \Sigma} dom(A)$, such that $t(A) \in dom(A)$, for each $A \in \Sigma$. Let $\tau(\Sigma)$ denote the set of all tuples on Σ.

Definition 1: An ordinary relation on scheme Σ is any subset of $\tau(\Sigma)$.

Definition 2: A *paraconsistent relation* on scheme Σ is a pair $R = \langle R^+, R^- \rangle$, where R^+ and R^- are any subsets of $\tau(\Sigma)$.

Intuitively, R^+ is the set of tuples *known to be in* R and R^- is the set of tuples *known not to be in* R. Notice that the sets R^+ and R^- may not be disjoint in which case they represent inconsistent information and when R^+ and R^- together do not cover all the tuples in $\tau(\Sigma)$ they represent incomplete information.

Definition 3: A paraconsistent relation R on scheme Σ is called a *consistent* relation when $R^+ \cap R^- = \phi$. R is called a *complete* relation when $R^+ \cup R^- = \tau(\Sigma)$. If R is both consistent and complete, i.e., $R^- = \tau(\Sigma) - R^+$, then it is a *total* relation.

ALGEBRAIC OPERATORS ON PARACONSISTENT RELATIONS

In this section, we define the relational algebra on paraconsistent relations. The operators are generalizations of the usual operators on ordinary relations. To reflect generalization, we place a dot over the symbols of the operators. For example, σ denotes selection on ordinary relations and $\dot{\sigma}$ denotes selection on paraconsistent relations.

Set-Theoretic Operators

Definition 4: Let R and S be paraconsistent relations on scheme Σ. Then, the union of R and S, denoted by $R \dot{\cup} S$, is a paraconsistent relation on scheme Σ given by

$$(R \dot{\cup} S)^+ = R^+ \cup S^+, (R \dot{\cup} S)^- = R^- \cap S^-$$

The union operator may be understood in terms of Boolean laws. The positive component of the union is the set of tuples appearing in either R or S, which is simply $R^+ \cup S^+$. The negative component of the union is the logical complement of this set, $(R^+ \cup S^+)' = (R^+)' \cap (S^+)' = R^- \cap S^-$.

Definition 5: Let R be a paraconsistent relations on scheme Σ. Then, the complement of R, denoted by $\dot{-} R$, is a paraconsistent relation on scheme Σ given by

$$(\dot{-} R)^+ = R^-, (\dot{-} R)^- = R^+$$

The complement of a paraconsisent relation is obtained by simply flipping the positive and negative components.

Definition 6: Let R and S be paraconsistent relations on scheme Σ. Then, the intersection of R and S, denoted by $R \dot{\cap} S$, is a paraconsistent relation on scheme Σ given by

$$(R \dot{\cap} S)^+ = R^+ \cap S^+, (R \dot{\cap} S)^- = R^- \cup S^-$$

The positive component of the intersection is simply the intersection of the positive components and the negative component of the intersection can also be explained in terms of Boolean laws:

$$(R^+ \cap S^+)' = (R^+)' \cup (S^+)' = R^- \cup S^-.$$

Definition 7: Let *R* and *S* be paraconsistent relations on scheme Σ. Then, the difference of *R* and *S*, denoted by $R \dot{-} S$, is a paraconsistent relation on scheme Σ given by,

$$(R \dot{-} S)^+ = R^+ \cap S^-, (R \dot{-} S)^- = R^- \cup S^+$$

The tuples in the positive component of $R \dot{-} S$ are those that are **in** R **and not in** S, which is exactly the set $R^+ \cap S^-$. The negative component can again be obtained by applying boolean laws: $(R^+ \cap S^-)' = (R^+)' \cup (S^-)' = R^- \cup S^+$.

Relation-Theoretic Operators

If Σ and Δ are schemes such that $\Sigma \subseteq \Delta$, then for any tuple $t \in \tau(\Sigma)$, we let t^Δ denote the set $\{t' \in \tau(\Delta) \mid t'(A) = t(A), \forall A \in \Sigma\}$ of all extensions of *t*. We now define the relation-theoretic operators on paraconsistent relations.

Definition 8: Let *R* and *S* be paraconsistent relations on schemes Σ and Δ respectively. Then the natural join of *R* and *S*, denoted by $R \dot{\infty} S$, is a paraconsistent relation on scheme $\Sigma \cup \Delta$, given by

$$(R \dot{\infty} S)^+ = R^+ \infty S^+, (R \dot{\infty} S)^- = (R^-)^{\Sigma \cup \Delta} \cup (S^-)^{\Sigma \cup \Delta}$$

The positive component of the join is simply the join of the positive components. For the negative component, any tuple that is false in *R* or *S* cannot participate in the join. Hence all extensions of tuples in R^- and S^- are in the negative component of the join.

Definition 9: Let *R* be a paraconsistent relation on scheme Σ and let $\Delta \subseteq \Sigma$. Then, the projection of *R* onto Δ, denoted by $\dot{\pi}_\Delta(R)$, is a paraconsistent relation on scheme Δ given by

$$\dot{\pi}_\Delta(R)^+ = \pi_\Delta(R^+),$$

$$\dot{\pi}_\Delta(R)^- = \{t \in \tau(\Delta) \mid t^\Sigma \subseteq R^-\}$$

The positive component of the projection is simply the projection of the positive component. The negative component of the projection is the set of tuples on scheme Δ all of whose extensions are in R^-. The projection operator is similar to the existential quantifier in first order logic. Consider for instance, the *teaches(instructor,class)* relation, which stores the names of instructors and the classes they teach. The projection of *teaches* onto *instructor* would be the names of all instructors who teach a class. This is the first order formula $(\exists c)$(teaches(i,c)). The negative component of the projection is the negation of this formula: $\sim(\exists c)$(teaches(i,c)) = $(\forall c)\sim$teaches(i,c). i.e. the set of instructors for whom for every class in the database, it is false that they teach the class. This is simply the set of instructors *i* such that for every class *c* in the database the tuple <i,*c*> is in the negative component of the *teaches* paraconsistent relation. In general, for any scheme $\Delta \subseteq \Sigma$, the negative component is simply the tuples on scheme Δ all of whose extensions are in R^-.

Definition 10: Let *R* be a paraconsistent relation on scheme Σ, and let F be any formula involving the attribute names in Σ, constant symbols (denoting value sin the attribute domains), equality symbol =, the negation symbol ~, and the connectives $\vee$ and $\wedge$. Then, the selection of *R* by *F*, denoted by $\dot{\sigma}_F(R)$, is a paraconsistent relation on scheme Σ, given by,

$$\dot{\sigma}_F(R)^+ = \sigma_F(R^+), \dot{\sigma}_F(R)^- = R^- \cup \sigma_{\sim F}(\tau(\Sigma))$$

The positive component of the selection is the set of tuples such that they are in *R* and satisfy the condition *F*. The negative component of the selection is the set of tuples that are either not in *R* **or** do not satisfy the condition *F*. This is exactly the set denoted by $R^- \cup \sigma_{\sim F}(\tau(\Sigma))$.

The correctness of these operators can be established. Please refer to (Bagai R. and Sunderraman R., 1995) for proofs of correctness.

QUERY EXAMPLE

In this section, we demonstrate the relational operators on an example supplier-parts database. The database consists of two entities: *suppliers(sno,sname,city)* and *parts(pno,pname,color)* and the relationship *supplies(sno,pno)* between the two entities.

The tuples shaded in gray indicate the negative components. Also, we use the set-valued notation as shorthand for sets of tuples. For instance, the set-valued tuple <1,{Tom,Jack},Atlanta> is shorthand for the set {<1,Tom,Atlanta>,<1,Jack,Atlanta>}.

Suppliers

Sname	City
John	Atlanta
Tom	Atlanta
Jack	Atlanta
{Tom,Jack}	Atlanta
{John,Jack}	Atlanta
{John,Tom}	Atlanta

Parts

Pname	Color
Nut	Blue
Bolt	Red
Axle	Green
Nut	{Red,Green}
Bolt	Green
Axle	{Red,Blue}

Supplies

Sname	Pname
John	Bolt
Tom	Nut
Jack	Axle
Jack	Nut
Jack	Bolt

Consider the following query to this database:

Find the names of the suppliers who supply neither red nor blue parts.

The query may be expressed in relational algebra as follows:

$$\div\left(\dot{\pi}_{<sname>}\left(\left(supplies \dot{\infty} \dot{\sigma}_{color='red' \vee color='blue'}\left(parts\right)\right)\right)\right)$$

Shown below is the answer to the query. The answer is split into four parts – selection, join, projection and complement for clarity.

$$\dot{\sigma}_{color='red' \vee color='blue'}\left(parts\right)$$

Pname	Color
Nut	Blue
Bolt	Red
Nut	{Red,Green}
Bolt	Green
Axle	{Red,Blue,Green}

$$supplies \dot{\infty} \dot{\sigma}_{color='red' \vee color='blue'}\left(parts\right)$$

Sname	Pname	Color
John	Bolt	Red
Tom	Nut	Blue
Jack	Nut	Blue
Jack	Bolt	{Red,Blue}
{Jack,John,Tom}	Nut	{Red,Green}
{Jack,John,Tom}	Bolt	Green
{Jack,John,Tom}	Axle	{Red,Blue,Green}

$$\dot{\pi}_{<sname>}\left(supplies \dot{\infty} \dot{\sigma}_{color='red' \vee color='blue'}\left(parts\right)\right)$$

Sname
Jack
John
Tom

$$\dot{-}\left(\dot{\pi}_{<sname>}\left(\left(supplies \dot{\infty} \dot{\sigma}_{color='red' \vee color='blue'}\left(parts\right)\right)\right)\right)$$

Sname
John
Tom
Jack

APPLICATION

Deductive databases are a generalization of relational databases where in addition to manipulating explicitly represented facts, deductive databases provide ways to infer new facts through rules. The semantics of deductive databases without negation is defined by Apt and van Emden in (Apt K.R. and van Emden M.H, 1982). When negation is introduced in a deductive database, it creates some problems. A subset of the deductive databases with negation, called *stratified deductive databases* have a clear semantics. A deductive database is stratified if there is no recursion through negation. However, there is some disagreement among researchers about the semantics of non-stratified deductive databases. A number of semantics have been proposed for deductive databases with negation and one of them is the one proposed by Fitting (Fitting M., 1985). Here, Fitting defines the semantics of a deductive database with negation as the least fixpoint of an operator on partial interpretations. An application of the paraconsistent relational data model is that it can be used in the bottom-up computation of the Fitting model of deductive databases. We provide here just a brief overview of the technique. For details, please refer to (Bagai R. and Sunderraman R., 1996). First, we briefly introduce deductive databases. For a more detailed exposition, please refer to (Lloyd J.W, 1987).

Let L be a given underlying language with a finite set of constants, variables and predicate symbols, but no function symbols. A *term* is either a constant or a variable. An *atom* is of the form $p(t_1 \ldots t_n)$ where *p* is a predicate symbol and the t_i's are terms. A *literal* is either a *positive literal* *A* or a *negative literal* $\neg A$ where *A* is an atom. For any literal *l* we let l' denote its complementary literal, i.e., if *l* is positive then $l' = \neg l$, otherwise $l = \neg l'$.

Definition 11: A *deductive database* is a finite set of clauses of the form

$$a \leftarrow b_1 \ldots b_m$$

where $m \geq 0$ and each b_i is an atom.

A term, atom, literal or clause is called *ground* if it contains no variables. The *Herbrand Universe* of the underlying language is the set of all ground terms; the *Herbrand Base* of the language is the set of all ground atoms; A *Herbrand interpretation* of the language is any subset of the Herbrand base. A *ground instance* of a term, atom, literal or a clause is the term, atom, literal or clause respectively, obtained by replacing each variable by a constant. For any deductive database *P* we let P^* denote the set of all ground instances of clauses in *P*.

The semantics of a deductive database is the least fixpoint of an immediate consequence operator T_P with respect to the partial order of set inclusion. The reader is referred to (Lloyd J.W, 1987) for a detailed exposition.

Definition 12: A *general deductive database* is a finite set of clauses of the form

$$a \leftarrow l_1 \ldots l_m$$

where *a* is an atom, $m \geq 0$ and each l_i is a literal.

We describe very briefly one of the semantics for general deductive databases. For a detailed explanation, the reader is referred to (Fitting M., 1985).

Definition 13: A *partial interpretation* is a pair $I = \langle I^+, I^- \rangle$, where I^+ and I^- are any subsets of the Herbrand base.

A partial interpretation I is consistent if $I^+ \cap I^- = \Phi$. For any partial interpretations I and J, we let $I \cap J$ be the partial interpretation $\langle I^+ \cap J^+, I^- \cap J^- \rangle$, and $I \cup J$ be the partial interpretation $\langle I^+ \cup J^+, I^- \cup J^- \rangle$. We also say that $I \subseteq J$ whenever $I^+ \subseteq J^+$ and $I^- \subseteq J^-$.

For any general deductive database, recall that P^* is the set of all ground instances of clauses in *P*. The weak well-founded model of *P* is the least fixpoint of the immediate consequence function T_P^F on consistent partial interpretations defined as follows:

Definition 14: Let *I* be a partial interpretation. Then, $T_P^F(I)$ is a partial interpretation given by,

$T_P^F(I)^+ = \{ a \mid$ for some clause $a \leftarrow l_1 \ldots l_m$ in P^*, for each $i, 1 \le i \le m$, if l_i is positive then $l_i \in I^+$, and if l_i is negative then $l_i \in I^- \}$

$T_P^F(I)^- = \{ a \mid$ for every clause $a \leftarrow l_1 \ldots l_m$ in P^*, there is some $i, 1 \le i \le m$, such that if l_i is positive then $l_i \in I^-$, and if l_i is negative then $l_i \in I^+ \}$

It can be shown that T_P^F preserves consistency and possesses a least fixpoint. This least fixpoint is called the weak well-founded model for *P*.

A mechanism that computes the ordinal powers of T_P^F until it reaches the least fixpoint can be employed to construct the weak well-founded model of *P*. The mechanism uses paraconsistent relations as the semantic objects associated with the predicate symbols occurring in *P*. We illustrate the method through an example. For a detailed explanation of the procedure, please refer to (Bagai R. and Sunderraman R., 1996).

Consider the following deductive database:

$r(a, c)$.
$r(b, b)$.
$s(a, a)$.
$p(X) \leftarrow r(X, Y), \neg p(Y)$.
$p(Y) \leftarrow s(Y, a)$.

We associate with each predicate in the database a paraconsistent relation with the same name. Here, predicates *r* and *s* constitute the *extensional database* and predicate *p* is the *intensional database*.

Next, we translate the set of rules for each predicate into an expression in relational algebra following (Ullman, J.D., 1988). In this particular instance, the expression for *p* is given by

$$p = \dot{\pi}_{\{X\}}\left(r(X,Y) \dot{\infty} \div p(Y)\right) \dot{\cup} \dot{\pi}_{\{Y\}}\left(\dot{\sigma}_{Z=a}\left(s(Y,Z)\right)\right)$$

We now mimic the evaluation of the T_P^F operator by evaluating the above expression in steps. i.e. evaluating the expression once corresponds to one application of T_P^F. This process continues until there is no change in the values of all the predicate symbols: this corresponds to the fixpoint of T_P^F. We trace here the values in each predicate symbol step by step:

Step 1:

s

a	a
a	b
a	c
b	a
b	b
b	c
c	a
c	b
c	c

r

a	c
b	b
a	a
a	b
b	a
b	c
c	a
c	b
c	c

The paraconsistent relation corresponding to the intensional predicate *p* is empty to begin with.

Step 2:

The paraconsistent relations corresponding to r and s stay the same and hence we do not reproduce them here. Evaluating the expression for p however, now produces new information in the paraconsistent relation corresponding to p.

By applying the operator definitions introduced earlier, $\dot{\pi}_{\{X\}}\left(r\left(X,Y\right)\dot{\infty}\dot{-}p\left(Y\right)\right)$ can be seen to be the paraconsistent relation

c

and $\dot{\pi}_{\{Y\}}\left(\dot{\sigma}_{Z=a}\left(s\left(Y,Z\right)\right)\right)$ is the paraconsistent relation

a
b
c

Their union is thus the paraconsistent relation

a
c

Further iterations do not change the value of p. Step 2 can be seen to mimic the application of the T_P^F function.

CONSTRAINTS AND STORAGE

One of the issues with the paraconsistent relational data model is the huge amount of space required in order to store it since the negative information in a database generally tends to be very large. One of the ways in which this issue may be addressed is to adopt a non-first normal form by introducing set-valued tuples. This has been investigated in (Viswanath N. and Sunderraman R., 2007). Here relations are extended to allow only sets as tuple components. The notation is extended in order to allow the complement operation from set theory. For instance, $\overline{\{a\}}$ as a tuple component denotes all elements in the domain of that attribute except a. Φ and $\overline{\Phi}$ denote the empty set and the entire domain respectively. A by-product of this is that it now becomes easy to incorporate the negative information introduced by integrity constraints on the database.

Consider for example a relation *supervisor(SSN, SuperSSN)* along with the functional dependency constraint $SSN \rightarrow SuperSSN$. Consider a tuple $\langle 111,333\rangle$ in the relation. In the set-valued paraconsistent model, this dependency may be enforced by introducing the tuple $\langle 111,\overline{333}\rangle$ in the negative component of the corresponding set-valued paraconsistent relation. In the paper, a relational algebra and an operator to remove redundancies that may be introduced using the set notation are defined. The reader is referred to the paper for a more detailed exposition.

FUTURE TRENDS

The chapter introduced a data model that includes both positive and negative information thus extending the relational model. The OWA is increasingly becoming necessary in a number of application areas, databases involving a process of discovery being a prime example. We are currently interested in including two kinds of negation in relational databases: an explicit negation (of the kind introduced here) and a default negation, which is, in a sense, a weaker form of negation. The 2-negation concept has already received wide attention in the logic programming community.

Another interesting area of future work is the handling of inconsistencies introduced by the violation of integrity constraints imposed on the database (Arenas M., Bertossi L. and Chomicki J., 1999). Here, a concept of a repair is introduced where a minimal set of updates (insertion and deletions) is made to the database so that the consistency is restored. A consistent

query answer is defined as the set of tuples that is obtained as the answer in every minimal repair of the database. The paraconsistent data model may be used as a framework for such databases since inconsistency is handled in a very natural manner and kept local to a few tuples without affecting the whole database.

CONCLUSION

We have presented a data model that is a generalization of the relational data model and can represent both indefinite and incomplete information. The paraconsistent data model is strictly more general than the relational model in the sense that for every relation there is a paraconsistent relation with the same information content but not vice versa. It is an appropriate data structure to represent both inconsistent and incomplete information. As an application, it is shown that paraconsistent relations can be used in order to construct the weak well-founded model of deductive databases.

REFERENCES

Antova, L., Koch, C., & Olteanu, D. (2007). 10^10^6 Worlds and Beyond: Efficient Representation and Processing of Incomplete Information. *Proceedings of the 23rd International Conference on Data Engineering ICDE'07,* Istanbul, Turkey.

Apt, K. R., & van Emden, M. H. (1982). Contributions to the Theory of Logic Programming. *Journal of the ACM, 29*(3), 841-862.

Arenas, M., Bertossi, L., Chomicki, J.(1999). Consistent query answers in inconsistent databases. *PODS '99: Proceedings of the eighteenth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, Philadelphia, Pennsylvania, United States.

Bagai, R., & Sunderraman, R. (1995). A paraconsistent relational data model. *International Journal of Computer Mathematics, 55*(3).

Bagai, R., & Sunderraman, R. (1996). Bottom-up computation of the fitting model for general deductive databases. *Journal of Intelligent Information Systems, 6*(1), 59–75.

Benjelloun, O., Sarma, A. D., Halevy, A., Theobald, M., & Widom, J. (2008). Databases with Uncertainty and Lineage. *VLDB Journal, 17*(2), 243-264.

Chan, E. P. F. (1993). A Possible World Semantics for Disjunctive Databases, IEEE Trans. Knowl. *Data Eng., 5*(2), 282-292.

Codd, E. F. (1970). A relational model for large shared data banks. Comm. of the ACM, 13(6):377–387.

Fitting M (1985). A Kripke-Kleene semantics for logic programs. Journal of Logic Programming, 4:295–312.

Grant J. and Subrahmanian V.S.(2000). Applications of Paraconsistency in Data and Knowledge Bases, Synthese 125:121-132.

Imieli´nski T. and Lipski W.(1984). Incomplete information in relational databases. J. ACM, 31(4):761–791.

Liu K.-C. and Sunderraman R.(1990). Indefinite and maybe information in relational databases. ACM Trans. Database Syst., 15(1):1–39.

Liu K.-C. and Sunderraman R.(1991). A generalized relational model for indefinite and maybe information. IEEE Trans. Knowl. Data Eng., 3(1):65–77.

Lloyd J.W (1987). Foundations of Logic Programming. Springer Verlag, second edition.

Maier D. (1983) The Theory of Relational Databases, Computer Science Press.

Reiter R.(1987). On closed world data bases. Readings in nonmonotonic reasoning, pages 300–310. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

Sarma A.D and Benjelloun O. and Halevy A. and Widom J. (2006). Working Models for Uncertain Data, Proceedings of the 22nd International Conference on Data Engineering ICDE'06, Washington, DC, USA.

Ullman J.D (1988). Principles of Database and Knowledge-base Systems , Volume 1. Computer Science Press.

Vadaparty K.V. and Naqvi S.A.(1995). Using Constraints for Efficient Query Processing in Nondeterministic Databases, IEEE Trans. Knowl. Data Eng., 7(9):860-864.

Viswanath N. and Sunderraman R.(2007). Query Processing in Paraconsistent Databases in the Presence of Integrity Constraints, In the Proceedings of the Nineteenth International Conference on Software Engineering and Knowledge Engineering SEKE 2007, Boston, USA.

KEY TERMS

Closed World Assumption (CWA): The closed world assumption is the presumption that what is not currently known to be true is false.

Credulous Reasoning: Accepting a set of beliefs from a theory that are the beliefs of some rational reasoner.

Deductive Database: A generalization of relational databases that includes both facts and rules from which new facts can be inferred.

Extensional Database: The facts stored in a deductive database.

Incompleteness: A database is incomplete if there is a sentence whose truth value cannot be ascertained.

Inconsistency: A state in which both a sentence and its negation are derivable from the theory.

Intensional Database: The rules in a deductive database from which new facts may be inferred.

Open World Assumption (OWA): The open world assumption is the view that what is stated in the database is what is known; everything else is unknown.

Paraconsistency: An inconsistency-tolerant logical notion in which the concept of "a contradiction entails everything" is dropped.

Query: A question posed in order to retrieve answers from the database, usually represented as a formula in first order logic.

Relational Algebra: An abstract query language for relational databases.

Relational Data Model: A database model based on first order logic and set theory.

Skeptical Reasoning: Accepting the set of beliefs from a theory that are a part of the beliefs of every rational reasoner.

Chapter IV
Managing Temporal Data

Abdullah Uz Tansel
Baruch College, CUNY, USA

INTRODUCTION

In general, databases store current data. However, the capability to maintain temporal data is a crucial requirement for many organizations and provides the base for organizational intelligence. A **temporal database** maintains time-varying data, that is, past, present, and future data. In this chapter, we focus on the relational data model and address the subtle issues in modeling and designing temporal databases.

A common approach to handle temporal data within the traditional relational databases is the addition of time columns to a relation. Though this appears to be a simple and intuitive solution, it does not address many subtle issues peculiar to temporal data, that is, comparing database states at two different time points, capturing the periods for concurrent events and accessing times beyond these periods, handling multi-valued attributes, coalescing and restructuring temporal data, and so forth, [Gadia 1988, Tansel and Tin 1997].

There is a growing interest in temporal databases. A first book dedicated to temporal databases [Tansel at al 1993] followed by others addressing issues in handling time-varying data [Betini, Jajodia and Wang 1988, Date, Darwen and Lorentzos 2002, Snodgrass 1999].

TIME IN DATABASES

The set T denotes time values and it is a total order under '$\leq$' relationship. Because of its simplicity, we will use natural numbers to represent time {*0, 1 ... now*}. The symbol *0* is the relative origin of time and *now* is a special symbol that represents the current time. *Now* advances according to the

time granularity used. There are different time granularities, such as seconds, minutes, hours, days, month, year, etc. (for a formal definition see [Betini, Jajodia and Wang 1988]).

A subset of *T* is called a *temporal set*. A temporal set that contains consecutive time points $\{t_1, t_2 ... t_n\}$ is represented either as a closed interval $[t_1, t_n]$ or as a half open interval $[t_1, t_{n+1})$. A *temporal element* [Gadia 1988] is a temporal set that is represented by the disjoint maximal intervals corresponding to its subsets having consecutive time points. Temporal sets, intervals, and temporal elements can be used as time stamps for modeling temporal data and are essential constructs in temporal query languages. Temporal sets and temporal elements are closed under set theoretic operations whereas intervals are not. However, intervals are easier to implement. Time intervals, hence temporal elements and temporal sets, can be compared. The possible predicates are *before*, *after*, *meet*, *during*, etc. [Allen 1983]. An interval or a temporal set (element) that includes *now* expends in its duration. Other symbols such as *forever* or *until changed* are also proposed as alternatives to the symbol *now* for easier handling of future data.

There are various aspects of time in databases [Snodgrass 1987]. *Valid time* indicates when a data value becomes effective. It is also known as *logical* or *intrinsic time*. On the other hand, the *transaction time* (or *physical time*) indicates when a value is recorded in the database. *User defined time* is application specific and is an attribute whose domain is time. Temporal databases are in general append-only that is, new data values are added to the database instead of replacing the old values. A database that supports valid time keeps historical values and is called a *valid time* (*historical*) database. A *rollback* database supports transaction time and can roll the database back to any time. Valid time and transaction time are orthogonal. Furthermore, a bitemporal database that supports both valid time and transaction time is capable of handling retroactive and post-active changes on temporal data. In the literature, the term temporal database is generically used to mean a database with some kind of time support.

In this chapter we focus our discussion on the valid time aspect of temporal data in relational databases. However, our discussion can easily be extended to databases that support transaction time or both as well.

REPRESENTING TEMPORAL DATA

A *temporal atom* is a time stamped value, <*t*, *v*> and represents a temporal value. It asserts that the value *v* is valid over the period of time stamp *t* that can be a time point, an interval, temporal set, or a temporal element. Time points are only suitable for values that are valid at a time point not over a period. Time can be added to tuples or attributes and hence, temporal atoms can be incorporated differently into the relational data model. To represent temporal atoms in tuple time stamping, a relation is augmented with two attributes that represents the end points of an interval or a time column whose domain is intervals, temporal sets, or temporal elements (temporally ungrouped). Figure 1 depicts salary (SAL) history of an employee, E1 where intervals or temporal elements are used as time stamps with a time granularity of month/year. Salary is 20K from 1/01 to 5/02 and 8/02 to 6/03. The discontinuity is because the employee quitted at 6/02 and came back at 8/02. The salary is 30K since 6/03. Figure 2 gives the same salary history in attribute time stamping (temporally grouped). An attribute value is a set of temporal atoms. Each relation has only one tuple that carries the entire history. It is also possible to create a separate tuple for each time stamped value (temporal atom) in the history, i.e. three tuples for Figure 2.b (two tuples for Figure 2.c). Temporally grouped data models are more expressive than temporally ungrouped data models but their data structures are also more complex [Clifford, Croker, and Tuzhilin 1993].

Figure 1. Salary in tuple time stamping

E#	SAL	FROM	TO
E1	20K	1/01	5/02
E1	20K	8/02	6/03
E3	30K	6/03	now

a. intervals

E#	SAL	TIME
E1	20K	{[1/01, 5/02) U[8/02, 6/03)}
E1	30K	{[6/03, now]}

b. Temporal Elements

Figure 2. Salary in attribute time stamping

E#	SALARY
E1	{<[1/01, 5/02), 20K>
	<[8/02, 6/03), 20K>
	<[6/03, *now*], 30K>}

a. Intervals

E#	SALARY
E1	{<{[1/01, 5/02) ∪ [8/02, 6/03)}, 20K>
	<{[6/03, *now*]}, 30K>}

b. Temporal elements

One noteworthy aspect of data presented in Figure 2 is that the timestamps are glued to attribute values. In other words attribute values are temporal atoms. In forming new relations as a result of query expressions these timestamps stay with the attribute values. On the other hand, in tuple time stamping a timestamp may be implicit (glued) or explicit (unglued) to tuples. This is a design choice and the relations in figure 1 can be interpreted as having implicit or explicit time stamps. An implicit time stamp is not available to the user as a column of a relation though the user can refer to it. On the other hand an explicit time stamp is like any other attribute of a relation and it is defined on a time domain. Implicit time stamps restricts the time of a new tuple created from two constituent tuples, since each tuple may not keep its own timestamp and a new timestamp needs to be assigned to the resulting tuple. Explicit timestamps allow multiple timestamps in a tuple. In this case, two tuples may be combined to form a new tuple, each carrying its own time reference. However, the user needs to keep track of these separate time references.

TEMPORAL RELATIONS

In Figure 3, we show some sample employee data for the EMP relation over the scheme E# (Employee number), ENAME (Employee name), DNAME (Department name) and SALARY. E# and ENAME are (possibly) constant attributes, whereas DNAME and SALARY change over time. In EMP relation, temporal elements are used in temporal atoms for representing temporal data. As is seen, there are no department values for Tom in the periods [2/02, 4/02) and [8/02, *now*]. Perhaps, he was not assigned to a department during these time periods. Time stamp of E# represents the lifespan of an employee that is stored in the database. Note that EMP is a nested [NINF] relation. It is one of the many possible relational representations of the employee data [Gadia 1988, Clifford and Tansel, 1985, Tansel 2004]. Figure 4 gives, in tuple time stamping, three 1NF relations, EMP_N, EMP_D, and EMP_S for the EMP relation of Figure 3 [Lorentzos and Johnson 1987, Navathe and Ahmed 1987, Sarda 1987, Snodgrass 1988]. In Figure 3 temporal sets

Figure 3. The EMP relation in attribute time stamping

E#	ENAME	DNAME	SALARY
<[1/01,*now*], 121>	Tom	<[1/01, 2/02), Sales>	<[1/01, 5/02), 20K>
		<[4/02, 8/02), Mktg>	<[5/02, 7/02), 25K>
			<[7/02, *now*], 30K>
<[3/03,8/03), 133>	Ann	<[3/03, 8/03), Sales>	<[3/03, 8/03), 35K>
<[8/02, *now*], 147>	John	<[8/02, *now*], Toys>	<[8/02, *now*], 42K>

Figure 4. The EMP relation in tuple time stamping

E#	NAME
121	Tom
133	Ann
147	John

(a) EMP N Relation

E#	DNAME	START	END
121	Sales	1/01	2/02
121	Marketing	4/02	8/02
133	Sales	3/03	8/03
147	Toys	8/03	*Now*

(b) EMP D Relation

E#	SALARY	START	END
121	20K	1/01	5/02
121	25K	5/02	7/02
121	30K	7/02	*Now*
133	35K	3/03	8/03
147	42K	1/01	Now

(c) EMP S Relation

(elements) can also be used as the time reference. Similarly, in the relations of Figure 4 intervals or temporal sets (elements) can also be used as the time reference in a time attribute that replaces the Start and End columns.

Note that in tuple time stamping a relation may only contain attributes whose values change at the same time. However, attributes changing at different times require separate relations. Each particular time stamping method imposes restrictions on the type of base relations allowed as well as the new relations that can be generated from the base relations. The EMP relation in Figure 3 is a unique representation of the employee data where each tuple contains the entire history of an employee [Clifford and Tansel 1985, Tansel 1987, Gadia 1988]. E# is a temporal grouping identifier regardless of the timestamp used [Clifford, Croker and Tuzhilin 1993]. In the case of tuple time stamping an employee's data is dispersed into several tuples, i.e., there are three salary tuples for the employee 121 in Figure 4.c. These tuples belong to the same employee since their E# values are equal.

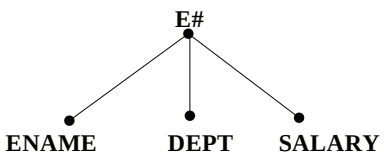
For the relations in Figures 3 and 4 there are many other possible representations that can be obtained by taking subsets of temporal elements (intervals) and creating several tuples for the same employee. These relations are called weak relations [Gadia 1988]. Though they contain the same data as the original relation in unique representation, query specification becomes very complex. Weak relations naturally occur in querying a temporal database. Weak relations can be converted to an equivalent unique relation by coalescing tuples that belong to the same object (employee) into one single tuple [Sarda 1987, Bohlen, Snodgrass and Soo 1996]. In coalescing, a temporal grouping identifier such as E# is used to determine the related tuples.

DESIGNING TEMPORAL RELATIONS

Design of relational databases is based on functional and multivalued dependencies. These dependencies are used to decompose relations to 3NF, BCNF or 4NF that have desirable features. In temporal context, functional dependencies turn into multivalued dependencies whereas multivalued dependencies stay as multivalued dependencies. For instance, in the current employee data depicted in Figure 3, E# → SALARY holds. When temporal data for the employees are considered, this functional dependency turns into a multivalued dependency, i.e. E# →→ (Time, SALARY).

Designing nested relations based on functional and multivalued dependencies are explored in [Ozsoyoglu and Yuan 1987]. They organize the attributes into a scheme tree where any root to leaf branch is a functional or multivalued dependency. Such a scheme tree represents a N1NF relation without undesirable data redundancy. This methodology is applied to the design of temporal relations in [Tansel and Garnett 88]. Let E# be a temporal grouping identifier. Figure 5 gives the scheme tree for the EMP relation of Figure 3. The dependencies in EMP are E# → ENAME, E#→→ (Time, DNAME), and E# →→ (Time, SALARY). Naturally, the flat equivalent of EMP is not in 4NF since it includes multivalued dependencies. When we apply 4NF decomposition on EMP we obtain flat relation schemes (E#, ENAME), (E#, (Time, DEPT)), and (E#, (Time, SALARY)) which are all in 4NF. In fact, these are the EMP_N, EMP_D, and EMP_S relations in Figure 4 where Time is separated as two additional columns. This is the reason for including only one time varying attribute in a temporal relation in case of tuple timestamping.

Figure 5. Scheme Tree for EMP



Thus, any attribute involved in a multivalued dependency on the temporal grouping identifier is placed into a separate relation in the case of tuple time stamping as is seen in Figure 4. If attribute time stamping and nested relations are used all the time dependent attributes that belong to similar entities, such as employees can be placed in one relation as seen in Figure 3.

REQUIREMENTS FOR TEMPORAL DATA MODELS

A temporal database should meet the following requirements [Tansel and Tin 1997]. Depending on application needs some of these requirements can be relaxed. Let DB_t denote the database state at time t:

1. The data model should be capable of modeling and querying the database at any instance of time, i.e., D_t. Note that when t is *now*, D_t corresponds to a traditional database.
2. The data model should be capable of modeling and querying the database at two different time points, intervals and temporal set (elements) i.e., D_t and $D_{t'}$ where $t \neq t'$.
3. The data model should allow different periods of existence in attributes within a tuple, i.e., non-homogenous (heterogeneous) tuples should be allowed. Homogeneity requires that all the attribute values in a tuple should be defined on the same period of time [Gadia 1988].
4. The data model should allow multi-valued attributes at any time point, i.e., in D_t.
5. A temporal query language should have the capability to return the same type of objects it operates on. This may require coalescing several tuples to obtain the desired result.

6. A temporal query language should have the capability to regroup the temporal data according to a different grouping identifier that could be one of the attributes in a temporal relation.
7. The model should be capable of expressing set-theoretic operations, as well as set comparison tests, on the timestamps, be it time points, intervals, or temporal sets (elements).

TEMPORAL QUERY LANGUAGES

Modeling of temporal data presented in the previous sections also has implications for the temporal query languages that can be used. A temporal query languages is closely related how the temporal atoms (temporal data) are represented, i.e. the type of timestamps used, where they are attached (relations, tuples, or attributes), and whether temporal atoms are kept atomic or broken into their components. This in turn determines possible evaluation (semantics) of temporal query language expressions. There are two commonly adopted approaches: 1) Snapshot evaluation that manipulates the snapshot relation at each time point, like Temporal Logic [Gadia 1986, Snodgrass 1987, Clifford, Croker and Tuzhilin 1993]; 2) Traditional evaluation that manipulates the entire temporal relation much like the traditional query languages [Tansel 1986, Tansel 1997, and Snodgrass 1987]. The syntax of a temporal query language is designed to accommodate a desired type of evaluation.

Temporal Relational Algebra includes temporal versions of relational algebra operations in addition to special operations for reaching time points within intervals or temporal elements [Lorentzos, and Mitsopoulos, 1997, Sarda 1997, Tansel 1997], slicing times of attributes or tuples [Tansel 1986], rollback to a previous state in case of transactions time databases, and temporal aggregates. There are also projection and selection operations on the time dimension. These operations are all incorporated into temporal relational calculus languages too.

There are many language proposals for temporal databases [Lorentzos, and Mitsopoulos 1997, Snodgrass 1995, Snodgrass 1987, Tansel, Arkun and Ozsoyoglu 1989]. SQL2 has a time data type for implementing tuple time stamping with intervals that is used in TSQL2 [Snodgrass 1995]. SQL3 has the capabilities to implement tuple timestamping, temporal elements as well as attribute time stamping and nested relations.

IMPLEMENTATION OF TEMPORAL RELATIONS

Temporal relations that use tuple timestamping have been implemented on top of conventional DBMS [Snodgrass 1995]. This was doable because, augmenting a relation with two additional columns representing the end points of time intervals is available in any DBMS. However, recently, commercial DBMS include object relational features that allow definition of temporal atoms, sets of temporal atoms and temporal relations that are based on attribute timestamping. Following code illustrates the definition of EMP relation in ORACLE 9i. Lines 1 and 2 define temporal atoms with data types of Varchar and Number. Lines 3 and 4 define the sets of temporal atoms for representing the department and salary histories. Finally line 5 creates the EMP table. Note that, transaction time bitemporal relations may similarly be defined [Tansel & Atay 2006].

```
1.  CREATE TYPE Temporal_Atom_Varchar
AS OBJECT (
Lowe_Bound TIMESTAMP,
    Upper_Bound TIMESTAMP,
    TA_Value VARCHAR(20)   );
2.  CREATE  TYPE  Temporal_Atom_Number
AS OBJECT (
Lowe_Bound TIMESTAMP,
    Upper_Bound TIMESTAMP,
    TA_Value NUMBER        );
```

```
3.  CREATE TYPE Dept _ History AS TABLE OF
Temporal _ Atom _ Varchar;
4.  CREATE TYPE Salary _ History AS TABLE
OF emporal _ Atom _ Number;
5.  CREATE TABLE Emp (
E# NUMBER,
  Name VARCHAR(20),
  Dept Dept _ History,
  Salary Salary _ History   );
```

CONCLUSION

In this chapter, we have examined the management of temporal data by using relational database theory. We have covered the two types of approached to modeling temporal data: tuple timestamping and attribute timestamping. We also discussed design of temporal databases and temporal query languages. The feasibility of implementing tuple timestamping has already been demonstrated. We also show the feasibility of implementing temporal atoms and attribute timestamping by using object relational databases that are currently available in the market.

ACKNOWLEDGMENT

Research is supported by the grant# 68180 00-37 from the PSC-CUNY Research Award Program.

REFERENCES

Allen, J. F. (1983). Maintaining knowledge about temporal intervals. *Communications of the ACM, 26*(11), 832-843.

Betini, C., Jajodia, S., & Wang, S. (1988). Time granularities in databases, data mining and temporal reasoning. Springer Verlag.

Bohlen, M. H., Snodgrass, R. T., & Soo, M. D. (1996). Coalescing in temporal databases. *In Proceedings of International Conference on Very Large Databases.*

Clifford J., & Tansel, A. U. (1985). On an algebra for historical relational databases: Two views. *In Proceedings of ACM SIGMOD International Conference on Management of Data*, 247-265.

Clifford, J., Croker, A., & Tuzhilin, A. (1993). On completeness of historical data models. *ACM Transactions on Database Systems, 19*(1), 64-116.

Date, C. D., Darwen, H., & Lorentzos, N. (2003). *Temporal data and the relational data model.* Morgan Kaufmann, 2002.

Etzion, O., Jajodia, S., & Sripada, S. (1998). *Temporal databases: Research and practice.* Springer Verlag.

Gadia, S. K (1988). A homogeneous relational model and query languages for temporal databases. *ACM Transactions on Database Systems, 13*(4), 418-448.

Lorentzos, N. A., & Johnson, R. G. (1988). Extending relational algebra to manipulate temporal data. *Information Systems, 13*(3), 289-296.

Lorentzos, N. A., & Mitsopoulos, Y. G. (1997). SQL extension for interval data. *IEEE Transactions on Knowledge and Data Engineering, 9*(3), 480-499.

McKenzie, E., & Snodgrass, R. T. (1991). An evaluation of relational algebras incorporating the time dimension in databases. *ACM Computing Surveys, 23*(4), 501-543.

Navathe, S. B., & Ahmed, R. (1987). TSQL-A language interface for history databases. *Proceedings of the Conference on Temporal Aspects in Information Systems,* 113-128.

Ozsoyoglu, M. Z., & Yuan, L-Y (1987). A new normal form for nested relations. *ACM Transactions on Database Systems, 12*(1), January 1987.

Sarda, N. L. (1987). Extensions to SQL for historical databases. *IEEE Transactions on Systems, 12*(2), 247-298.

Snodgrass, R. T. (1987). The temporal query language Tquel,. *ACM Transactions on Database Systems, 12*(2), 247-298.

Snodgrass, R. T. (1999). Developing time oriented applications in SQL. Morgan Kaufmann.

Snodgrass, R. T. (1995). *The TSQL2 temporal query language*. Kluwer 1995.

Tansel, A.U (1986). Adding time dimension to relational model and extending relational algebra. *Information Systems 11*(4), 343-355.

Tansel, A. U (1997). Temporal relational data model. *IEEE Transactions on Knowledge and Database Engineering, 9*(3), 464-479.

Tansel, A. U (2004). On handling time-varying data in the relational databases. *Journal of Information and Software Technology, 46*(2), 119-126.

Tansel, A. U., Arkun, M. E., & Ozsoyoglu, G. (1989). Time-by-Example query language for historical databases. *IEEE Transactions on Software Engineering, 15*(4), 464-478.

Tansel, A. U., & Garnett, L. (1989). Nested temporal relations. I*n Proceedings of ACM feshi SIGMOD International Conference on Management of Data*, 284-293.

Tansel, A. U., & Tin, E. Expressive power of temporal relational query languages. *IEEE Transactions on Knowledge and Data Engineering, 9*(1), 120-134.

Tansel, A. U et al. (1993). (Ed.). *Temporal databases: Theory, design and implementation*. Benjamin/Cummings.

Tansel, A. U, & Eren-Atay, C., (2006). Nested bitemporal relational Algebra. *ISCIS 2006*, 622-633.

KEY TERMS

For a detailed coverage of the terminology, see [appendix A in Tansel et al 1993] and [pp. 367 – 413 in Etzion, Jajodia and Sripada 1998].

Coalescing: Combining tuples whose times are contiguous or overlapping into one tuple whose time reference includes the time of constituent tuples.

Homogenous Temporal Relation: Attribute values in any tuple of a relation are all defined on the same period of time. In a heterogeneous relation, attribute values in a tuple may have different time periods.

Rollback Operation: Returns a relation that is recorded as of a given time point, interval, or temporal element in a database that supports transaction time.

Temporal Data Model: A data model with constructs and operations to capture and manipulate temporal data.

Temporal Database: A database that has transaction time and/or valid time support. In the literature it is loosely used to mean a database that has some kind of time support.

Temporal Element: Union of disjoint time intervals where no two time intervals overlap or meet.

Time Granularity: Unit of time such as seconds, minutes, hours, days, month, year, etc. Time advances by each clock tick according to the granularity used.

Time Interval (period): The consecutive set of time points between a lower bound (l) and an upper bound (u) where l < u. The closed interval [l, u] includes l and u whereas the open interval (l, u) does not include l and u. Half open intervals, [l, u) or (l, u] are analogously defined.

Transaction Time: Designates the time when data values are recorded in the database.

Valid Time: Designates when data values become valid.

Chapter V
Data Reengineering of Legacy Systems

Richard C. Millham
Catholic University of Ghana, Ghana

INTRODUCTION

Legacy systems, from a data-centric view, could be defined as old, business-critical, and stand-alone systems that have been built around legacy databases, such as IMS or CODASYL, or legacy database management systems, such as ISAM (Brodie & Stonebraker, 1995). Because of the huge scope of legacy systems in the business world (it is estimated that there are 100 billion lines of COBOL code alone for legacy business systems; Bianchi, 2000), data reengineering, along with its related step of program reengineering, of legacy systems and their data constitute a significant part of the software reengineering market.

Data reengineering of legacy systems focuses on two parts. The first step involves recognizing the data structures and semantics followed by the second step where the data are converted to the new or converted system. Usually, the second step involves substantial changes not only to the data structures but to the data values of the legacy data themselves (Aebi & Largo, 1994).

Borstlap (2006), among others, has identified potential problems in retargeting legacy ISAM data files to a relational database. Aebi (1997), in addition to data transformation logic (converting sequential file data entities into their relational database equivalents), looks into, as well, data quality problems (such as duplicate data and incorrect data) that is often found with legacy data.

Due to the fact that the database and the program manipulating the data in the database are so closely coupled, any data reengineering must address the modifications to the program's data access logic that the database reengineering involves (Hainaut, Chandelon, Tonneau, & Joris, 1993).

In this article, we will discuss some of the recent research into data reengineering, in particular the transformation of data, usually legacy data from a sequential file system, to a different type of database system, a relational database. This article outlines the various methods used in data reengineering to transform a legacy database (both its structure and data values), usually stored as sequential files, into a relational database structure. In addition, methods are outlined to transform the program logic that accesses this database to access it in a relational way using WSL (wide spectrum language, a formal language notation for software) as the program's intermediate representation.

RELATED WORK

In this section, we briefly describe the various approaches that various researchers have proposed and undertaken in the reengineering of legacy data. Tilley and Smith (1995) discuss the reverse engineering of legacy systems from various approaches: software, system, managerial, evolution, and maintenance.

Because any data reengineering should address the subsequent modifications to the program that the program's data access' logic entails, Hainaut et al. (1993) have proposed a method to transform this data access logic, in the form of COBOL read statements, into their corresponding SQL relational database equivalents.

Hainaut et al. (1993) identify two forms of database conversion strategies. One strategy (physical conversion) is the physical conversion of the database where each construct of the source database is translated into the closest corresponding construct of the target database without any consideration of the semantic meaning of the data being translated. One of the problems with this strategy is that the resulting target database produced is of very low quality. The second strategy (conceptual conversion) is the recovery of precise semantic information, the conceptual schema, of the source database through various reverse engineering techniques, and then the development of the target database, using this conceptual schema, using standard database development techniques. This strategy produces a higher quality database with full documentation as to the semantic meaning of the legacy data, but this approach is more expensive in terms of time and effort that it entails (Hainaut et al., 1993a). Hainaut et al.'s approach first uses the physical conversion strategy to convert data and then uses a trace of the program, which accesses the legacy data, in order to determine how the data are used and managed. In this way, additional structures and constraints are identified through the procedural code. Through an analysis of the application's variable dependency graph and of the record and file definitions, data fields are refined, foreign keys are determined, and constraints on multivalued fields are discovered. During the database conceptualization phase, the application's physical constructs of indexes and files are removed and the program's objects of arrays, data types, fields, and foreign keys are transformed into their database equivalents (Hainaut et al., 1993a).

Initially, database reengineering focused on recognizing the legacy database structure and transforming these structures into a new model (Aiken & Muntz, 1993; Joris, 1992; Pomerlani & Blaha, 1993; Sabanis & Stevenson, 1992). The values of legacy data were used solely to identify the legacy system's dependencies in terms of keys between records (Pomerlani & Blaha).

Aebi and Largo (1994), in the transformation of database structures, recognize that the transformation of structural schemas involves many issues. The first issue is that the different attributes and entities of the old system must be mapped to the new schema of the transformed database. Constraints, during the migration from the old to the new system, may be added, dropped, or changed. Entity sets in the new system may be identified by new attributes or by old attributes with changed domains or data types.

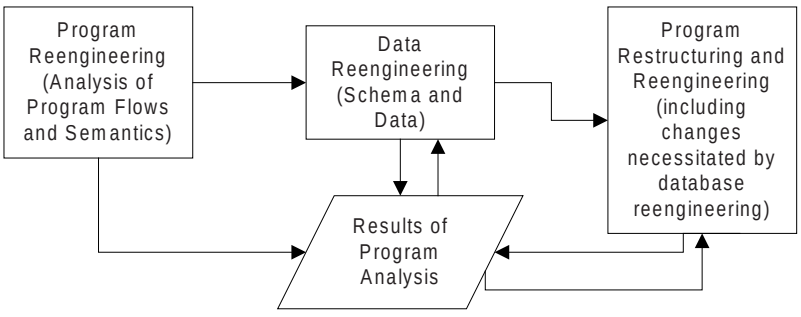
Wu et al. (1997), with their "butterfly" approach, assume that the legacy data are the most important part of the legacy system and it is the schema rather than the values of this legacy data that are the most crucial. This legacy data are modified in successive iterations with the legacy data being frozen and used for reading purposes only. The "Chicken Little" strategy allows the legacy system to interact with the target system during migration, using a gateway to serve as a mediator. This gateway is used to translate and redirect calls from the legacy system to the target database system, and then the gateway translates the results of the target database for use by the legacy system and by the legacy database. Although the legacy system is allowed to interact with the target database during migration, each data access involves two database accesses: one to the target database and another to the legacy database (Bisbal, Lawless, Wu, & Grimson, 1999).

Bianchi, Caivano, and Visaggio (2000) proposed a different method of data reengineering where the data structures are reengineered rather than simply migrated. This reengineering involves the use of several steps. The first step is analyzing the legacy data through monitoring of calls to the legacy database from the legacy system (dynamic trace). This dynamic trace is used to identify which data could be classified as conceptual (data specific to the application domain and that describe specific application concepts), control (data that are used for program decisions or to record an event), structural (data that are used to organize and support the data structures of the system), or calculated (data that are calculated by the application). The second step involves redesigning the legacy database after identifying the dependencies among the data. This dependency diagram is then converted to a target database schema. After the legacy data are migrated to the new target schema, the legacy system is then altered such that data accesses to the target database reflect the new database schema (Bianchi et al.). Using dynamic traces along with data flow and dependency analyses are some successful techniques used in data reverse engineering that have been obtained from the program understanding phase of program reverse engineering (Cleve, Henrard, & Hainaut, 2006; Millham, 2005).

Aebi and Largo (1994) also recognize problems with the transformation of database values. Some problems include duplicate data being used in the same context, changes in the primary keys of tables often entailing changes in foreign keys of the same table, values of the attribute possibly exceeding its given range, different encoding schemes for data values may be used, recording errors incorporated into the existing data, and no existing distinction between unknown and null values.

Hainaut et al. (1993b) outline some practical schema-relational transformations, such as project-join, extension transformation, and identifier substitution, that are dependent on the file manager such as CODASYL, relational, TOTAL/IMAGE, and IMS DBMS in order to function. For other databases, such as COBOL file structure databases, such transformations are impractical. In the case of COBOL file structure databases, multivalued attributes can be represented by list attributes only. Referential constraints upon files can be detected through careful analysis of the COBOL procedural code, file contents, and secondary keys. Identifying one-to-many relationships in files is accomplished through depicting multivalued, compound attributes B of A as a many-to-one relationship of B to A. Multirecord types within a sequential file may be portrayed as a many-to-one relationship. Multivalued attributes, such as those used in foreign keys, are represented in the relational target model as rows in a separate table with a link to the main referencing table. If the source database is of a network type, a recursive relational type may be represented by an entity type and two one-to-many or one-to-one relational types (Hainaut et al., 1993b).

Figure 1. Flow of reengineering processes and data store



MAIN FOCUS: DATA REENGINEERING WITHIN PROGRAM TRANSFORMATION

Because the semantic understanding of the data to be reengineered depends to a large degree upon the program accessing this data and because the program accessing the target database needs to be modified in order to account for reengineered data, one of the areas of focus must be the analysis of the underlying program and its usage of the data (Bianchi et al., 2000; Hainaut et al., 1993b). One of the problems with analyzing a program for data access usage of legacy data is determining exactly when and where this legacy data are accessed. Design pattern analysis of code has been proposed as a method of determining what sections of code might be used for accessing this legacy data (Jarazabek & Hitz, 1998).

Another method is to convert the programming-language-specific code into a programming-language-independent intermediate representation. This intermediate representation can be in a formal notation. Formal notations have been used to specify the transformation of both programs and their associated data mathematically. One advantage of these transformations are that these transformations are generic (programming-language independent) such that a transformation for a construct accessing a legacy database to a construct accessing the target database will be the same regardless of the types of legacy and target databases. For example, the formal notation WSL (Ward, 1992), with a proven set of program transformations, has been used to transform a procedurally structured and driven COBOL legacy system using a WSL intermediate program representation into an object-oriented, event-driven system (Millham, 2005).

Although WSL has been extended to represent data structures, such as records (Kwiatkowski & Puchalski, 1998), little attention has been paid to transforming the data structures as represented in WSL as their original sequential file form into another database format. However, in Millham (2005), the sequential records and their accesses by the application have been specified in terms of sets and set operations. Codd, in his relational database model, has specified this model in terms of sets and relational calculus; WSL, in turn, may be specified in terms of sequences, sets, and set operations (Ward, 1992). Consequently, a method to integrate the relational database model of Codd, in its formal notation involving database specification and operation, and WSL, in its formal notation involving program transformations and sequential file structures, could be accomplished. Other formal notations, similar to WSL, with the same representational capacity, could be utilized in the same manner. In this article, a method to transform a WSL-represented hierarchical database, with its underlying program logic, into its relational database equivalent is provided. This method, a combination of the data reverse engineering techniques of Hainaut et al. (1993) and Bianchi et al. (2000), occurs in conjunction with the program understanding phase of the WSL program reverse engineering process. During this program understanding phase, a static and dynamic analysis of the program code is undertaken. This analysis produces a dependency graph of variables and identifies which variables serve as control, structural, conceptual, and calculated field variables. While WSL, in its representation, does not distinguish between programs and file record types, the source code to WSL translation process keeps track of which records are of type

Figure 2. A sample COBOL record and its WSL equivalent

COBOL Record	WSL Equivalent
01 2000-USOC-DETAIL. 05 2000-USOC PIC X(06). 05 FILLER PIC X(14) VALUE SPACES.	Var struct Begin Var 2000-USOC Var Filler End

file record. Similarly, WSL is type-less but, during the source code to WSL representation, the original data types of the source code are recorded for later use during the data reengineering process (these processes are outlined in Figure 1).

In Figure 2, the data types of each record field, along with their default values, are recorded in a database for future use during the data reengineering process. Hence, it is possible to derive the program's file and record types, along with the foreign and primary keys, from the information obtained during these translation and analysis processes. If one, A, has an array field that forms a dependency in another record B, this dependency is depicted in a one-to-many relationship between records A and B. Anti-aliasing of records and their fields reduces the number of records that refer to the same file but use different names. Calculated data may appear as calculated fields in the database schema. Structural data are used to represent the target database schema. Record fields that are used as control data may indicate a relationship, involving the control data, between the record that defines this control data and any record(s) enclosed within the control block that is governed by that control data (Millham, 2005). Constraints on data may be determined through an analysis of their declaration within a program or through the use of Hainault et al.'s (1993b) methods. Because this data reengineering process occurs in conjunction with program reengineering, any changes to the underlying database structure can easily be propagated to the program reengineering phase where the program data access logic will be altered to reflect these database changes. Through this data reengineering, a more logical database structure is derived along with the necessary subsequent program data access transformations. These transformations are based on a formal, platform-independent representation (WSL) such that the source system, whether COBOL or assembly language, is not important and the advantages of a formal notation, such as preciseness, are achieved.

Wong and Sun (2006) have been working on detecting data dependencies in programs using a hybrid UML (unified modeling language) collaboration and activity diagram that is expressed in a platform-independent XML (extensible markup language) markup graph.

FUTURE TRENDS

Because up to 50% of a legacy system's maintenance costs, in terms of program and database changes, can be attributed to changing business rules, there is a strong need to adopt new techniques in reengineering legacy data and their associated applications. Jarazabek and Hitz (1998) propose the use of domain analysis and the use of generic architectural design techniques in reengineering as a method of substantially reducing these maintenance costs. Atkinson, Bailes, Chapman, Chilvers, and Peake's (1998) preferred approach is to develop interfaces to general persistent data repositories in the direction of generic reengineering environment design. From these interfaces, binding between the data stores, represented in the target databases, and the underlying application, which will access this target database, can be made. Another method is to use formal-language intermediate representations that can represent both the database and program along with a set of corresponding transformations to transform the legacy to the target database and transform the program accessing the legacy database to a program capable of accessing the new target database without errors. Through the use of formal-language intermediate representations, a set of generic reengineering tools for both the legacy database and its underlying application(s) could be designed.

CONCLUSION

Data reengineering has gone beyond a simple physical conversion of a legacy database to its target database. Both Hainaut et al. (1993a) and Wu et al. (1997) utilize an analysis of the underlying application(s) accessing the legacy database

in order to determine the database's underlying data types and foreign keys. Bianchi et al. (2000) go further in using an analysis of the underlying application(s) accessing the legacy database in order to identify the categories of the data being used in the application such as whether the data is used as conceptual, structural, control, or calculated data and in order to identify the dependencies among them. In this manner, a better determination of the data usage can be made and a better target database schema can be derived from analysis of this usage.

Because the underlying application(s) that access the legacy database and the legacy database are so intrinsically linked during the data reengineering phase, there is a need to be able to analyze the applications, in a programming-language-independent way, for data usage and then transform these applications' data accesses and the legacy database, using a set of generic transformations and tools, to the target database. Formal notations (with their programming-language independence, sets of transformations, and basis in relational database theory and software reengineering) have been proposed as a means to generically analyze the application(s), once in the formal notation's intermediate representation, for data usage and then transform this analysis into an accurate and meaningful target database schema for database reengineering.

REFERENCES

Aebi, D. (1997). Data engineering: A case study. In C. J. Risjbergen (Ed.), *Proceedings in advances in databases and information systems.* Berlin, Germany: Springer Verlag.

Aebi, D., & Largo, R. (1994). Methods and tools for data value re-engineering. In *Lecture notes in computer science: Vol. 819. International Conference on Applications of Databases* (pp. 1-9). Berlin, Germany: Springer-Verlag.

Aiken, P., & Muntz, A. (1993). A framework for reverse engineering DoD legacy information systems. *WCRE*.

Atkinson, S., Bailes, P. A., Chapman, M., Chilvers, M., & Peake, I. (1998). A re-engineering evaluation of software refinery: Architecture, process and technology.

Behm, A., Geppert, A., & Diettrich, K. R. (1997). *On the migration of relational schemas and data to object-oriented database systems.* Proceedings of Re-Technologies in Information Systems, Klagenfurt, Austria.

Bianchi, A., Caivano, D., & Visaggio, G. (2000). Method and process for iterative reengineering of data in a legacy system. *WCRE*. Washington, DC.

Bisbal, J., Lawless, D., Wu, B., & Grimson, J. (1999). Legacy information systems: Issues and directions. *IEEE Software, 16*(5), 103-111.

Bohm, C., & Jacopini, G. (1966). Flow diagrams, Turing machines, and languages with only two formation rules. *CACM, 9*(5), 266.

Borstlap, G. (2006). *Understanding the technical barriers of retargeting ISAM to RDBMS.* Retrieved from *http://www.anubex.com/anugenio!technicalbarriers1.asp*

Brodie, M. L., & Stonebraker, M. (1995). *Migrating legacy systems: Gateways, interfaces, and the incremental approach.* Morgan Kaufmann.

Cleve, A., Henrard, J., & Hainaut, J.-L. (2006). Data reverse engineering using system dependency graphs. *WCRE*.

Hainaut, J.-L., Chandelon, M., Tonneau, C., & Joris, M. (1993a). Contribution to a theory of database reverse engineering. *WCRE*. Baltimore, MD.

Hainaut, J.-L., Chandelon, M., Tonneau, C., & Joris, M. (1993b). Transformation-based database reverse engineering. *Proceedings of the 12th International Conference on Entity-Relationship Approach* (pp. 1-12).

Janke, J.-H., & Wadsack, J. P. (1999). Varlet: Human-centered tool for database reengineering. *WCRE*.

Jarazabek, S., & Hitz, M. (1998). Business-oriented component-based software development and evolution. *DEXXA Workshop*.

Jeusfeld, M. A., & Johnen, U. A. (1994). *An executable meta model for reengineering of database schemas.* Proceedings of Conference on the Entity-Relationship Approach, Manchester, England.

Joris, M. (1992). Phenix: Methods and tools for database reverse engineering. *Proceedings 5th International Conference on Software Engineering and Applications*.

Kwiatkowski, J., & Puchalski, I. (1998). Pre-processing COBOL programs for reverse engineering in a software maintenance tool. *COTSR*.

Mehoudj, K., & Ou-Halima, M. (1995). Migrating data-oriented applications to a relational database management system. *Proceedings of the Third International Workshop on Advances in Databases and Object-Oriented Databases* (pp. 102-108).

Millham, R. (2005). *Evolution of batch-oriented COBOL systems into object-oriented systems through the unified modelling language.* Unpublished doctoral dissertation, De Montfort University, Leicester, England.

Pomerlani, W. J., & Blaha, M. R. (1993). An approach for reverse engineering of relational databases. *WCRE*.

Rob, P., & Coronel, C. (2002). *Database systems: Design, implementation, and management.* Boston: Thomas Learning.

Sabanis, N., & Stevenson, N. (1992). Tools and techniques for data remodeling COBOL applications. *Proceedings 5th International Conference on Software Engineering and Applications*.

Tilley, S. R., & Smith, D. B. (1995). *Perspectives on legacy system reengineering* (Tech. Rep.). Carnegie Mellon University, Software Engineering Institute.

Ward, M. (1992). *The syntax and semantics of the wide spectrum language* (Tech. Rep.). England: Durham University.

Weiderhold, G. (1995). Modelling and system maintenance. *Proceedings of the International Conference on Object-Orientation and Entity-Relationship Modelling*.

Wong, K., & Sun, D. (2006). On evaluating the layout of UML diagrams for program comprehension. *Software Quality Journal, 14*(3), 233-259.

Wu, B., Lawless, D., Bisbal, J., Richardson, R., Grimson, J., Wade, V., et al. (1997). The butterfly methodology: A gateway-free approach for migrating legacy information system. In *ICECOS* (pp. 200-205). Los Alamos, CA: IEEE Computer Society Press.

Zhou, Y., & Kontogiannis, K. (2003). *Incremental transformation of procedural systems to object-oriented platform.* Proceedings of COMPSAC, Dallas, TX.

KEY TERMS

Butterfly Approach: An iterative data reengineering approach where the legacy data are frozen for read-only access until the data transformation process to the target database is complete. This approach assumes that the legacy data are the most important part of the reengineering process and focuses on the legacy data structure, rather than its values, during its migration.

Chicken Little Approach: An approach that allows the coexistence of the legacy and target databases during the data reengineering phase through the use of a gateway that translates data access requests from the legacy system for use by the target database system and then translates the result(s) from the target database for use by the legacy system.

Conceptual Conversion Strategy: A strategy that focuses first on the recovery of the precise semantic meaning of data in the source database and then the development of the target database using the conceptual schema derived from the recovered semantic meaning of data through standard database development techniques.

Domain Analysis: A technique that identifies commonalties and differences across programs and data. Domain analysis is used to identify design patterns in software and data.

Legacy Data: Historical data that are used by a legacy system that could be defined as a long-term mission-critical system that performs important business functions and contains comprehensive business knowledge

Multivalued Attribute: When an attribute, or field, of a table or file may have multiple values. For example, in a COBOL sequential file, its corresponding record may have a field, A, with several allowable values (Y, N, D). Translating this multivalued attribute to its relational database equivalent model is difficult; hence, lists or linked tables containing the possible values of this attribute are used in order to represent it in the relational model.

Physical Conversion Strategy: A strategy that does not consider the semantic meaning of the data but simply converts the existing legacy constructs of the source database to the closest corresponding construct of the target database.

Chapter VI
Different Kinds of Hierarchies in Multidimensional Models

Elzbieta Malinowski
Universidad de Costa Rica, Costa Rica

INTRODUCTION

In the database design, the advantages of using conceptual models for representing users' requirements are well known. Nevertheless, even though data warehouses (DWs) are databases that store historical data for analytical purposes, they are usually represented at the logical level using the star and snowflake schemas. These schemas facilitate delivery of data for online analytical processing (OLAP) systems. In particular, hierarchies are important since traversing them, OLAP tools perform automatic aggregations of data using the roll-up and drill-down operations. The former operation transforms detailed data into aggregated ones (e.g., daily into monthly sales) while the latter does the opposite.

In spite of the advantages of star and snowflake schemas, there are some inconveniences in using them. For example, since these schemas are based on the relational logical model, some implementation details (e.g., foreign keys) must be considered during the design process. Further, the star and snowflake schemas are not adequate for representing different kinds of hierarchies existing in real-world applications. Therefore, users are not able to express their analysis needs, and consequently, developers cannot implement them.

We advocate that it is necessary to represent DW data requirements at the conceptual level.

The conceptual model should clearly distinguish different kinds of hierarchies since they exist in real-world situations and are important for DW and OLAP applications. Further, developers should be able to implement these hierarchies. Therefore, considering that DWs and OLAP can use relational storage, we present how hierarchies can be mapped to a relational model.

BACKGROUND

The star and snowflake schemas include relational tables known as fact and dimension tables. The fact table represents the focus of analysis (e.g., analysis of sales). It usually contains numeric data called measures (e.g., quantity). Dimension tables contain attributes that allow users to see measures from different perspectives (e.g., analyze sales in different stores). Since users usually start from a general view of data and then, if required, the detail explorations follow, dimensions may contain attributes that form hierarchies. OLAP tools allow users to traverse hierarchies, aggregating measures automatically. For example, "moving" (i.e., using the roll-up operation) from store to city, the quantity of sold products in each store will be added according to the cities where the stores are located.

Depending on whether hierarchies are represented using flat (Figure 1(a)) or normalized tables (Figure 1b)), the relational structure is called star or snowflake schemas, respectively. Nevertheless, both schemas are not adequate for representing different kinds of hierarchies existing in real-world situations. The star schema does not represent hierarchies clearly, and the hierarchy structure should be deduced based on the knowledge of the application domain. On the other hand, the snowflake schema only allows us to represent simple hierarchies such as Store, City, and State in Figure 1(b), even though there are different kinds of hierarchies in real-world applications.

There are several proposals of conceptual multidimensional models[1] that include hierarchies. Nevertheless, as we will see later, these models do not include all hierarchies as presented in this chapter. This lack of a general classification of hierarchies, including their characteristics at the schema and at the instance levels, leads to repeated research efforts in "rediscovering" hierarchies and providing solutions for managing them.

MAIN FOCUS

We first describe the MultiDim model, a conceptual multidimensional model used for representing requirements for DW and OLAP applications, including different kinds of hierarchies. Then, we present the hierarchy classification and refer in more detail to each hierarchy type. Last, we present mapping of these hierarchies to the relational model.

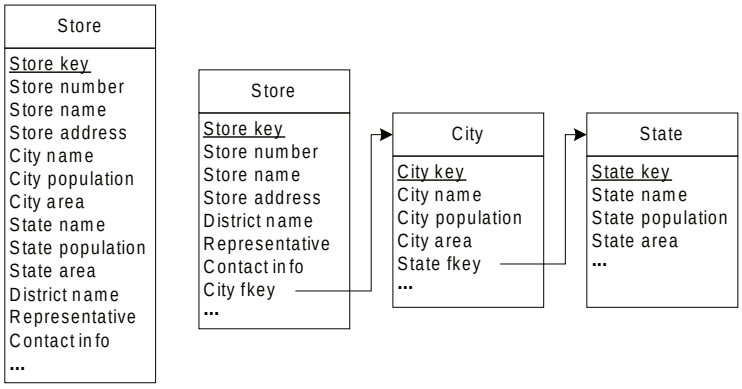
The MultiDim Model

To describe the MultiDim model (Malinowski & Zimányi, 2008), we use an example of a Sales DW shown in Figure 2 that contains different kinds of hierarchies; we refer to them in the next section.

A MultiDim schema is a finite set of dimensions and fact relationships. A dimension is an abstract concept for grouping data that share a common semantic meaning within the domain being modeled. A dimension is composed of a level or one or more hierarchies. The Store dimension in Figure 2 includes two hierarchies representing the administrative division and organizational structure.

Levels, such as a Product level in Figure 2, correspond to entity types in the ER model. Every instance of a level is called member. Levels contain one or several key attributes (underlined in Figure 2) identifying uniquely the members of

Figure 1. A Store dimension represented as (a) star and (b) snowflake schemas



a level and used for aggregation purposes. Levels may also have other descriptive attributes.

Hierarchies are required for establishing meaningful paths for roll-up and drill-down operations. A hierarchy contains several related levels: Product, Category, and Department levels in Figure 2. Hierarchies express different structures according to an analysis criterion (e.g., Product groups).

Given two consecutive levels of a hierarchy, the lower level is called child and the higher-level parent (e.g., the Product and Category levels in the Product groups hierarchy are, respectively, child and parent levels). A level of a hierarchy that does not have a child level is called leaf **(**e.g., the Product level), while a level that does not have a parent level is called root **(**e.g., the Department level).

The child-parent relationships are characterized by cardinalities. They indicate the minimum and maximum numbers of members in one level that can be related to a member in another level. In Figure 2, the cardinality between the Product and Category levels is many-to-many, indicating that a product can belong to many categories and a category can include many products. Various notations are used for representing cardinalities: ⟨ indicates (1,n), ○⟨ implies (0,n), — means (1,1), and ○ represents (0,1).

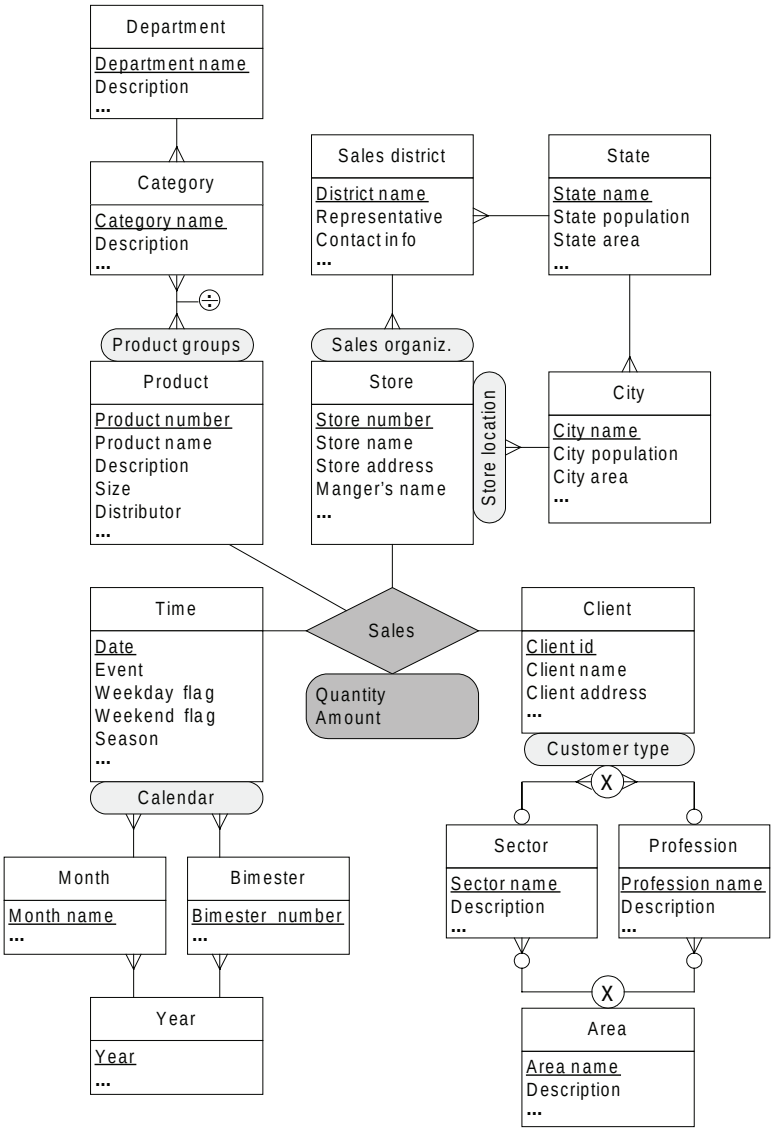
A fact relationship represents an n-ary relationship between leaf levels, and it may contain attributes commonly called measures. The schema in Figure 2 contains a Sales fact relationship with the measures Quantity and Amount.

Different Kinds of Hierarchies: Their Classification and Conceptual Representations

Considering the difference at the schema and instance levels, we classify hierarchies as shown in Figure 3 (Malinowski & Zimányi, 2008). The distinction at the schema level allows us to establish aggregation paths for roll-up and drill-down operations. The distinction at the instance level is important for the development of aggregation procedures since they depend on whether the instances form a tree (balanced or unbalanced), use some exclusive paths, or are represented as an acyclic graph.

Simple hierarchies. The relationship between their members is represented as a tree. Simple hierarchies can be further specialized in balanced, unbalanced, and noncovering:

Figure 2. A conceptual multidimensional schema of a Sales DW



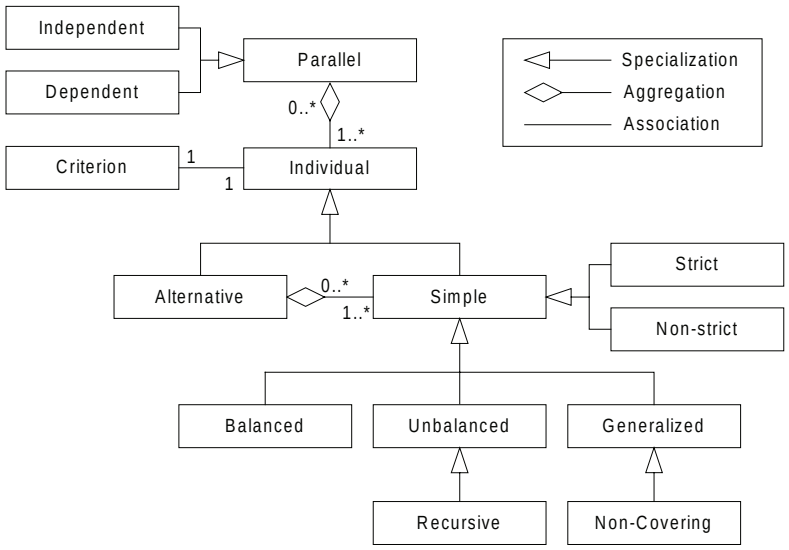
Balanced

- Schema: only one path.
- Members: they form a balanced tree since all parent members must have at least one child member, and a child member belongs to only one parent member.
- Example: the Store location hierarchy in Figure 2 comprising Store, City, and State levels.

Unbalanced

- Schema: only one path.
- Members: they form unbalanced trees since some parent members may not have associated child members.
- Example: not included in Figure 2; however, they are present in many DW applications. For example, a bank may include a hierar-

Figure 3. Hierarchy classification



chy composed by ATM, Agency, and Branch levels. However, some agencies do not have ATMs, and small branches do not have organizational divisions.

- Special case: recursive hierarchies, where the same level is linked by two or more roles forming a cyclic child-parent relationship such as an employee-supervisor relationship.

Generalized

- Schema: multiple exclusive paths sharing some levels. All these paths represent one hierarchy and account for the same analysis criterion.
- Members: each member of the hierarchy belongs to only one path. We use the symbol (X) for indicating that for every member, the paths are exclusive.
- Example: the Customer type hierarchy in Figure 2 where a customer can be a person or a company having common attributes belonging to Customer and Area levels. However, the buying behavior of a customer can be analyzed according to the specific level Profession for a person type and Sector for a company type.
- Special case: noncovering hierarchies that are generalized hierarchies with the additional restrictions that at the schema level, the leaf level is the same for all paths and the alternative paths are obtained by skipping one or several intermediate levels.

Nonstrict Hierarchies

- Schema: one path with at least one many-to-many child-parent relationship.
- Members: they form an acyclic graph since a child member may have more than one parent member. A distributing factor symbol ((÷)) may be included to indicate how the measures should be assigned to several parents.

- Example: the Product groups hierarchy in Figure 2 allows designers to model the situation when mobile phones can be classified in different products categories (e.g., phone, PDA, MP3 player).

Alternative Hierarchies

- Schema: aggregations of individual hierarchies accounting for the same analysis criterion. In such hierarchies, it is not semantically correct to simultaneously traverse the different composing hierarchies (i.e., the user must choose one of the alternative hierarchies for analysis).
- Members: they form a graph since a child member can be associated with more than one parent member belonging to different levels.
- Example: the Calendar hierarchy in the Time dimension in Figure 2 with two nonexclusive balanced hierarchies sharing Time and Year levels. During analysis, the user will choose a hierarchy composed by either Time, Month and Year, or Time, Bimester, and Year levels.

Parallel hierarchies. Several hierarchies accounting for different analysis criteria associated with the same dimension**.** These hierarchies may be:

- **Independent**: the different hierarchies do not share levels.
- **Dependent***:* the different hierarchies do share some levels, such as a State level between the Store location and Sales organization hierarchies in the Store dimension.

Table 1 compares multidimensional models that, to our knowledge, cope with hierarchies. We did not include some models that represent only balanced and parallel hierarchies (Torlone, 2003; Golfarelli & Rizzi, 1998; Sapia, Blaschka, Höfling & Dinter, 1998). We use three symbols: — when no reference to the hierarchy exists; ± when only a description and/or definition of the hierarchy are presented; and ✓ when a description and a graphical representation are given. If a different name for a hierarchy exists, we include it in the table.

As we can see from Table 1, some models give only a description and/or definition of some of the hierarchies, without a graphic representation. Further, for the same type of hierarchy, different names are used in the different models. None of the models takes into account different analysis criteria applied for hierarchies; consequently, the alternative and parallel hierarchies cannot be distinguished. In several cases, the hierarchy definition is general, and it may represent different kinds of hierarchies. This can be considered as an advantage, given the flexibility that allows the inclusion of other kinds of hierarchies in the proposed models. Nevertheless, this can also be an undesirable situation since users cannot express their requirements clearly; therefore, application developers cannot implement them.

Mapping Rules to Relational Model

Since the MultiDim model is based on the entity-relationship (ER) model, its mapping to the relational model uses well-known rules. We will refer only to mapping rules required for representing hierarchies:

- A level corresponds to an entity type in the ER model. It maps to a table containing all attributes of the level and includes an additional attribute for a primary key.
- A child-parent relationship corresponds to a binary relationship type in the ER model. In order to have meaningful hierarchies, the maximum cardinality of the parent role must be n. Thus, two mappings exist, depending on the cardinality of the child role:
  - If the cardinality of the child role is (0,1) or (1,1) (i.e., the child-parent relationship is many-to-one), the table corresponding to the child level is ex-

Table 1. Comparison of conceptual models for inclusion of different kinds of hierarchies

Model/ Hierarchy	Balanced	Unbalanced	Recursive	Generalized	Noncovering	Nonstrict	Alternative or Parallel
Pedersen, Jensen, and Dyreson (2001)	Onto ±	Non-onto ±	—	—	Noncovering ±	Nonstrict ±	Multiple ±
Hüsemann, Lechtenbörger, and Vossen (2000)	Simple ✓	—	—	Multiple Optional ✓	—	—	Multiple Alternative ✓
Tryfona, Busborg, and Borch (1999)	✓	✓	—	—	—	Nonstrict ✓	Multiple ✓
Hurtado and Gutierrez (2007)	Homogenous ±	Homogenous ±	—	Heterogeneous ±	Heterogeneous ±	—	Heterogeneous ±
Pourabbas and Rafanelli (2003)	Total classification ±	Partial classification ±	—	Multiple ±	Multiple ±	—	Multiple Multiplicity ±
Bauer, Hümmer, and Lehner (2000)	—	—	—	±	—	—	—
Abelló, Samos, and Saltor (2006)	✓	Non-onto ±	—	—	±	±	✓
Tsois, Karayannidis, and Sellis (2001)	✓	±	—	—	✓	±	✓
Rizzi (2007)	✓	±	✓	—	Ragged ✓	Multiple arc ✓	Convergence ✓
Luján-Mora, Trujillo, and Song (2006)	✓	—	—	—	—	✓	✓
OMG (2002)	Level-based ±	Value-based ±	Value-based ±	—	Value-based ±	—	—

tended with the foreign key referencing the corresponding parent level.

- If the cardinality of the child role is (0,n) or (1,n) (i.e., the child-parent relationship is many-to-many), a new table containing as attributes the foreign keys referencing the child and parent levels is created.

Logical Representation of Different Kinds of Hierarchies

The result of applying the previous specified rules to balanced hierarchies gives the snowflake schema. For example, the relational schema in Figure 1(b) represents the Store location hierarchy from Figure 2. Notice that denormalizing the tables produces a star schema as shown in Figure 1(a).

The usual practice for nonbalanced hierarchies is to transform them into balanced ones by including placeholders in missing child members. Then, the mapping for balanced hierarchies is applied. A different approach is taken for recursive hierarchies for which applying mapping rule gives as a result the so-called parent-child table.

The mapping of the Customer type generalized hierarchy in Figure 2 is shown in Figure 4(a) (ignoring for now the dotted line). Even though this mapping clearly represents the hierarchical structure, it does not allow users to only traverse the common hierarchy levels. Therefore, for that purpose, we include additional links between common levels. For example, we include a foreign key in the Customer table referencing the Area table (dotted line in the figure). This structure allows choosing different alternatives for analysis, either to use paths with specific levels (i.e., Sector and Profession), or to traverse the common levels for all members (i.e., Client and Area). Since a noncovering hierarchy is a special case of a generalized hierarchy, the same mapping rules can be applied. However, a usual solution is to transform the noncovering hierarchies into balanced ones by including placeholders or null values in missing levels. Then, the mapping to the snowflake or star schemas is used.

The traditional mapping of nonstrict hierarchies to relational model creates tables for representing levels and an additional table for representing a many-to-many relationship between them. The latter table is called a *bridge table*. When a distributing factor is included in the conceptual schema, an additional attribute is added to the bridge table to store this information.

For alternative and parallel hierarchies, the traditional mapping to relational tables can be applied; it is shown in Figure 4(b) and Figure 4(c) for the Calendar hierarchy and for both hierarchies of the Store dimension from Figure 2.

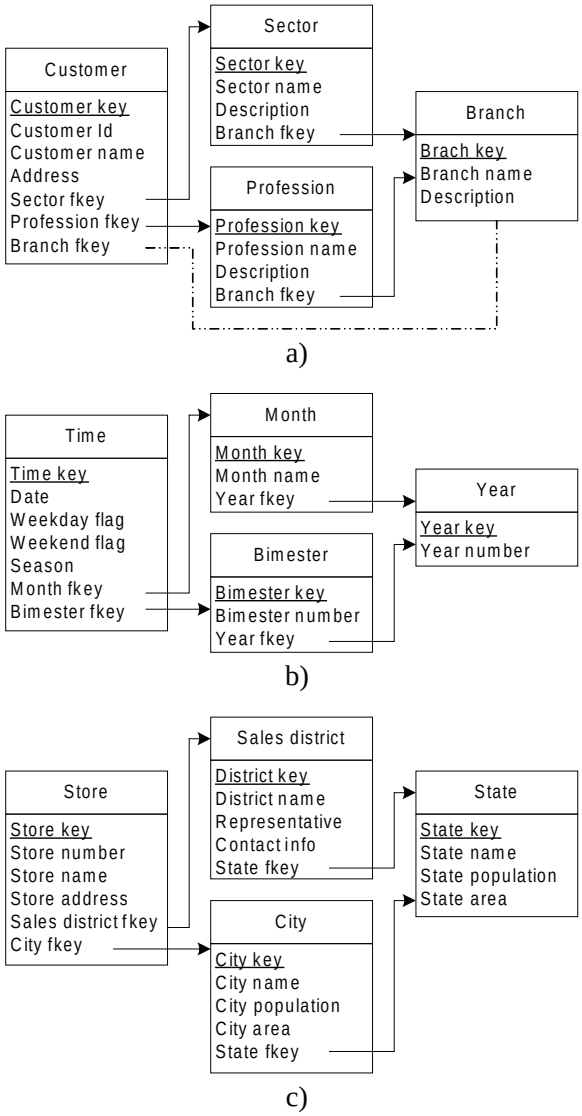
Notice that even though generalized and alternative hierarchies can be easily distinguished at the conceptual level (the Customer type and Calendar hierarchies in Figure 2, respectively), this distinction cannot be easily done at the logical level (Figure 4(a) and Figure 4(b)). A similar situation occurs in distinguishing alternative and parallel hierarchies (Figure 4(b) and Figure 4(c)), even though both kinds of hierarchies represent different situations. In the alternative hierarchies, the user cannot combine the levels from different composing hierarchies, but they can combine them for parallel hierarchies; for example, to analyze sales made in different cities belonging to the specific sales district.

Several commercial DBMS products (e.g., from Microsoft, Oracle, IBM) make it possible to implement some hierarchies, including unbalanced, noncovering, recursive, and parallel. Additionally, Microsoft Analysis Services 2005 allows the definition of a bridge table for dealing with nonstrict hierarchies.

FUTURE TRENDS

Even though several conceptual multidimensional models were proposed, the usual practice is to use logical models. Therefore, the research com-

Figure 4. Relational schemas for the (a) client, (b) time, and (c) store dimensions from Figure 2



munity should demonstrate the benefits of using conceptual models for DW and OLAP applications and provide a commonly accepted formalism to support interoperability and standardization (Torlone, 2003).

Furthermore, the purpose of having hierarchies is to perform aggregation of measures while traversing them. Since not all hierarchies are considered in commercial tools and research proposals, there is still the necessity to develop aggregation procedures for all kinds of hierarchies.

CONCLUSION

In this chapter, we discussed the need to have conceptual multidimensional models able to express requirements for DW and OLAP applications, including different kinds of hierarchies existing in real-world situations. We presented a classification of hierarchies and showed how they can be represented at both the conceptual and

logical levels using, respectively, the MultiDim and relational models.

The conceptual representations of hierarchies permits a clear distinction of each type of hierarchy at the schema and instance levels, preserving at the same time the general characteristics of the snowflake schema (i.e., aggregation paths). Nevertheless, this distinction cannot be done at the logical level (e.g., relational) for generalized, alternative, and parallel hierarchies. Therefore, by using a conceptual model independent from implementation details, users will be able to better understand the data to be analyzed, and developers can have a common vision of different kinds of hierarchies and focus on aspects related to their implementation.

REFERENCES

Abelló, A., Samos, J., & Saltor, F. (2006). YAM^2: A multidimensional conceptual model extending UML. *Information Systems, 32*(6), 541–567.

Bauer, A., Hümmer, W., & Lehner, W. (2000). *An alternative relational OLAP modeling approach*. Proceedings of the 2nd International Conference on Data Warehousing and Knowledge Discovery, 189–198.

Golfarelli, M., & Rizzi, S. (1998). *A methodological framework for data warehouse design*. Proceedings of the 1st ACM International Workshop on Data Warehousing and OLAP, 3–9.

Hurtado, C., & Gutierrez, C. (2007). Handling structural heterogeneity in OLAP. In R. Wrembel, & C. Koncilia (Eds.), *Data warehouses and OLAP: Concepts, architectures and solutions* (pp. 27–57). Hershey, PA: Idea Group Publishing.

Hüsemann, B., Lechtenbörger, J., & Vossen, G. (2000). *Conceptual data warehouse design*. Proceedings of the 2nd International Workshop on Design and Management of Data Warehouses, 6.

Luján-Mora, S., Trujillo, J., & Song, I. (2006). A UML profile for multidimensional modeling in data warehouses. *Data & Knowledge Engineering, 59*(3), 725–769.

Malinowski, E., & Zimányi. E. (2008). *Advanced data warehouse design: From conventional to spatial and temporal applications*. Springer

Object Management Group. (2002). Common warehouse metamodel. Retrieved from http://www.omg.org/docs/formal/03-03-02.pdf

Pedersen, T., Jensen, C.S., & Dyreson, C. (2001). A foundation for capturing and querying complex multidimensional data. *Information Systems, 26*(5), 383–423.

Pourabbas, E., & Rafanelli, M. (2003). Hierarchies. In M. Rafanelli. (Ed.), *Multidimensional databases: Problems and solutions* (pp. 91–115). Hershey, PA: Idea Group Publishing.

Rizzi, S. (2007). Conceptual modeling solutions for the data warehouse. In R. Wrembel, & C. Koncilia (Eds.), *Data warehouses and OLAP: Concepts, architectures and solutions* (pp. 1–26). Hershey, PA: Idea Group Publishing.

Sapia, C., Blaschka, M., Höfling, G., & Dinter, B. (1998). *Extending the E/R model for multidimensional paradigms*. Proceedings of the 17th International Conference on Conceptual Modeling, 105–116.

Torlone, R. (2003). Conceptual multidimensional models. In M. Rafanelli (Ed.), *Multidimensional databases: Problems and solutions* (pp. 91–115). Hershey, PA: Idea Group Publishing.

Tryfona, N., Busborg, F., & Borch, J. (1999). *StarER: A conceptual model for data warehouse design*. Proceedings of the 2nd ACM International Workshop on Data Warehousing and OLAP, 3–8.

Tsois, A., Karayannidis, N., & Sellis, T. (2001). *MAC: Conceptual data modeling for OLAP*.

Proceedings of the 3rd International Workshop on Design and Management of Data Warehouses, 5.

KEY TERMS

Conceptual Multidimensional Model: A set of objects and rules that facilitates an abstract representation of requirements for DW and OLAP applications. It usually includes dimensions with hierarchies and facts with associated measures.

Dimension: An abstract concept for grouping data that shares a common semantic meaning within the domain being modeled.

Hierarchy: A sequence of levels providing data at different granularities for establishing meaningful aggregation paths.

Level: A set of elements representing the same data granularity.

Logical Representation (or Schema): A specification of data structures according to the features of the given logical model, such as relational or object-relational.

Snowflake Schema: A variation of the star schema except that dimensions are normalized representing each hierarchy level in a separate table.

Star Schema: A relational schema consisting of a fact table, which links to other de-normalized tables called dimension tables.

ENDNOTE

[1] A more detailed description of proposals for multidimensional modeling can be found in Torlone (2003).

Chapter VII
Spatial Data in Multidimensional Conceptual Models

Elzbieta Malinowski
Universidad de Costa Rica, Costa Rica

INTRODUCTION

Data warehouses (DWs) are used for storing and analyzing high volumes of historical data. The structure of DWs is usually represented as a star schema consisting of fact and dimension tables. A fact table contains numeric data called measures (e.g., quantity). Dimensions are used for exploring measures from different analysis perspectives (e.g., according to products). They usually contain hierarchies required for online analysis processing (OLAP) systems in order to dynamically manipulate DW data. While traversing hierarchy, two operations can be executed: the roll-up operation, which transforms detailed measures into aggregated data (e.g., daily into monthly sales); and the drill-down operation, which does the opposite.

Current DWs typically include a location dimension (e.g., store or client address). This dimension is usually represented in an alphanumeric format. However, the advantages of using spatial data in the analysis process are well known, since visualizing data in space allows users to reveal patterns that are otherwise difficult to discover. Therefore, spatial databases (SDBs) can give in-

sights about how to represent and manage spatial data in DWs.

SDBs provide mechanisms for storing and manipulating spatial objects. These databases typically are used for daily business operations (e.g., to indicate how to get to a specific place from the current position given by a GPS). SDBs are not well suited for supporting the decision-making process (Bédard, Rivest & Proulx, 2007) (e.g., to find the best location for a new store). This is how the field of spatial data warehouses (SDWs) emerged.

SDWs combine SDB and DW technologies for managing significant amounts of historical data that include spatial location. To better represent users' requirements for SDW applications, a conceptual model should be used. The advantages of using conceptual models are well known in database design. Nevertheless, the lack of a conceptual approach for DW and OLAP system modeling in addition to the absence of a commonly accepted conceptual model for spatial applications make the modeling task difficult. Existing conceptual models for SDBs are not adequate for DWs since they do not include the concepts of dimensions, hierarchies, and measures. Therefore, there is a need for extending multidimensional models by including spatial data to help users have a better understanding of the data to be analyzed.

BACKGROUND

To the best of our knowledge, very few proposals address the issue of conceptual modeling for SDWs (Ahmed & Miquel, 2005; Bimonte, Tchounikine & Miquel, 2005; Jensen, Klygis, Pedersen & Timko, 2004; Pestana, Mira da Silva & Bédard, 2005). Some of these models include the concepts presented in Malinowski and Zimányi (2004), as explained in the next section; other models extend nonspatial multidimensional models with different aspects such as imprecision in location-based data (Jensen et al., 2004) or continuous phenomena (e.g., temperature or elevation) (Ahmed & Miquel, 2005).

Other models for SDWs use the logical relational representation based on the star/snowflake schemas. These proposals introduce concepts of spatial dimensions and spatial measures (Fidalgo, Times, Silva & Souza, 2004; Rivest, Bédard & Marchand, 2001; Stefanovic, Han & Koperski, 2000); however, they impose some restrictions on the model, as discussed in the next section.

We consider that a conceptual multidimensional model with spatial support should not only include dimensions, hierarchies, and measures, but should also refer to various aspects that are not present in conventional multidimensional models related to particularities of spatial objects.

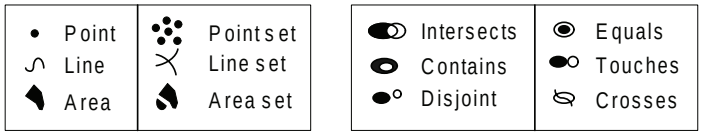
Spatial objects correspond to real-world entities for which the application needs to keep their spatial characteristics. Spatial objects consist of a thematic (or descriptive) component and a spatial component. The thematic component describes general characteristics of spatial objects (e.g., name) and is represented using traditional DBMS data types (e.g., integer, string, date). The spatial component includes its geometry that can be of type point, line, surface, or a collection of them.

Spatial objects can relate to each other forming topological relationships. Various topological relationships have been defined (Egenhofer, 1993). They allow determining, for example, whether a store is located within city limits or whether bus and tramway lines intersect in some location.

Pictograms are typically used for representing spatial objects and topological relationships in conceptual models, such as the ones shown in Figure 1 (Parent, Spaccapietra & Zimányi, 2006).

Even though the experience gained in SDBs can be useful for SDWs, the inclusion of spatial objects in a multidimensional model requires additional analysis with respect to topological relationships existing between various elements of the multidimensional model or aggregations of spatial measures, among others. While some of

Figure 1. Pictograms for (a) spatial data types and (b) topological relationships



(a) *(b)*

these aspects are mentioned briefly in the existing literature (e.g., spatial aggregations) (Pedersen & Tryfona, 2001.), others are neglected (e.g., the influence on aggregation procedures of the topological relationships between spatial objects forming hierarchies).

MAIN FOCUS

The Spatially-Extended MultiDim Model

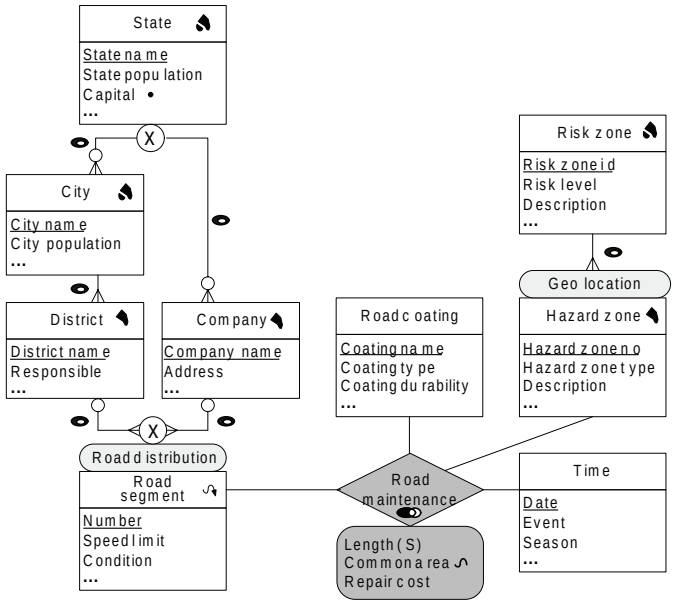
The MultiDim model is a conceptual multidimensional model (Malinowski, 2009) able to represent fact relationships, measures, dimensions, and hierarchies. This model was extended by providing spatial support to various elements (Malinowski & Zimányi, 2008).

To describe our model, we use an example on the analysis of road maintenance costs[1]. Roads are located in zones that can be affected by various types of hazard events such as avalanche, landslide, flood, and so forth. For every event type that takes place in the same area, the corresponding area is defined as a hazard zone. Then, the hazard zones of the same or different type are grouped according to risk level. The risk level is established based on the importance and frequency of the hazard, and may be classified as high, medium, low, or irrelevant. Therefore, a new geographical distribution is elaborated considering the risk level. When a natural hazard event occurs and affects the road structure, the road must be repaired. Roads are divided into segments that may belong to either city roads or highways. Districts are responsible for city road maintenance, while maintenance to highways is provided by private entities. In addition, the repair cost should be delivered for each state in order to support the formulation of cost allocation policies. The analysis should also help to reveal how the various types of road coatings affect maintenance costs. The multidimensional schema that represents these requirements is shown in Figure 2. It contains dimensions, hierarchies, a fact relationship, and measures.

A dimension is an abstract concept for grouping data that share a common semantic meaning within the domain being modeled. It represents either a level or one or more hierarchies.

A level corresponds to an entity type in the ER model and represents a set of instances called members. For example, Road coating in Figure 2 is a one-level dimension. Spatial levels are levels for which the application needs to keep their spatial characteristics. This is captured by its geometry, which in our model is represented using the pictograms from Figure 1(a). In Figure 2, we have several spatial levels, such as Road segment, District, Company, and so forth. Notice that a level may have spatial attributes whether it is spatial or not; for example, in Figure 2, the spatial level State contains a spatial attribute Capital.

Figure 2. An example of a multidimensional schema with spatial elements



Hierarchies are required for establishing meaningful paths for the roll-up and drill-down operations. Since hierarchies can express various structures according to an analysis criterion, we use the criterion name to differentiate them (e.g., Geo location in Figure 2).

Hierarchies contain several related levels. Depending on whether the levels include more detailed or more general data, they are called, respectively, child and parent. In Figure 2, Hazard zone is a child level, while Risk zone is a parent level. A level of a hierarchy that does not have a child level is called leaf (e.g., Road segment), while a level that does not have a parent level is called root (e.g., State). Levels contain one or several key attributes (underlined in Figure 2) and may also have other descriptive attributes. Key attributes indicate how child members are grouped into parent members during the roll-up operation (e.g., cities will be grouped according to the state name to which they belong.

The relationships between child and parent levels are characterized by cardinalities. They indicate the minimum and maximum numbers of members in one level that can be related to a member in another level. Different notations are used to represent cardinalities: —< (1,n), —o< (0,n), —— (1,1), and —o (0,1). In Figure 2, the cardinality between the Hazard zone and the Risk zone levels is many-to-one, indicating that a hazard zone may only belong to one risk zone and that a risk zone may include many hazard zones. Different cardinalities may exist between levels leading to different types of hierarchies (Malinowski, 2009). It is important to include them in the conceptual model since they exist in real-world situations and may be required by decision-making users. For example, the Road distribution hierarchy is the so-called generalized hierarchy; at the schema level, it is comprised of different paths but, at the instance level, each member belongs to only one path. The symbol

⊗ indicates that for every member, the paths are exclusive.

If a hierarchy includes at least one spatial level, it is called a spatial hierarchy. In a spatial hierarchy, two consecutive spatial levels are topologically related. This is represented using the pictograms of Figure 1(b). By default, we suppose the *within* topological relationship, which indicates that the geometry of a child member is included in the geometry of a parent member; for example, in Figure 2, the geometry of each hazard zone is included in the geometry of its corresponding risk zone. However, in real-world situations, different topological relationships may exist among spatial levels. They determine the complexity of the procedures for aggregation of measures during the roll-up operations (Malinowski & Zimányi, 2008). For example, in a spatial hierarchy formed by the Store and City levels, some points referring to store locations may be on the border between two cities represented as surfaces. In this situation, when traversing from the Store to the City levels, it is necessary to determine whether the measure (e.g., required taxes) should be distributed between two cities or considered only for one of them.

A fact relationship (e.g., Road maintenance in Figure 2) represents an n-ary relationship between leaf levels. This fact relationship can be spatial if at least two leaf levels are spatial (e.g., Road segment and Hazard zone). A spatial fact relationship may require the inclusion of a spatial predicate for the spatial join operation. In the figure, an intersection topological relationship indicates that users require focusing their analysis on those road segments that intersect hazard zones. If this topological relationship is not included, users are interested in any topological relationships that may exist.

A (spatial) fact relationship may include thematic or spatial measures. The former are usually numeric values used for quantitative analysis (e.g., to analyze the changes in repair costs during various periods of time). Spatial measures can be represented by a geometry or calculated using spatial operators (e.g., distance, area). To indicate that the measure is calculated using spatial operators, we use the symbol (S). The schema in Figure 2 contains two spatial measures: Length and Common area. Length is a number representing the length of a road segment that belongs to a hazard zone, and Common area represents the geometry of this common part.

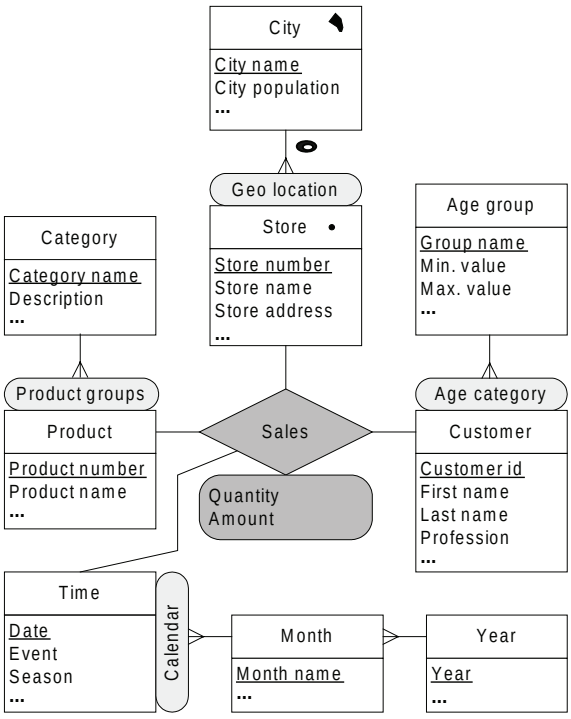
Measures require the specification of the function used for aggregations along the hierarchies. By default, we suppose *sum* for the measures represented as numbers and *spatial union* for the measures represented as geometries.

The schema in Figure 2 provides a multidimensional view of data with clearly distinguished elements: the fact relationship indicating the focus of analysis, measures used for aggregation purposes, and dimensions with hierarchies allowing users to analyze measures from various perspectives and with different levels of detail. Further, our model can include spatial as well as nonspatial elements in an orthogonal way; therefore, it gives users various alternatives for choosing the kind of data that better fit their analysis requirements.

Modeling Aspects

A common practice in modeling spatial data warehouses is to convert the location dimension into a spatial dimension by including spatial representation of hierarchy levels. Figure 3 shows an example for analysis of sales where, instead of relying on alphanumeric representation for the Store and City locations, we refer to their geographic locations. Therefore, users are able to analyze sales figures considering the territorial distribution of stores and cities using the roll-up or drill-down operations. For example, they can compare sales in various neighboring cities or sales in various cities according to different age groups. This may help to identify stores that are not frequently visited by certain groups of customers.

Figure 3. Spatially-enhanced model of a sales DW



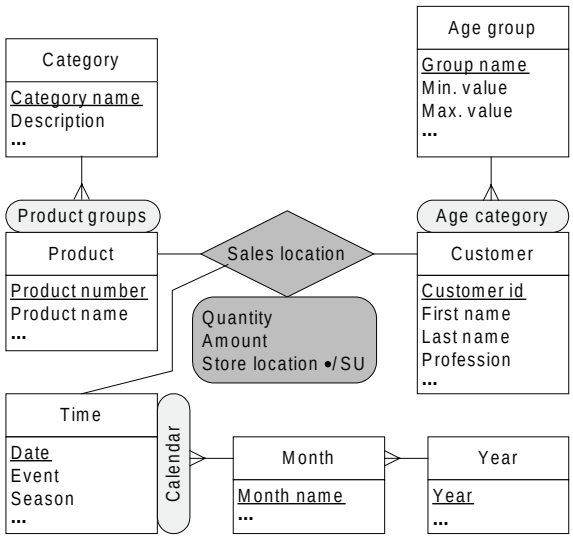
An alternative schema for analysis of sales is shown in Figure 4 with a store location represented now as a spatial measure. Since the dimensions include hierarchies, a spatial aggregation function (*spatial union* [SU]) is defined. When a user rolls up, for example, to the Age group level, the store locations corresponding to certain age groups will be aggregated and represented as a set of points. Other spatial operators can also be used; for instance, *center of n points*. This may help to discover patterns such as preferred store locations of clients belonging to a specific age group.

Even though both schemas are similar, different analyses can be undertaken when a location is handled as a spatial hierarchy or a spatial measure. For example, comparison of amount of sales in various geographic zones can be done for the schema in Figure 3 but not for the schema in Figure 4, since only the exact locations of stores are given without their geographic distribution. On the other hand, aggregation of store locations according to some predicate involving time, product, and/or customer can be easily done for the schema in Figure 4, whereas it cannot be achieved for the schema in Figure 3 since all dimensions are independent and traversing a hierarchy and one of them does not aggregate data in another hierarchy. Therefore, the designer of the application must determine which of these schemas better represents user needs.

Other Approaches

The MultiDim model extends those of Stefanovic, et al. (2000) and Rivest, Bédard, Proulx, Nadeau, Hubert, and Pastor (2005), particularly by allowing a nonspatial level (e.g., address represented as an alphanumeric data type) to roll up to a spatial level

Figure 4. Another variant of conceptual schema from Figure 3



(e.g., city represented by a surface). Further, we extend the classification for spatial dimensions, including a spatial dimension, even if it only has one spatial level (e.g., a State dimension that is spatial without any other geographical division). The extension also includes a classification of various kinds of spatial hierarchies existing in real-world situations and a classification of topological relationships according to complexity of aggregation procedures; these aspects are currently ignored in SDW research.

With respect to spatial measures, we based our proposal on Stefanovic, et al. (2000) and Rivest, et al. (2001). However, both authors refer to implementation details stating that spatial measures should be represented using pointers to spatial objects. In our model, we clearly separate the conceptual and implementation aspects. In addition, Rivest, et al. (2001) require the presence of spatial dimensions for including spatial measures represented by a geometry. On the contrary, in our model, a spatial measure can be related to nonspatial dimensions, as can be seen in Figure 4.

Fidalgo, et al. (2004) do not allow spatial measures and convert them in spatial dimensions. However, the resulting model corresponds to different analysis criteria and answers to different queries, as already discussed in the previous section.

FUTURE TRENDS

An interesting research problem is the inclusion of spatial data represented as a continuous (or field) view (e.g., temperature, altitude, soil cover). Although some proposals already exist (Ahmed & Miquel, 2005), additional analysis is still required for spatial hierarchies formed by levels represented by field data, spatial measures representing continuous phenomena and their aggregations, fact relationships that include spatial dimensions with field data, among others.

Another issue is to cope with multiple representations of spatial data (i.e., allowing the same real-world object to have different geometries). Multiple representations are a common practice

in SDBs. They are also an important aspect in the context of DWs since spatial data may be integrated from various source systems that use different geometries for the same spatial object.

The MultiDim model allows the inclusion of two-dimensional (2D) objects. Nevertheless, many application domains (e.g., urban planning, disaster management) require three-dimensional (3D) objects. The extension to manage 3D objects should address different issues (e.g., different topological relationships between hierarchy levels, aggregation of 3D measures, etc.).

CONCLUSION

In this chapter, we referred to SDWs and presented various elements of a spatial multidimensional model, such as spatial levels, spatial hierarchies, spatial fact relationships, and spatial measures.

The inclusion of spatial objects in the conceptual multidimensional model aims at improving the data analysis and design for SDW and spatial OLAP applications. Since our model is platform-independent, it reduces the difficulties of modeling spatial applications. This is an important feature because decision-making users do not usually possess the expertise required by the software currently used for managing spatial data. Furthermore, spatial OLAP tools developers can have a common vision of the various features that comprise a spatial multidimensional model and of the different roles that each element of this model plays. This can help to develop accurate and efficient solutions for spatial data manipulations.

REFERENCES

Ahmed, T., & Miquel, M. (2005). *Multidimensional structures dedicated to continuous spatio-temporal phenomena*. Proceedings of the 22nd British National Conference on Databases, 29–40.

Bédard, Y., Rivest, S., & Proulx, M. (2007). Spatial on-line analytical processing (SOLAP): Concepts, architectures and solutions from a geomatics engineering perspective. In R. Wrembel, & C. Koncilia (Eds.), *Data warehouses and OLAP: Concepts, architectures and solutions* (pp, 298–319). Hershey, PA: Idea Group Publishing.

Bimonte, S., Tchounikine, A., & Miquel, M. (2005). *Towards a spatial multidimensional model*. Proceedings of the 8th ACM International Workshop on Data Warehousing and OLAP, 39–46.

Egenhofer, M. (1993). A model for detailed binary topological relationships. *Geomatica*, *47*(3-4), 261–273.

Fidalgo, R., Times, V., Silva, J., & Souza, F. (2004). *GeoDWFrame: A framework for guiding the design of geographical dimensional schemes.* Proceedings of the 6th International Conference on Data Warehousing and Knowledge Discovery, 26–37.

Jensen, C.S., Klygis, A., Pedersen, T., & Timko, I. (2004). Multidimensional data modeling for location-based services. *VLDB Journal*, *13*(1), 1–21.

Malinowski, E. (2009). Different kinds of hierarchies in multidimensional models.

Malinowski, E., & Zimányi, E. (2004). *Representing spatiality in a conceptual multidimensional model*. Proceedings of the 12th ACM Symposium on Advances in Geographic Information Systems, 12–21.

Malinowski, E., & Zimányi. E. (2008). *Advanced data warehouse design: from conventional to spatial and temporal applications.* Springer.

Parent, C., Spaccapietra, S., & Zimányi, E. (2006). *Conceptual modeling for traditional and spatio-temporal applications: The MADS approach.* Springer.

Pedersen, T.B., & Tryfona, N. (2001). *Pre-aggregation in spatial data warehouses.* Proceedings of the 7th International Symposium on Advances in Spatial and Temporal Databases, 460–478.

Pestana, G., Mira da Silva, M., & Bédard, Y. (2005). *Spatial OLAP modeling: An overview base on spatial objects changing over time.* Proceedings of the IEEE 3rd International Conference on Computational Cybernetics, 149–154.

Rivest, S., Bédard, Y., & Marchand, P. (2001). Toward better support for spatial decision making: Defining the characteristics of spatial on-line analytical processing (SOLAP). *Geomatica, 55*(4), 539–555.

Rivest, S., Bédard, Y., Proulx, M.J., Nadeau, M., Hubert, F., & Pastor, J. (2005). SOLAP technology: Merging business intelligence with geospatial technology for interactive spatio-temporal exploration and analysis of data. *ISPRS Journal of Photogrammetry & Remote Sensing, 60,* 17–33.

Stefanovic, N., Han, J., & Koperski, K. (2000). Object-based selective materialization for efficient implementation of spatial data cubes. *Transactions on Knowledge and Data Engineering, 12*(6), 938–958.

KEY TERMS

Multidimensional Model: A model for representing the information requirements for data warehouse and OLAP applications. It includes facts, measures, dimensions, and hierarchies.

Spatial Data Warehouse: A data warehouse that includes spatial dimensions, spatial measures, or both, thus allowing spatial analysis.

Spatial Dimension: An abstract concept for grouping data that share common semantics within the domain being modeled. It contains a spatial level or one or more spatial hierarchies.

Spatial Fact Relationship: An n-ary relationship between two or more spatial levels belonging to different spatial dimensions.

Spatial Hierarchy: One or several related levels where at least one of them is spatial.

Spatial Level: A type defining a set of attributes, keeping track of the spatial extent of its instances (members).

Spatial Measure: An attribute of a (spatial) fact relationship that can be represented by a geometry or calculated using spatial operators.

ENDNOTE

[1] This example is inspired from Parent, Spaccapietra, and Zimányi (2006).

Chapter VIII
Requirement Specification and Conceptual Modeling for Data Warehouses

Elzbieta Malinowski
Universidad de Costa Rica, Costa Rica

INTRODUCTION

Data warehouses (DWs) integrate data from different source systems in order to provide historical information that supports the decision-making process. The design of a DW is a complex and costly task since the inclusion of different data items in a DW depends on both users' needs and data availability in source systems.

Currently, there is still a lack of a methodological framework that guides developers through the different stages of the DW design process. On the one hand, there are several proposals that informally describe the phases used for developing DWs based on the authors' experience in building such systems (Inmon, 2002; Kimball, Reeves, Ross, & Thornthwaite, 1998). On the other hand, the scientific community proposes a variety of approaches for developing DWs, discussed in the next section. Nevertheless, they either include features that are meant for the specific conceptual model used by the authors, or they are very complex. This situation has occurred since the need to build DW systems that fulfill user expectations was ahead of methodological and formal approaches for DW development, just like the one we had for operational databases.

Specifically, the approaches for requirements specifications and conceptual modeling differ significantly because some of them rely mainly on user requirements, while others take into con-

sideration the underlying operational databases instead of user needs. Such diversity of approaches may overwhelm designers who could find it difficult to identify the one approach that better fits to particularities of a DW project.

In this chapter we refer to requirements specification and conceptual modeling phases for DW design. Our proposal unifies the already existing approaches by giving an overall perspective of different alternatives available to designers when developing a DW.

BACKGROUND

The requirements specification phase is one of the earliest steps in system development and it has a major impact on the success of DW projects (Winter & Strauch, 2003). This phase will help to identify the essential elements of a multidimensional schema, i.e., facts with associated measures, dimensions, and hierarchies, required to facilitate future data manipulations and calculations. These elements should be clearly and concisely represented in a conceptual schema in a later stage. This schema will serve as basis for analysis tasks performed by the users and will be used by the designers during future evolutions of the DW.

There are different approaches for requirements specification and conceptual modeling of DWs. The so-called **user-driven approach**[1] takes into account the fact that users play a fundamental role during requirement analysis and must get actively involved in the elucidation of relevant facts and dimensions (Freitas, Laender, & Campos, 2002; Luján-Mora & Trujillo, 2003). The **business-driven approach**[2] bases derivation of DW structures on the analysis of either business requirements or business processes (Giorgini, Rizzi, & Garzetti, 2005; List, Schiefer, & Min Tjoa, 2000). On the other hand, the **source-driven approach**[3] analyzes the underlying source systems in order to obtain the DW schema (Böehnlein & Ulbrich-vom Ende, 1999; Cabibbo & Torlone, 1998; Golfarelli & Rizzi, 1998; Moody & Kortink, 2000). Finally, the c**ombined approach**[4] puts together the business- or user- driven and data-driven approaches representing what the business or user demands are and what the source systems can provide (Bonifati, Cattaneo, Ceri, Fuggetta, & Paraboschi, 2001; Winter & Strauch, 2003).

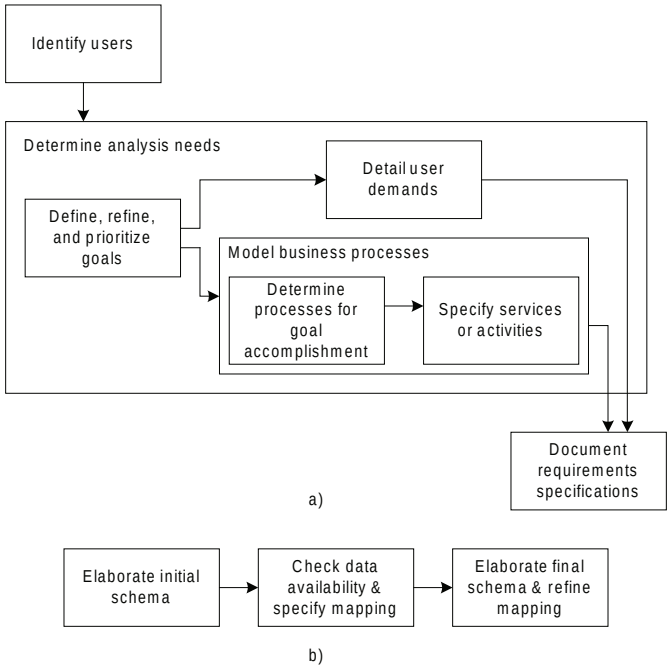
Nevertheless, the proposed approaches are difficult to use. On the one hand, different authors use techniques or models that require specific technical knowledge. On the other hand, the variety of existing approaches may overwhelm even experienced designers who tend to concentrate more on technical issues, e.g., physical modeling or query performance, and therefore, many DW projects skip the requirements specification and conceptual modeling phases.

Presenting the variety of approaches in a coherent whole may facilitate their understanding without loosing their semantic differences. It could also facilitate the developer's team to choose one approach that better fits the particular needs of a DW project. Furthermore, professionals who are highly skilled in the development of operational databases but inexperienced in the development of DWs could better understand the different aspects that must be considered during the DW design process.

MAIN FOCUS

In this section we present three different approaches for requirements specifications and conceptual modeling. Depending on whether users (business) or source systems are the driving force for requirements specifications, we propose, respectively, analysis-driven and source-driven approaches. We also present what we call the analysis/source-driven approach, which combines both previously-mentioned approaches. Although we separate the requirements specifications and the conceptual modeling phases for readability purposes, in reality these phases often overlap.

Figure 1. Analysis-driven approach: a) requirements specification and b) conceptual design phases



Analysis-Driven Approach

In the analysis-driven approach, the driving force for developing a conceptual schema is business or user requirements. These requirements express the organizational goals and needs that a DW is expected to address in order to support the decision-making process. Below we refer in more detail to the analysis-driven approach presenting the required steps for its realization.

Requirements Specification (Figure 1a)

- **Identify users**: Due to the fact that a DW provides an enterprise-wide decision-support infrastructure, users at different hierarchical levels of the organization must be included (List *et al.*, 2000). *Executive users* at the top organizational level may express their business needs that help to identify high-level objectives and goals and the overall business vision (Kimball et al., 1998). *Management users* may refer to a more specific area of the organization by providing more insights into the business processes or the tactics used for achieving business goals. Finally, *professional users* may be responsible for a specific section or services and may demand specific information related to their area of interest.
- **Determine analysis needs**: Determining analysis needs helps to understand what data should be available to respond to users' expectations for having a DW. Inasmuch as we focus on multidimensional modeling, this phase should identify facts with measures, dimensions with hierarchies, and the preliminary list of possible queries or analytical scenarios. The process of determining analysis needs is complex and includes several steps:
  - **Define, refine, and prioritize goals**: The starting point is the consideration of business goals in order to guide

user needs and convert them into data elements or to find critical business processes required for goal accomplishment. Since participating users belong to different management levels, analysis needs may be expressed by considering both general and specific goals. The latter should be aligned with the general ones to ensure a common direction of the overall development. The goal-gathering process may include interviews, facilitating sessions, or brainstorming (Giorgini *et al.,* 2005; Bonifati *et al.,* 2001; Kimball *et al.*, 1998). The list of goals should be analyzed taking into consideration their similarities or inconsistencies; this will help to synthesize them in a manageable number (Bonifati *et al.*, 2001). For every goal subsequent steps as shown in Figure 1a) are realized. We propose two different approaches based either on a more specific definition of user demands (upper part in the figure) or on the modeling of business processes (lower part in the figure).

- **Detail user demands**: Additional interviews with specific users focusing on more precise goal definition will be conduced. They allow designers to elicit the information needed for multidimensional schemas. Techniques other than interviews may also be used, e.g., workshops, questionnaires, or prototyping.
- **Model business processes**: Goal accomplishment is closely related to business processes. Therefore, the relevant business processes should be determined for every specific goal. Since a business process is a group of services or activities that together create a result of value for a customer or a market, the next step is the identification of these services or activities. Activities or services include data required for their realization, which may be included in future multidimensional schema. Business processes can be considered implicitly and informally (Kimball *et al.,* 1998) or a more formal business process model can be created (Böehnlein & Ulbrich-vom Ende, 2000).

- **Document requirements specifications**: The information obtained in the previous step should be documented. The delivered documentation is the starting point for the technical metadata (i.e., description of all DW structure and application components) and will form part of business metadata (i.e., description of all DW components using business terminology). This document is not a final specification of requirements since additional interactions may be necessary during the conceptual modeling phase in order to refine or clarify some aspects.

Conceptual Modeling (Figure 1b)

- **Elaborate the initial schema**: Well-specified business or user requirements lead to clearly-distinguishable multidimensional elements. Therefore, the first approximation of a conceptual schema can be developed. It is recommended to use a conceptual multidimensional model (e.g., as proposed by Malinowski, (2008)) to improve the communication with non-expert users. This schema should be validated against its potential usage in analytical processing by first revising the list of queries and analytical scenarios as well as by consulting the users directly. During this step, the refinement of the conceptual schema may require several iterations with the users.
- **Determine data availability and specify mappings**: The data contained in source

systems determines whether the proposed conceptual schema can be transformed into logical and physical schemas and fed with the data required for analysis. All elements included in the conceptual schema are verified against the data items in source systems. This process may be time consuming if the underlying source systems are not documented, are denormalized, or are legacy systems. The result of this step is the specification of mappings for all elements of a multidimensional schema that match with data in source systems. This specification also includes the description of required transformations between source and DW data, if necessary.

- **Elaborate final schema and refine mappings**: If data is available in source systems for all elements of the conceptual schema, the initial schema may be considered as a final schema. However, if not all multidimensional elements can be fed with data from source systems, a new iteration with users to modify their requirements according to data availability is required. As a result, a new schema that may require the modification of existing mappings is designed.

During all steps of the conceptual modeling phase the specification of business and technical metadata is in continuous development.

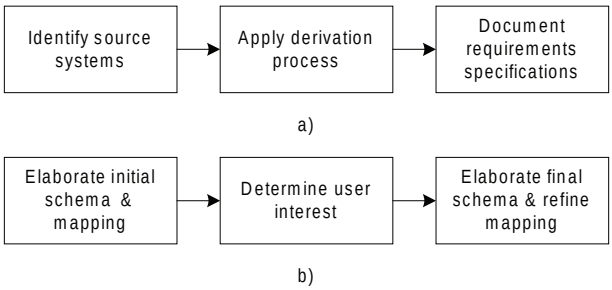
Source-Driven Approach

The source-driven approach relies on the data available in source systems. It aims at identifying all candidate multidimensional schemas that can be realistically implemented on the top of the available operational databases. We next describe the steps of the source-driven approach.

Requirements Specification (Figure 2a)

- **Identify source systems**: The objective of this step is to determine the existing operational systems that can serve as data providers for the DW. External sources are not considered in this stage; they can be included later on if additional information is required. This step relies on system documentation represented using preferably the ER model or relational tables. However, since in many situations this may be difficult to obtain, reverse engineering processes may be applied to rebuild the logical and/or conceptual schemas of the source systems.
- **Apply derivation process**: Different techniques can be used for deriving multidimensional elements from operational databases (Bonifati *et al.*, 2001; Böehnlein & Ulbrich-vom Ende, 1999; Cabibbo & Torlone, 2000; Golfarelli & Rizzi, 1998;

Figure 2. Source-driven approach: a) requirements specification and b) conceptual modeling phases



Moody & Kortink, 2000). All these techniques require that operational databases be represented using the ER model or relational tables. In general, and as a first step, the facts with associated measures are determined by analyzing the existing documentation or the database structures. Facts and measures are elements that correspond to events occurring dynamically in the organization, i.e., frequently updated. If the operational databases are relational, they may correspond, respectively, to tables and attributes. If the operational databases are represented using the ER model, facts may be entity or relationship types while measures are attributes of these elements. An alternative option may be the involvement of users that understand the operational systems and can help to determine which data can be considered as measures. Identifying facts and measures is the most important aspect in this approach since they form the basis for constructing multidimensional schemas.

Different procedures can be applied for deriving dimensions and hierarchies. They may be automatic (Bonifati *et al.*, 2001; Golfarelli & Rizzi, 1998), semi-automatic (Böehnlein & Ulbrich-vom Ende, 1999), or manual (Cabibbo & Torlone, 2000; Moody & Kortink, 2000). The semi-automatic and manual procedures require knowledge of the specific conceptual models that are used for the initial schema and its subsequent transformations. Unlike automatic or semi-automatic procedures, manual procedures allow designers to find hierarchies embedded within the same entity or table, e.g., to find the city and the province attributes in a store entity type.

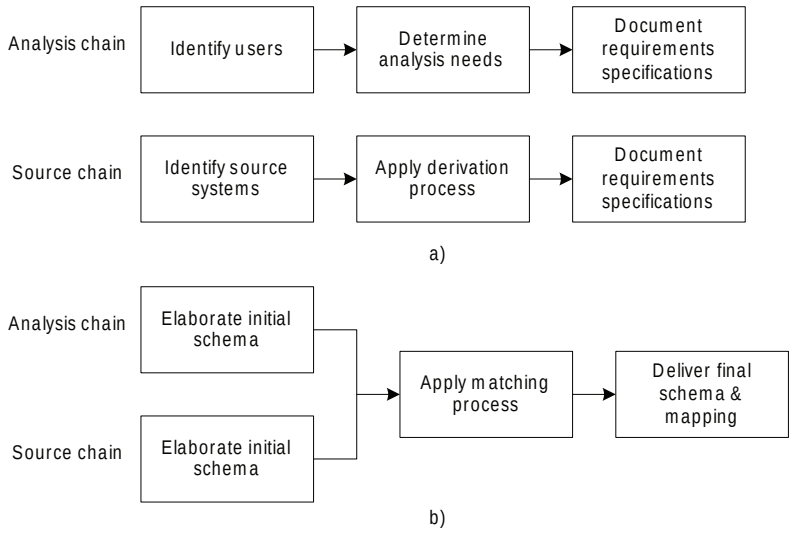
- **Document requirements specifications**: As in the analysis-driven approach, the requirements specification phase should be documented. The documentation should describe those source systems elements that can be considered as facts, measures, dimensions, and hierarchies.

Conceptual Modeling (Figure 2 b)

- **Elaborate initial schema:** Based on the elements identified in the previous phase, the conceptual DW schema is developed. Similarly to the analysis-driven approach, we recommend to use a conceptual model in order to facilitate future communication with business users and schema evolutions.
- **Determine user interest**: Until now user participation was minimal, responding only to specific inquiries from the designer. In this step, users are involved in a more active role. The schema is examined in detail in order to determine the type of analysis that can be done. However, the initial schema may require some modifications for several reasons: (1) it may contain more elements than those required by users for analytical purposes, (2) some elements may require transformations (e.g., attributes into hierarchies), (3) some elements may be missing (e.g., due to confusing names).
- **Elaborate final schema and mappings**: User recommendations about changes are incorporated into the initial schema leading to the final conceptual schema. In this stage, an abstract specification of mappings and transformations (if required) between data in source systems and DW schema is defined.

In every one of the above-mentioned steps of the conceptual modeling phase, the specification of business and technical metadata and the corresponding transformations should be developed.

Figure 3. Analysis/source-driven approach: a) requirements specification and b) conceptual modeling phases



Analysis/Source-Driven Approach

Requirements Specification (Figure 3a)

The analysis/source-driven approach for requirements specifications combines both previously-described approaches and may be used in parallel to achieve an optimal design. Therefore, two chains of activities can be distinguished: one that corresponds to business demands and another one that explores the source systems. Each type of activity results in identifying elements for initial multidimensional schemas.

Conceptual Modeling (Figure 3b)

Based on requirements specifications obtained from both chains, two initial DW schemas are developed. The schema obtained from the analysis-driven approach identifies the structure of the DW as it emerges from business requirements. The source-driven approach results in a DW schema that can be extracted from existing operational databases. In the next step of matching process, both schemas are compared (or integrated). This process is not an easy task. Different aspects should be considered, e.g., terminology used, degree of similarity between dimensions, levels, attributes, or hierarchies. Some solutions are provided by Bonifati *et al.* (2001) and Giorgini *et al.* (2005). An ideal situation arises when schemas cover the same analysis aspects, i.e., user demands are covered by data in operational systems and no other data exist for expanding the analysis spectrum. In this case, the schema is accepted and mappings between elements of the source systems and the DW are specified. Additionally, the documentation containing technical and business metadata about the DW, source systems, and required transformations is developed.

Nevertheless, in real-world scenario, both schemas rarely cover the same analysis aspects. Two situations may occur:

1. Users demand less information than what operational databases can provide. In this

case it is necessary to determine whether users may take into account new analysis aspects or eliminate from the schema those elements that are not of user interest. Therefore, another iteration in the analysis and the source chains is required. In this iteration either new users who may be interested in the new analysis possibilities will get involved or a new initial schema will be developed eliminating some elements from the schema.

2. Users demand more information than what operational databases can provide. In this case users may reconsider their demands and limit them to those proposed by the source-driven solution. Alternatively, users may require the inclusion of external sources or legacy systems that were not considered in the previous iteration but contain the necessary data. Therefore, new iterations in the analysis and the source chains may be needed.

FUTURE TRENDS

Defining a DW design methodology is still an ongoing research issue. Since the requirement specification phase is the most important phase in the overall DW design process, the different approaches should be evaluated. Therefore, it is necessary to determine a set of evaluation criteria to be able to compare the different approaches. Then, these criteria may be applied considering DWs with similar analysis purposes and data availability.

CONCLUSION

In this article we described three different approaches for requirements specifications and conceptual modeling of DWs that use, as a driving force, either the analysis of requirements, the data available in source systems, or a combination of both. Since these approaches are model and software independent, they can be used in different application domains.

Providing a systematic specification of different approaches to the DW developer team helps designers to understand their particularities and to choose an approach that better fits to user time constraints, their identification with business goals, and their motivation to be involved in the DW project.

REFERENCES

Bonifati, A., Cattaneo, F., Ceri, S., Fuggetta, A., & Paraboschi, S. (2001). Designing data marts for data warehouses. *ACM Transactions on Software Engineering and Methodology, 10*(4), 452-483.

Böehnlein, M., & Ulbrich-vom Ende, A. (1999). Deriving initial data warehouses structures from the conceptual data models of the underlying operational information systems. *Proceedings of the 2nd ACM International Workshop on Data Warehousing and OLAP,* 15-21.

Böehnlein, M., & Ulbrich-vom Ende, A. (2000). Business process oriented development of data warehouse structures. *Proceedings of Data Warehousing,* 3-21.

Cabibbo, L., & Torlone, R. (2000). The design and development of a logical system for OLAP. *Proceedings of the 2nd International Conference on Data Warehousing and Knowledge Discovery,* 1-10.

Freitas, G. M., Laender, A. H. F., & Campos, M. L. (2002). MD2: Getting users involved in the development of data warehouse application. *Proceedings of the 4th International Workshop on Design and Management of Data Warehouses,* 3-12.

Giorgini, P., Rizzi, S., & Garzetti, M. (2005). Goal-oriented requirements analysis for data

warehouse design. *Proceedings of the 8h ACM International Workshop on Data Warehousing and OLAP*, 47-56.

Golfarelli, M., & Rizzi, S. (1998). A methodological framework for data warehouse design. *Proceedings of the 1st ACM International Workshop on Data Warehousing and OLAP*, 3-9.

Inmon, W. (2002). *Building the data warehouse*. John Wiley & Sons Publishers

Kimball, R., Reeves, L., Ross, M., & Thornthwaite, W. (1998). *The data warehouse lifecycle toolkit: Expert methods for designing, developing, and deploying data warehouses*. John Wiley & Sons Publishers.

List, B., Schiefer, J., & Min Tjoa, A. (2000). Process-oriented requirement analysis supporting the data warehouse design process - A use case-driven approach. *Proceedings of the 11th International Conference on Database and Expert Systems Applications*, 593-603.

Luján-Mora, S., & Trujillo, J. (2003). A comprehensive method for data warehouse design. *Proceedings of the 5th International Workshop on Design and Management of Data Warehouses*.

Malinowski, E. (2008). Different kinds of hierarchies in multidimensional models. *In this book*.

Moody, D., & Kortink, M. (2000). From enterprise models to dimensional models: A methodology for data warehouse and data mart design. *Proceedings of the 2nd International Workshop on Design and Management of Data Warehouses*, 6.

Schiefer, J., List, B., & Bruckner, R. (2002). A holistic approach for managing requirements of data warehouse systems. *Proceedings of the 8th Americas' Conference on Information Systems*, 77-87.

Winter, R., & Strauch, B. (2003). Demand-driven information requirements analysis in data warehousing. *Proc. of the 36th Hawaii International Conference on System Sciences*, 1359-1365.

KEY TERMS

Analysis-Driven Design: Approach for designing a DW based on the analysis of user requirements or business processes.

Analysis/Source-Driven Design: Approach for designing a DW that combines analysis-driven and source-driven approaches.

Business Metadata: The information about the meaning of data as well as business rules and constrains that should be applied to data.

Conceptual Multidimensional Model: A set of objects and rules for representing in an abstract way multidimensional view of data consisting of facts with measures and dimensions with hierarchies.

Source-Driven Design: Approach for designing a DW based on the analysis of data available in source systems.

Source Systems: Systems that contain data to feed a DW. This may include operational databases and other internal or external systems.

Technical Metadata: The information about data structures and storage as well as applications and processes that manipulate the data.

ENDNOTES

1 This approach is also called demand driven.

2 Other names used are process driven, goal driven, or requirements driven.

3 This approach receives also the name of data or supply driven.

4 This approach is also called top-down/bottom-up analysis.

Chapter IX
Principles on Symbolic Data Analysis

Héctor Oscar Nigro
Universidad Nacional del Centro de la Provincia de Buenos Aires, Argentina

Sandra Elizabeth González Císaro
Universidad Nacional del Centro de la Provincia de Buenos Aires, Argentina

INTRODUCTION

Today's technology allows storing vast quantities of information from different sources in nature. This information has missing values, nulls, internal variation, taxonomies, and rules. We need a new type of data analysis that allows us represent the complexity of reality, maintaining the internal variation and structure (Diday, 2003).

In Data Analysis Process or Data Mining, it is necessary to know the nature of null values - the cases are by absence value, null value or default value -, being also possible and valid to have some imprecision, due to differential semantic in a concept, diverse sources, linguistic imprecision, element resumed in Database, human errors, etc (Chavent, 1997). So, we need a conceptual support to manipulate these types of situations. As we are going to see below, Symbolic Data Analysis (SDA) is a new issue based on a strong conceptual model called Symbolic Object (SO).

A "SO" is defined by its "intent" which contains a way to find its "extent". For instance, the description of habitants in a region and the way of allocating an individual to this region is called "intent", the set of individuals, which satisfies this intent, is called "extent" (Diday 2003). For this type of analysis, different experts are needed, each one giving their concepts.

Basically, Diday (Diday, 2002) distinguishes between two types of concept:

1. The *concepts of the real world*: That kind of concept is defined by an "intent" and an "extent" which exist, have existed or will exist in the real world.
2. The *concepts of our mind* (among the so called "mental objects" by J.P. Changeux (1983)) which frame in our mind concepts of our imagination or of the real world by their properties and a "way of finding their extent" (by using the senses), and not the

extent itself as (undoubtedly!), there is no room in our mind for all the possible extents (Diday, 2003).

A "SO" models a concept, in the same way our mind does, by using a description "d" (representing its properties) and a mapping "a" able to compute its extent, for instance, the description of what we call a "car" and a way of recognizing that a given entity of in the real world is a car. Hence, whereas a concept is defined by intent and extent, it is modeled by intent and a way of finding its extent is by "SOs" like those in our mind. It should be noticed that it is quite impossible to obtain all the characteristic properties of a concept and its complete extent. Therefore, a SO is just an approximation of a concept and the problems of quality, robustness and reliability of this approximation arise (Diday, 2003).

The topic is presented as follows: First, in the background section, the History and Fields of Influence and Sources of Symbolic Data. Second, in the focus section Formal definitions of SO and SDA, Semantics applied to the SO Concept and Principles of SDA. Third: Future Trends. Then Conclusions, References, Terms and Definitions.

BACKGROUND

Diday presented the first article on 1988, in the Proceedings of the First Conference of the International Federation of Classification Societies (IFCS) (Bock & Diday 2000). Then, much work has been done up to the publication of Bock, Diday (2000) and the Proceedings of IFCS'2000 (Bock & Diday 2000). Diday has directed an important quantity of PhD Thesis, with relevant theoretical aspects for SO. Some of the most representatives works are: Brito P. (1991), De Carvalho F. (1992), Auriol E. (1995), Périnel E. (1996), Stéphan V. (1996), Ziani D. (1996), Chavent M. (1997), Polaillon G. (1998), Hillali Y. (1998), Mfoummoune E. (1998), Vautrain F. (2000), Rodriguez Rojas O. (2000), De Reynies M. (2001), Vrac M. (2002), Mehdi M. (2003) and Pak K. (2003).

Now, we are going to explain the fundamentals that the SDA holds from their fields of influence and the most representative authors:

- ***Statistics:*** From Statistics the SO counts. It *knows* the distributions.
- ***Exploratory analysis***: The capacity of showing *new relations* between the descriptors {Tukey, Benzecri}(Bock & Diday 2000).
- ***Cognitive sciences and psychology***: The membership function of the SO is to provide *prototypical instances* characterized by the most representative attributes and individuals {Rosch} (Diday, 2003).
- ***Artificial intelligence***: The representation of *complex knowledge*, and the form of manipulation. This is more inspired from languages based on first order logic {Aristotle, Michalski, Auriol} (Diday, 2003).
- ***Biology:*** They use taxonomies and solutions widely investigated in this area {Adamson}(Bock & Diday 2000).
- ***Formal concept analysis***: The Complete Object Symbolic is a Galois Lattice {Wille R.} (Polaillon, 1998).

In some of these sciences, the problem is how to obtain classes and their descriptions (Bock & Diday, 2000)

The "*Symbolic data tables*" constitute the main input of a SDA, helped by the *Background Knowledge* (Diday, 2003). In the chapter 1 of Bock H and Diday E's book are mentioned as follow: each cell of this symbolic data table contains data of different types:

(a) *Single quantitative value*: if « height » is a variable and w is an individual: height (w) = 3.5.
(b) *Single categorical value*: Town (w) = Tandil.

(c) *Multivalued*: in the quantitative case height (w) = {3.5, 2.1, 5} means that the height of w can be either 3.5 or 2.1 or 5.

(d) *Interval*: height (w) = [3, 5], which means that the height of w varies in the interval [3, 5].

(e) *Multivalued with weights*: for instance a histogram or a membership function (Possibility, Beliefs or Capacity Theory can be also applied). Variables can be:

  (a) *Taxonomic*: the colour is considered to be "light" if it is "white" or "pink".

  (b) *Hierarchically dependent*: we can describe the kind of computers of a company only if it has a computer, hence the variable "does the company have computers?" and the variable "kind of computers" are hierarchically linked.

  (c) *With logical dependencies*: if age (w) is less than 2 months then height (w) is less than 10.

Example: We need to model the concept of work activity by continent. We have stored the Tables 1 and 2.

Now we have second order units representing the concept activity of our clients by continent. Each row of symbolic table represent one SO. The variables show the values for the class, for example SO-Manufactures: the variable Study Level shows equal probability. The clients are distributed 33 % in Asia and 66 % in Europe. The age is between 28 and 50 years.

Symbolic data are generated when we summarize huge sets of data. The need for such a summary can appear in different ways, for instance, from any query to a database, which induces categories and descriptive variables. Symbolic Data can also appear after a clustering in order to describe in an explanatory way (by using the initial variables), the obtained clusters (Diday, 2004).

Symbolic data may also be "native" in the sense that they result from *expertise knowledge* (type of emigration, species of insects), the probable distribution, the percentages or the range of any random variable associated to each cell of a stochastic data table, time series, confidential data, etc. Also, they result from Relational Databases in order to study a set of units whose description needs the merging of several relations (Billard & Diday, 2007).

Table 1. Customer

#Customer	…	Age	Country	Study Level	Work's Activity
041	…	50	Spain	Medium	Manufactures
033	…	45	China	High	Manufactures
168	…	30	Australia	Low	Agriculture
457	…	39	Sudan	High	Services
542	…	35	Argentina	Medium	Agriculture
698	…	48	India	High	Services
721	…	60	France	High	Services
844	…	53	Canada	Medium	Services
987	…	42	Italy	Low	Agriculture
1002	…	28	Germany	Low	Manufactures
1299	…	34	EEUU	Medium	Agriculture

Table 2. Taxonomy

Country	Continent
Spain	Europe
China	Asia
Australia	Oceania
Sudan	Africa
Argentina	America
India	Asia
France	Europe
Canada	America
Italy	Europe
Germany	Europe
EEUU	America

MAIN FOCUS

SO and SDA: Formal definitions

A SO is a triple s = (a, R, d) where R is a relation between descriptions, d is a description and "a" is a mapping defined from Ω *in L depending on R and d* (Diday, 2003).

SDA is the extension of standard Data Analysis to symbolic data tables as input in order to find SO as output. This is based on four spaces: the space of individuals, the space of concepts, the space of descriptions modeling individuals or classes of individuals, the space of symbolic objects modeling concepts (Asso page).

Most of the SDA algorithms give in their output the symbolic description "d" of a class of individuals by using a "generalization" process. By starting with this description, SOs give a way, to find at least, the individuals of this class. Example: The age of two individuals w_1, w_2 are age(w_1) = 30, age(w_2) = 35, the description of the class C = {w_1, w_2} obtained by a generalization process can be [30, 35]. In this simple case the SO "s" is defined by a triple: s = (a, R, d) where d = [30, 35], R = " ∈ " and "a" is the mapping: Ω → {true, false} such that a(w) = the true value of age(w) R d denoted [age(w) R d]. An individual w is in the extent of s iff a(w) = true (Diday, 2002).

There are two kinds of SO:

Boolean SOs: The instance of one binary relation between the descriptor of the object and the definition domain, that is defined to have values true or false. If [y (w) R d] = {true, false} is a Boolean SO (Diday, 2002). Example: s=(color ∈ {red, white}), here we are describing an individual /class those color is red or white.

Modal SOs: In some situations, we can't say true or false, we have a degree of belonging, or

Table 3. Symbolic table modeling the concept Work's activity

Work's activity	*Study level*	*Continent*	*Age*
Agriculture	"low"(0.50), "medium"(0.50)	"America"(0.5), "Europe"(0.25), "Oceania"(0.25)	[30:42]
Manufactures	"low"(0.33), "medium"(0.33), "high"(0.33)	"Asia"(0.33), "Europe"(0.66)}	[28:50]
Services	"medium"(0.25), "high"(0.75)	"Africa" (0.25), "America"(0.25), "Asia"(0.25), "Europe"(0.25)	[39:60]

Multivalued attribute with weight

Interval

some linguistic imprecision as always true, often true, fifty-fifty, often false, always false; now we say that the relation is fuzzy. If [y (w) R d] ∈ L = [0,1] is a Modal SO (Diday & Billard, 2002). Example: s=(color ∈[(0,25) red, (0,75) white]), at this point we are describing an individual/class that has color: 0,25 red; 0,75 white. This weighs can have different meaning, which are going explain to below.

Prof. Diday call "**assertion** " a special case of a SO defined by s = (a, R, d) where R is defined by [d' R d] = ∧ i =1, p [d'_i R_i d_i] where "∧" has the standard logical meaning and "a" is defined by: a(w) = [y(w) R d] in the Boolean case. Notice that considering the expression a(w) = ∧ i =1, p [y_i (w) R_i d_i] we are able to define the SO s = (a, R, d) (Diday & Billard, 2002).

The **extension** of a SO is a function that permit us recognize when an individual belongs the class description.

In the Boolean case, the extent of a SO is denoted Ext(s) and defined by the extent of a, which is: Extent (a) = {w ∈ Ω / a (w) = true}. In the Modal instance, given a threshold α, it is defined by Extα (s)= Extentα (a)= {w ∈ Ω / a (w) ≥ α}(Diday, 2002).

It is possible to work with SOs in two ways,

- **Induction:** We know values of their attributes then we know what class they belong.
- **Generalization:** We want to form a class from the generalization/specialization process of the values of the attributes of a set of individuals.

Semantics Applied to the SO Concept

The semantics are applied in the Modal SOs, and are the weights observed in the previous example of this type of SO (Bock & Diday, 2000):

Probabilities: This is the classical situation; Kolmogorov 's axiom:

Pr(E1 U E2)= Pr(E1) + Pr(E2) - Pr(E1 ∩ E2)

Capacities: At this point, there are groups with probabilistic multinomial variables. The capacity is interpreted in Choquet's sense: "The probability of at least one individual belongs this value", the capacity of a modality is the union capacity. Consequently, the capacity of SO_1 and SO_2 for M_1 is $p_{11} + p_{21} - p_{11} * p_{21}$, and the capacity of SO_1, SO_2, …, SO_n for M_1 is computed using the associative property.

Possibilities: When the frequency theory fails. Example: We have a Math's book, but the word "mathematics" appears few times. The mathematician knows what this book is about. In this situation we need different experts to give us their experience. So Zadeh's axiom is used for fuzzy sets:

Pos(E1 U E2)= Máx.(Pos(E1), Pos(E2))

Beliefs: Here, the expert says: "I think that the probability of the occurrence of A is p1, p2 is B, but I do not know everything of A and B ". If p1 + p2 < 1, then 1 - (p1 + p2) expresses his ignorance. Schaffer's axiom (Bender, 1996):

Bel(E1 U E2) ≥ Bel(E1) + Bel(E2) - Bel(E1 ∩ E2)

Dumpster's rule allows combining the beliefs of various experts.

We can see that in these theories emphasis falls mainly on the knowledge of the expert. The two first theories work with empirical information, the last ones have a higher level of information based on concepts.

Principles of SDA

Diday defines the main principles of SDA by the following steps (2008, pp. 22):

1. A SDA needs two levels of units. The first level is that of individuals, the second that of concepts.
2. A concept is described by using the symbolic description of a class of individuals defined by the extent of a category.
3. The description of a concept must take account of the variation of the individuals of its extent.
4. A concept can be modeled by a SO which can be improved in a learning process, taking into account the arrival of new individuals.
5. SDA extends standard exploratory data analysis and data mining to the case where the units are concepts described by symbolic data.
6. The output of some methods of SDA (clustering, symbolic Galois lattice, decision trees, Kohonen maps, etc.) provides new SOs associated with new categories (i.e., categories of concepts).
7. The new categories can be used in order to define new concepts such as in step 2, and so on.

There is an important number of methods (Diday et al, 2008) developed to create, to analyze and to visualize SO, which were implemented in Sodas 1.2 and Sodas 2.5 software through SODAS and ASSO projects respectively; whose aim is to analyze official data from Official Statistical Institutions (ASSO or SODAS Pages).

FUTURE TRENDS

The most important features to be incorporated from theoretical point of view are based on many mathematical results on reducing the information lost by the generalization process used from the first-level to the second level units (by copula models, rules, taxonomies), the learning of the structure of SOs (by lattices, non-linear pyramids or hierarchies) and their robustness, quality, validity, reliability have to be obtained. Much remains also to be done by extending statistics, multidimensional data analysis and data mining to symbolic data, for example, in time series, multidimensional scaling, textual data, sequential and stochastic classification, grid distributed data mining, spatial classification, rule extraction and neural networks extended to symbolic data.

We have developed a conceptual architecture for Data Warehouse and Data Mining based on SO concept (González-Císaro & Nigro, 2008). The next step is the implementation and testing with classical architectures.

CONCLUSION

The need to extend standard data analysis methods (exploratory, clustering, factorial analysis, discrimination ...) to symbolic data tables in order to extract new knowledge is increasing due to the expansion of information technology, now being able to store an increasingly larger amount of huge data sets. This requirement, has led to a new methodology called "SDA" whose aim is to extend standard data analysis methods to a new kind of data table called "symbolic data table" and to give more explanatory results expressed by real world concepts mathematically represented by easily readable "SOs".

Some advantages are: It preserves the confidentiality of the information; It supports the initial language in which they were created; It allows the spread of concepts between Databases. Being independent from the initial table, they are capable of identifying some individual coincidence described in another table.

The concept of SO is very important for the construction of the Data Warehouse and it is an important development for Data Mining, especially for the manipulation and analysis of aggregated information.

REFERENCES

ASSO, Project home page. http://www.info.fundp.ac.be/asso/. Last access on June 2008.

Billard, L., & Diday, E. (2007). *Symbolic Data Analysis: Conceptual Statistics and Data Mining .Wiley Series in Computational Statistics.* Chichester, West Sussex, England: John Wiley & Sons Ltd.

Bock, H., & Diday, E. (2000). *Analysis of Symbolic Data*. Studies in Classification, Data Analysis and Knowledge Organization. Heidelberg, Germany: Springer Verlag-Berlin.

Chavent, M. (1997). Analyse des Données symboliques. *Une méthode divisive de classification*. Thèse de doctorat in Sciences. I'Université Paris IX-Dauphine.

Diday, E. (2002). An introduction to Symbolic Data Analysis and the Sodas software. *The Electronic Journal of Symbolic Data Analysis.* Retrieved from http://www.jsda.unina2.it/volumes/Vol0/Edwin.PDF on December 2007.

Diday, E., & Billard, L. (2002). *Symbolic Data Analysis: Definitions and examples.* Retrieved from http://www.stat.uga.edu/faculty/LYNNE/tr_symbolic.pdf. on November 2007.

Diday, E. (2003). Concepts and Galois Lattices in Symbolic Data Analysis. *Journées de l'Informatique Messine*. Knowledge Discovery and Discrete Mathematics Metz, France.

Diday, E. (2004). From Data Mining to Knowledge Mining: Symbolic Data Analysis and the Sodas Software. *Workshop on Applications of Symbolic Data Analysis.* January 2004. Lisboa Portugal.

Diday, E., & Noirhomme-Fraiture, M. (Eds.) (2008). *Symbolic Data Analysis and the SODAS Software*. Chichester, West Sussex, England: Wiley-Interscience.

González Císaro, S., & Nigro, H. O. (2008). Architecture for Symbolic Object Warehouse. In J. Wang (Ed.), *Encyclopedia of Data Warehousing and Mining - 2nd Edition*. Hershey PA: Idea Group Publishing.

Noirhomme-Fraiture, M. (2004). Visualisation of Symbolic Data. *Workshop on Applications of Symbolic Data Analysis.* Lisboa Portugal. Retrieved from http://www.info.fundp.ac.be/asso/dissem/W-ASSO-Lisbon-Visu.pdf. on May 2008.

Polaillon, G. (1998). *Organisation et interprétation par les treilles de Gallois de données de type multivalué, intervalle ou histogramme*. Thèse Docteur in Informatique. I'Université Paris IX-Dauphine.

Touati, M., & Diday, E. Sodas Home Page. http://www.ceremade.dauphine.fr/~touati/sodas-page-garde.htm. Last access on June 2008.

KEY TERMS

Artificial Intelligence: The field of science that studies how to make computers "intelligent". It consists mainly of the fields of Machine Learning (neuronal networks and decision trees) and expert systems.

Decision Trees: A method of finding rules or *(rule induction)* which divide the data into subgroups which are as similar as possible with regard to a target variable (variable that we want to explain).

Exploratory Analysis: It is part of the Data Analysis French School, developed among 1960 and 1980. The principal authors are Tuckey and Benzecri. The process of analysis takes as a target

to discover new relations between the sets of the analyzed information.

Extension: "I call the extension of an idea the subjects to which it applies, which are also called the inferiors of a universal term, that being called superior to them." Arnault and Nicole (1662).

Formal Analysis Concept: is a theory of data analysis, which identifies conceptual structures among data sets; Rudolf Wille introduced it in 1982. It structures data into units that are formal abstractions of concepts of human thought, allowing meaningful and comprehensible interpretation. FCA models the world as being composed of objects and attributes. A strong feature of FCA is its capability of producing graphical visualizations of the inherent structures among data. In the field of information science there is a further application: the mathematical lattices that are used in FCA can be interpreted as classification systems. Formalized classification systems can be analyzed according to the consistency of their relations. (FAC Home Page http://www.upriss.org.uk/fca/fca.html).

Fuzzy Sets: Let $\boldsymbol{U}$ be a set of objects so called universe of discourse. A fuzzy set $\boldsymbol{F}$ in $\boldsymbol{U}$ is characterized by a function of inclusion μ_F taking values in the interval [0,1], i.e. $\mu_F : \boldsymbol{U} \rightarrow [0,1]$; where $\mu_F(u)$ represents the degree in which $u \in \boldsymbol{U}$ belongs to fuzzy set $\boldsymbol{F}$.

Galois Lattice: Galois Lattice provides some meanings to analyze and represent data. This refers to two-ordered set. An ordered set (I,#) is the set I together with a partial ordering # on I.

Intension: This is the comprehension of an idea. "I call the comprehension of an idea the attributes which it contains and which cannot be taken away from it without destroying it." Arnault and Nicole (1662).

Chapter X
Database Engineering Supporting the Data Evolution

Luiz Camolesi Júnior
State University of Campinas, UNICAMP, Brazil

Marina Teresa Pires Vieira
Methodist University of Piracicaba – UNIMEP, Brazil

INTRODUCTION

Researchers in several areas (sociology, philosophy and psychology), among them Herbert Spencer and Abraham Maslow, attribute human actions resulting in continual environmental changes to the search for the satisfaction of individual and collective needs. In other fields of science, this behavior represents a challenge in ethical researches on concepts, methodologies and technologies aimed at optimizing and qualifying the actions involved in these continual changes to obtain better results.

Specifically in computer science, software engineering is a critical sub-area for these researches and their application (Lehman & Stenning, 1997), since it involves the construction of models and orientation for their use in the development of resources, such as software, to support the user's needs. Databases should be included in this context as a component for data storage (Table 1).

Considering the premise of continuous changes (table 2) and the human needs involved (Khan & Khang, 2004), the consequences for software and for the required database are obvious. In the field of computational science, these changes in

Table 1. Laws of evolution (Lehman & Stenning, 1997)

Continuing Change
Increasing Complexity
Self Regulation
Conservation of Organization Stability
Conservation of Familiarity
Continuing Growth
Declining Quality
Feedback System

the modern world are reflected in evolutionary features for software and databases, based on Database concepts, structures and processes that allow for rapid, albeit not traumatic, shifts to new industrial, commercial or scientific systems (Mcfadden et al., 1999) in new contexts (*temporal scenarios*) (Camolesi, 2004).

BACKGROUND

Database models must comprise representation elements that are adaptable to the user's varying and dynamic needs, and contain the taxonomy needed for their manipulation. Thus, traditional (generic) database models such as the Entity-Relationship (ERM) and Relational (RM) models (Siau, 2004) have been expanded with appropriate "profile" for specific applications and requirements. Considering their purpose of supporting changes, "Profile Models" can be easily referenced in scientific researches as:

- **Version model**: Considering versions as database objects derived (originating from, but containing alterations) from others, models of this "profile" must be applied to a database characterized by the explicit and voluntary storage of the historical information about object changes (Conradi & Westfechtel, 1998). The features frequently specified in versions models are:
- **Derivation structure:** Establishes the data structure for organizing versions, e.g., stack, tree or not cyclic digraph, linked by special relationships representing linear or nonlinear derivation actions;
- **Versionable element:** Establishes which *variant* elements (database objects) can have versions created and represented in the database;
- **Property of versioning:** This is a feature that serves to define a versionable element. This property must be dynamically established for each element during either its creation or its definition;
- **Version status:** Status set (state or situation) for the versions;
- **Manipulation of versions:** The creation, update and deletion of versions can be accomplished implicitly either by the database system or by the user through specific command language;

Table 2. Type of changes

Evolution: actions for an (*variant*) element's technological progress, improvement, modernization or correction.	**Revolution**: alteration actions of an element can influence the element's purpose in the context.
Involution: simplification actions of an element, regression in its conception or content.	

- **Operation restrictions:** The manipulation of versions by users can be granted unconditionally or restrictions may be imposed for each operation (create, update and delete).
- **Time model**: In models of this "profile", the elements that represent the dimension time are established essentially to control the **Evolution of the Database**. The reliability of such time-based models depends on the unambiguous definition of temporal limits to be imposed in any business or scientific database system. In an evaluation of systems using the time representation, database designers find many variations and ambiguous representations that can degenerate the processing of the time value simply because they are ignorant of how time is represented, how it can be analyzed or how it should be converted. Using a Time Model for the homogeneous representation of the data type *time* and *interval* enables the designer to improve the evolution control performance. Based on many researches about time representation and utilization (Bettini et al., 1998), the following features are identified for a homogeneous data type definition:
- **Moment:** A time instant value;
- **Granularity:** Precision domain of time instant, this feature can be based on the ISO 8601 standard, e.g., PnYnMnDTnHnMnS, or any other standard established by the application;
- **Orientation:** Reference system for temporal representation, e.g., Gregorian calendar (UTC or Coordinated Universal Time), Chinese calendar, Jewish calendar or others;
- **Direction:** All orientation has an moment of origin (0) and a time may be the moment preceding or following this origin moment;
- **Application:** Specification of the use of the temporal representation, allowing for the semantic recognition of the type, independently of the context in which it is inserted. The attribute *application* should indicate *Occurrence*, *Duration* or *Frequency.*
- **Configuration model:** This "profile model" is based on and related to the version model, with version aggregations defined as *configurations* or *releases* (Conradi & Westfechtel, 1998). The *configurations* or *releases* are logical aggregation components or artifacts, selected and arranged to satisfy the needs of applications based on composition abstractions (Sabin & Weigel, 1998). Applications that require the composition of database objects are related with engineering, i.e., Computer Aided Software Engineering (CASE) and Computer Aided Design (CAD);
- **Integrity model:** Required in all data models (RM, OOM and ORM), elements are established in models of this "profile" to support data consistency and integrity. In certain typical databases, the number of constraints and actions for database integrity may involve hundreds of elements, which require periodical reviews since they are strictly related to continually changing real situations. The elements in these models vary in form and purpose, the most common being: rules, *business rules* (Date, 2000) and *database constraints* (Doorn & Rivero, 2002);
- **User model:** Necessary in all data models (RM, OOM and ORM), the elements that represent the users are established in models of this "profile". The modeling of user features is critical in the evolution of a database because it involves a diversity of needs and changes in the operational behavior (insert, delete, alter and select) in the database. The modeling of human behavior to design access privileges is a well-consolidated analytical process (Middleton et al., 2004). However, this process must take into account the static and dynamic aspects of

people and their activities. Analyses based on a static approach define *User Roles* that are appropriate for traditional applications, whose activities change with relative infrequence. In dynamic applications, however, static definitions are insufficient, since the requirements call for temporary and specific activities that are necessary to support the dynamic definition of *User Roles.*

Database engineering involves database models and "profile models" applied in development methodologies (Elmasri & Navathe, 2006), i.e., a collection of correlated techniques arranged in a logical order, with orientation rules for the materialization of an objective. The objective of a database engineering methodology should be the creation of an optimized database (based on ORM or OOM) flexible to changes implemented through well-executed engineering phases (elicitation of requirements, viability analysis, design, testing and implementation), particularly in the design stages (conceptual, logical and physical) in which database models and "profile models" are used.

The logical order of a methodology established by a database designing or engineering group should reflect the group's level of maturity and its knowledge of the work context, i.e., the group's dedication to the design, which can be based on two distinct methodologies:

- **Sequential engineering:** Traditional methodology for database engineering in which the phases are executed linearly, and can be based on the bottom-up approach, i.e., from details of the data requirements (attributes) to the recognition of elements (entity or objects). In top-down sequential engineering, the elements are first identified and then refined in detail;
- **Concurrent engineering:** Methodologies initially created for conventional areas of engineering (mechanical and electrical), they establish the simultaneous development of a "product" through the cooperation of designing groups working separately but in constant communication and exchange of information (Carter & Baker, 1992). Concurrent (or Simultaneous) Engineering has been adapted to the broad and complex process of software engineering but, in this case, systems may be required to support the cooperative work among project groups, such as Groupware software, researched in the CSCW (Computer Support Cooperative Work) area.

The methodology may also depend on the stage of the database's life cycle (creation, implementation, maintenance). Database Maintenance supervised by Administration (DBA) is motivated solely by the system's degenerating performance resulting from the constant modernization (restructuring) of the Database (Ponniah, 2003). If degeneration is not prevented, it can lead to increasing information access costs, rendering the use of a database unfeasible.

In some cases, the database maintenance operations are insufficient or not adapted to maintain the database qualities. This is usually the case when maintenance and documentation updates in a database have been neglected over a long period. Thus, evolution and change-oriented database engineering should be:

- **Reverse engineering:** Methodologies to recognize and represent the structural, functional and behavioral aspects of an already operating database. This mode of engineering focuses on the process steps needed to understand and represent a current database, and must include an analysis of the context in which the database was engineered. Depending on the case, the database should be treated as a "black box" to

recognize its data, or physically analyzed to identify special data structures and storage solutions;

- **Reengineering:** Methodologies for converting databases in obsolete environments to more modern technologies. Reengineering introduces techniques for the restudy and conversion of obsolete data to new realities and the resulting redesign of databases. Reengineering can comprise three phases: Reverse Engineering, Delta (Δ) and Forward Engineering. Reverse Engineering involves the abstract and clear definition of data representations. In the Delta phase, the designer executes modifications in database systems to incorporate new functionalities (positive Δ) or to reduce them (negative Δ) in order to accomplish complete or partial implementations. The third and last phase, Forward Engineering, refers to the normal development of database systems following the natural stages.

MAIN THRUST OF THE CHAPTER

The "profile models" were created or inserted in Object-Oriented Models (OOM); however, the creation and adaptation of the Object-Oriented paradigm to traditional data models enabled "profile models" to be widely applied, for example, in Unified Model Language – UML (Naiburg & Maksimchuk, 2002) and in Object-Relational Models (ORM). Although not described in the current literature as "profile models", these models have well-defined purposes with a variable degree of flexibility in relation to the generic database models.

Though still incipient, the composition of these "profile models" for use in database engineering allows a database to be defined with features which are essential for its changeability, characteristic associated to quality features listed in table 3, allowing rapid changes with the smallest possible impact on the database. Proving the importance of changeability, are the emergent frameworks and patterns searching to unify and to integrate the "profile models" in database engineering.

FUTURE TRENDS

Database development methodologies, models and processes are constantly being updated to support the changing requirements that occur during the recovery processes of requirements or

Table 3. Features to evolution

- **Traceability**: The monitoring is an essential task for efficiency tracing of the designer and team jobs, aiming at better evaluation of production (modeling and engineering). To this, the *influence* among elements should be inserted in the project to allow database engineers to recognize automatically which are the consequences of the alterations accomplished in relation to all the elements of a system. These consequences can be characterized as inconsistencies of data, what demand the revision of all the reached elements, direct or indirectly, for the accomplished alterations. The correct definition of the influences allows optimizing the actions of inconsistencies verification and consequently the reduction of the maintenance costs.
- **Flexibility**: Facility degree to changes with the smallest possible impact on the project and the database user. Models used and project accomplished must be capable to support new requirements (Domingues et al., 2003).
- **Portability**: The necessities to transfer the database to new environments (Operating Systems or DBMS), new technologies (programming language) or interfaces can be a demand for the evolution of a database. To this, the database modeling accomplished must be capable to support these changes with low impact.
- **Compatibility**: Facility degree to insert new elements in database model. To this, the database modeling must obey the rigid standards of concepts and techniques to accept specifications or components of the same standard.

maintenance (Paton & Diaz, 1999). With theses scenarios, the status of traditional database design shifted from a complex software engineering phase to that of engineering (Roddick et al., 2000). Thus, database engineering today serves as the foundation for the development of data modeling adapted to frequent changes in dynamic requirements established by the user.

The recognizing of needs for Database Evolution is a more complex task (table 4) that involves questions such as: What should be altered? When should it be executed? How should it be executed? What are the benefits? What is the cost of this process? Who will be affected?

These issues have motivated researches on emergent topic as:

- Database Administration Policy or *Database Maintenance Policy* with the definition of data-driven rules to management of expired solutions and innovation of models;
- Aspect-Oriented Database Engineering with new perspectives on evolution based on separation of concerns related to evolution. The aspects modeling (Table 3) supports the non-functional properties of database and the identification of influences, using specific languages to represent "profile models";
- Database Modeling Languages and Database Engineering Tools with resources to assure the Features to Evolution (Table 3) in database engineering, not forgetting that the choice of degree evolution depends on the maturity designer team and stability (or instability) of user's requirements.

Accompanying the evolution of database development methodologies and processes, the requirements to improve data access performance and to support decision making have resulted in the development of Data Warehouse (DW) technologies. A Data Warehouse maintains frequently data from multiple sources usually organized in a multidimensional model to support On-line Analytical Processing (OLAP). Materialized views are used to minimize the cost of answering frequently asked queries. They are a valuable component in the design of a Data Warehouse, requiring an approach for the selection and management of materialized data .The set of materialized views changes dynamically according to the needs of the users, thus requiring tools to manage dynamic collections of materialized aggregate views in a data warehouse under a resource constraint, typically disk space.

CONCLUSION

As can be seen from the above descriptions, Database Evolution is a vast subject under constant discussion and innovation, which explains why so many subjects require analysis, particularly those involving "profile models". Despite specific researches on this theme, pragmatic interest in the process of Database Evolution is not usually

Table 4. Database maintenance operations

Correction of the Database Schema in response to problems found during the database operation phase.	**Integration** of Database Schemas for the creation of a single database.
Adaptation of the Database Schema to the new elements resulting from new requirements.	**Updating** of the schema elements to conform the changing user requirements and business evolution.
Refinement of the elements of the Database Schema that were insufficiently detailed during the design phase	**Incorporation** of several databases into a "Federated Database", maintaining the independence of each individual database.

reflected in these researches due to the lack of details or progress achieved.

The difficulties encountered in many researches may be attributed to the theme's complexity. The evolutionary process can be defined by many variations in form: voluntary or involuntary; explicit or implicit; temporal or not temporal; recorded or not in databases in historical form; executed through a simple update of information or involving the physical or logical restructuring of the database; following predefined semantic rules; originating from the user (through an interface) or from the DBMS (Database Management System).

Database Administrators (DBA) should clearly indicate that the process of changes in a database, regardless of the characteristics and techniques utilized, should maintain or expand the database's adaptability, flexibility and efficiency (Domigues et al., 2003) to conform to new technologies and to the company's growth-related requirements, without neglecting the crucial problem of application adaptations (Hick & Hainaut, 2003). With respect to Data Warehouses, tools for dynamically create views intends to relieve the Data Warehouses Administrator (DWA) from having to monitor and calibrate the system constantly.

REFERENCES

Bettini, C., Dyreson, C. E., Evans, W. S., Snodgrass, R. T., & Wang, X. S. (1998). A Glossary of Time Granularity Concepts. *Lecture Notes in Computer Science, 1399, 406-413.* Springer-Verlag Publishing.

Camolesi, L. (2004). Survivability and Applicability in Database Constraints: Temporal Boundary to Data Integrity Scenarios. *Proceedings of V IEEE International Conference on Information Technology: Coding and Computing – ITCC, 1,* 518-522. IEEE Computer Society Press.

Carter, D. E., & Baker, B. S. (1992). *CE: Concurrent Engineering.* Boston, MA: Addison-Wesley Publishing.

Conradi, R., & Westfechtel, B. (1998). Version Models for Software Configuration Management. *ACM Computing Surveys, 30*(2), 232-282.

Date, C. J. (2000). *What not How: The Business Rules Approach to Applications Development.* Boston, MA: Addison-Wesley Publishing.

Domingues, E., Lloret, J., & Zapata, M. A. (2003). Architecture for Managing Database Evolution. *Lecture Notes in Computer Science, 2784,* 63-74. Springer-Verlag Publishing.

Doorn, J. H., & Rivero, L. C. (2002). *Database Integrity: Challenges and Solutions.* Hershey, PA: Idea Group Publishing.

Elmasri, R., & Navathe S. B. (2006). *Fundamentals of Database Systems* (5th ed.). Boston, MA: Addison-Wesley Publishing.

Hick J. M., & Hainaut J. L. (2003). Strategy for Database Application Evolution: The DB-MAIN approach. *Lecture Notes in Computer Science, 2813,* 291-306. Springer-Verlag Publishing.

Inmon, W.H. (2005). *Building the Data Warehouse* (5th ed.). Indianapolis, IN: Wiley.

Khan, K., & Khang Y. (2004). *Managing Corporate Information Systems Evolution and Maintenance*. Hershey, PA: Idea Group Publishing.

Lehman, M. M., & Stenning V. (1997). Laws of Software Evolution Revisited. *Lecture Notes in Computer Science, 1149,* 108-124. Springer-Verlag Publ.

McFadden, F. R., Hoffer, J. A., & Prescott, M. B. (1999). *Modern Database Management.* Boston, MA: Addison-Wesley Publishing.

Middleton, S. E., Shadbolt, N. R., & Roure D. C. de (2004). Ontological User Profiling in Recommender Systems. *ACM Transaction on Information Systems, 22*(1), 54-88.

Naiburg, E. J., & Maksimchuk, R. A. (2002). *UML for Database Design.* Boston, MA: Addison-Wesley Publishing.

Paton, N. W., & Diaz, O. (1999). Active Database Systems. *ACM Computing Surveys, 31*(1), 63-103.

Ponniah, P. (2003). *Database Design and Development: An Essential Guide for IT Professional.* Indianapolis, IN: Wiley-IEEE Press.

Roddick, J. F. et. al. (2000). Evolution and Change in Data Management- Issues and Directions. *ACM SIGMOD Record, 29*(1), 21-25.

Sabin, D., & Weigel, R. (1998). Product Configuration Framework – A survey. *IEEE Intelligente Systems, 13*(4), 42-49.

Siau, K. (2004). *Advanced Topics in Database Research, 3.* Hershey, PA: Idea Group Publishing.

KEY TERMS

Business Rule: Expression or statement that represents a restriction of data or operations in a business domain. A collection of Business Rules is a behavioral guide to support the Business Policy (organizational strategy). Business Rules can be implemented as Database Constraints, depending on the form and complexity of the restriction (Date, 2000).

Configuration: Collection of versions of different elements that make up a complex element. Versions in a configuration must be stable, i.e., they cannot be altered but can be changed by a new version of the same element. Configurations are abstractions representing semantic relationships among elements. Configurations serve to establish the scope of an application (temporal, spatial or user) (Sabin & Weigel, 1998). Revised Configurations consolidated by users can be defined as Releases. *Versions* of a release cannot be altered or changed by another version. Configurations defined as releases cannot be deleted because they are or were used. User identifiers, valid intervals for use and constraints are some important items of information associated with releases in databases.

Data Warehouse (DW): A subject-oriented, integrated, nonvolatile, time-variant collection of data in support of management's decisions (Inmon, 2005).

Database Constraints: Boolean functions associated with elements of a database being used to evaluate the integrity of actions or data that are inserted, removed or updated. A Database Constraint can be defined as a set of predicates $P_1 \wedge P_2 \wedge ... \wedge P_k$, where each predicate possesses the form $C_1 \theta C_2$, and where C_1 is an attribute, θ is a comparison operator and C_2 is either an attribute or a constant (Camolesi, 2004). The specification of a database constraint can be formalized using the Object Constraint Language – OCL (Object Management Group)

Database Maintenance Policy: Policy can be defined as a plan or guide to constrain the actions executed by an individual or a group. The Database Maintenance Policy establishes rules for the action of database designers that are executed in response to intrinsic and constant maintenance-related alteration needs, based on the designers' empirical knowledge and work capacity. The definition of goals, implementation phases, results and desired outcomes is information associated with every policy or subset of rules.

Influence: Type of dependency association to conceptually and logically represent interrelations among elements and to define their levels of involvement prior to a process of alteration of the data (Object Management Group). Influence modeling is based on impact specifications of modifications in database objects during the engineering process.

Temporal Scenario: Is a concept that adequately represents and includes the characteristics typical of evolution, and is used in adaptive, dynamic or flexible systems. The role of temporal scenarios is to represent situations in which the database requirements have been or will be modified (Camolesi, 2004). Every scenario reaches a moment in time when its existence begins (*opening*). Starting from this moment, the actors begin to perform (*acting*). Scenarios may reach a moment in time when they cease to exist (*closing*). Scenarios can *open*/*close* several times (*recurrent scenarios*). The definition of the opening and closing moments is obligatory for the specification of the temporal existence of *temporary scenarios.* The exceptions to this rule are the *permanent scenarios*, which do not *close*, and the ones that do not *open*.

Time and Interval (datatype): Fundamental datatypes to establish the temporal reference in a database. Time is symbolized by real numbers, based on a sequence of representative values to meet the application. Interval is an aggregation of two time moments intended to delimit and characterize the interval. The interval may represent *continuous* or *discrete* time.

Variant: Applied to many structural elements of a database (class, table, constraint etc), it is used to define mutable elements or elements whose modification is highly probable. An invariant is the opposite of a variant, i.e., an element not involved in the evolution of the database. Variants of database objects or object properties can be modified and can generate versions.

Version: An alternative of a database element, with variations in structure, function or behavior, used in the context of an application as a boundary of the work executed. A version can represent a state of an evolving element with variant and invariant properties (Conradi & Westfechtel, 1998). A collection of actions executed on a database element can generate an element version if it results in significant alterations of the element's characteristics.

Chapter XI
Versioning Approach for Database Evolution

Hassina Bounif
Ecole Polytechnique Fédérale de Lausanne (EPFL), Switzerland

INTRODUCTION

Schema evolution is an important research topic with an extensive literature built up over the years. However, databases are still reluctant to change and thus their evolution is difficult to achieve because the evolution of the schema involves several issues at different levels of the database schema such as the change management at the logical level. Several approaches have been proposed to achieve the evolution of a schema of a wide range of types of databases. Versioning, modification and views are examples of these chosen approaches. In this paper, we present and discuss one of these approaches, which is the versioning approach for database evolution. The future trends of the versioning are presented as well.

BACKGROUND

Databases are the core of information systems and their roles are essential within companies or organizations. These databases are subject to changes for several reasons that include the changes undergone to the real-world or the emer-

gence of new database user requirements. For example, (Banerjee and Kim, 1986) and (Peters and Ozsu,1997), consider that the schema of the database changes for several reasons such as: a) changes to the real-world, (b) changes necessitated by errors in the schema due to poor design, and (c) changes triggered by changes to applications that access the schema (and data). In addition to that, we consider that the evolution of databases can depend on other additional factors such as the technology. When the technology changes over time, this requires most of the time, the evolution of the schema; for example, when upgrading the DBMS (DataBase Management System), a database administrator is confronted to change some schema components types or structures to put forward better database system performances.

Schema evolution means modifying a schema within a populated database without loss of stored data taking into consideration two problems: firstly, the semantics of change (i.e., the effects of the changes on the schema) and, secondly, the change propagation (the propagation of the schema changes to the underlying existing instances. ontology

Considerable advances have been made in data structures, rules, constraints, schemata models and meta-models in order to resolve the problem of schema evolution. Rahm and Bernstein (2006) proposed an online catalogue of bibliographic references to schema evolution and related fields, such as generic data management, evolution of ontologies, software evolution and workflow evolution. Because of lack of space, we just cite some important works taken from this online catalogue:

a. **Four families of solutions:** the solutions of schema evolution belong to one of the four existing families, which are modification (Banerjee and Kim, 1986), versioning (Loomis and Chaudhri, 1997), views (Bellahsène, 1996) and combining the approaches. We realise that all of these families of approaches complement each other. For instance, in the schema modification, changing the schema may lead to a loss of information. However, in the schema versioning approach, replication of the schema avoids data loss, but creates complex navigation through the different generated versions and slows the DBMS (Database Management System). While with views, changes can be simulated on the schema without changing the underlying database and no conversion is needed, however, there are several issues associated with the view update such as the problem of performances of a system that needs to compute views based on other views and the problem to update the object returned from a view. Combining approaches allows to avoid the problems mentioned above, but is characterized by complexity and onerous mechanisms to be executed.

b. **Solutions applied to the conceptual level:** several studies have focused on the evolvability of database schema at the conceptual level because conceptual models:
   1. are the first artifacts to which a change is or should be applied (Borgida and Williamson, 1985)
   2. increase the level of abstraction that influences the evolution of schemas (Verelst, 2004).

Research studies that propose this avenue in order to resolve the problem of schema evolution are numerous. Examples include the EVER diagrams (Liu et al., 1994), metrics for conceptual models (Wedemeijer, 2000) and schemas mapping (Yu and Popa, 2005).

c. Solutions applied to the logical level: as at the conceptual level, the researchers have been very active at this level and have proposed many different solutions that suit the logical level, such as a content application platform CAP (Wienberg et al.,

2002), adaptive database via constrained relationships (Christodoulakis et al., 1989), equivalent schemas using path independence language (Lu et al., 1999), polymorphic reuse mechanisms (Osborn, 1989), semantic approach (Banerjee and kim, 1986), separation of concerns (Rashid and Sawyer, 2000), time-stamped versions and lazy strategy (Fontana et Dennebouy, 1995), temporal and versioning approach (Galante, 2002), relational algebra (McKenzie and Snodgrass, 1990) hybrid relations (Takahashi, 1990) and lossless conditional schema evolution (Jensen et al., 2004).

d. **Solutions applied to any level:** there are solutions that are independent and can be applied at any level, whether at the conceptual or the logical levels. In Perez-Schofield et al. (2002), the solution to schema evolution uses a container-based persistent model and a research prototype called Barbados programmed with C++. Lerner and Habermann (1990) propose an approach that involves the automation of the database schema evolution using OTGen (Object Transformer Generator) that supports complex schema changes and database reorganization.e.

e. **Solutions for specific types of databases and schemas:** other types of databases having particular schemas are also concerned about the evolution of their schemas. This is the case with the evolution of a data-warehouse schema that uses a set of materialized views to carry out the changes that occur (Bellahsène, 2002).

f. **Solutions for change management:** Karahasanovic (2001) presents a tool called SEMT (Schema Evolution Management Tool), which finds impacts of schema changes on applications in object-oriented systems. (Chen,1995) proposes a knowledge-based approach and mechanism. Kupfer et al. (2006) propose the co-evolution method in which a database schema and an ontology are connected, but are able to evolve independently without any constraints.

After this short presentation of the schema evolution and the main approaches adopted to resolve it, we return now to the presentation and the discussion of the versioning.

In the databases field, the concept of versioning means to preserve the existing components of the database that need to be changed over time in one or several versions and to incorporate the desired changes on these components in one or several other versions in such a way that these existing components continue to be used by existing applications along with the changed components within the new versions.

These components are on the one hand the database schema components (meta-data), and on the other hand their related data. They can or cannot be duplicated into the generated versions depending on the versioning strategies adopted on the database management system and the database administrator versioning strategy choice. The versioning methods and approaches are explained later in this paper.

SCALABILITY OF THE VERSIONING

The versioning can be executed at different levels. It depends on the data model. For example, numerous research proposals show it at the schema level in the relational model, others at the class level, at the object (instance) level and at the type level in the object model. A view can also be considered as a version. The unit to be versioned must have 1) a unique and immutable identity and 2) an internal structure. These two criteria have some implications for the possibilities of versioning within different data models. Object-oriented or ER-based data models clearly fulfill these two requirements (Munch, 1995). The adoption of an object-oriented data model is the most common choice in the literature on schema evolution by

schema versioning because this model does not require any special features. We discuss in the following three of the versioned units within an object-oriented data model which are: schema version, class version and object version.

- **Schema version:** a version of a schema can be represented in different ways. For example, it consists of:
  - a tuple < S, N> such as < S, N > is an identification of the schema S and version value N.
  - a triplet < S, N, V > such as < S, N > is an identification of the schema S and version value N. V is a set of classes connected by in-heritage relationship and references relationships, such as composition.

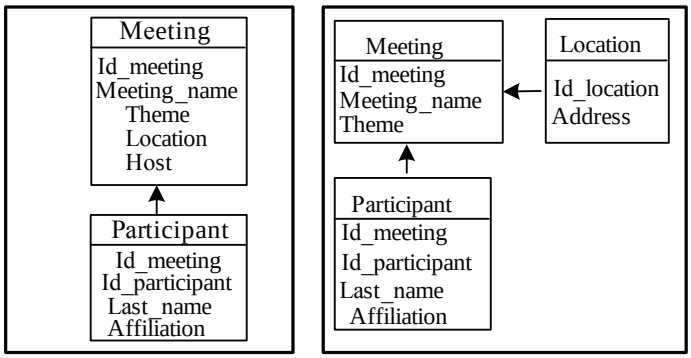
Orion (Banerjee and Kim, 1986) supports schema versioning

- Class version: the versions of a class are collected into a version set. When a class is changed, a new version of the class is created in its version set. Encore (Skarra and Zdonik, 1986) supports a class versioning mechanism. Generally classes are dependent of each other by several types of relationships such as generalization and aggregation. Therefore, when a class is versioned, the problem of the consistency of the configuration of the versioned class with the other existing classes is put forward.
- Object version: an object version is a triplet < O, N, v > in which O is an object identifier, N is a set of attributes and their values and v is the value of the version of an object. The versions of an object are called facets. Three operations associated with the concept of object version are: the creation of the first object version, the deletion of the object version and the derivation of a new object version. A direct acyclic graph is used to express all the relationships between all versions of a specified object (Lautemann, 1996).

Confusion can arise over the meaning of schema versions and the concepts schema modification and schema integration. However, according to Roddick (1995) and others, whose research focuses on database evolution, there is a consistent distinction between them, even if they are all linked to the database evolution domain at different levels:

- Schema Modification means making changes in the schema of a populated database. The old schema and its corresponding data are replaced by the new schema and its new data. This may lead to a loss of information.
- Schema Integration means combining or adding schemas of existing or proposed databases to a global unified schema that merges their structural and functional properties. Schema integration occurs in two contexts: 1) view integration for one database schema; 2) database integration in distributed database management where a global schema represents a virtual view of several schemas of databases taken from distributed database environments. This environment could be homogeneous, i.e., dealing with schemas of databases having the same data model and identical DBMSs (DataBase Management Systems), or heterogeneous, dealing with a variety of data models and DBMSs. This second environment deals with what is called, in Database domains, federated database schemas. In some research, schema integration is considered to be a particular case of schema evolution in which two or more schemata are integrated in one new schema (Roddick, 1995).

Figure 1. The versioning-by-copy (schema B is a version by copy of schema A)



VERSIONING METHODS AND APPROACHES

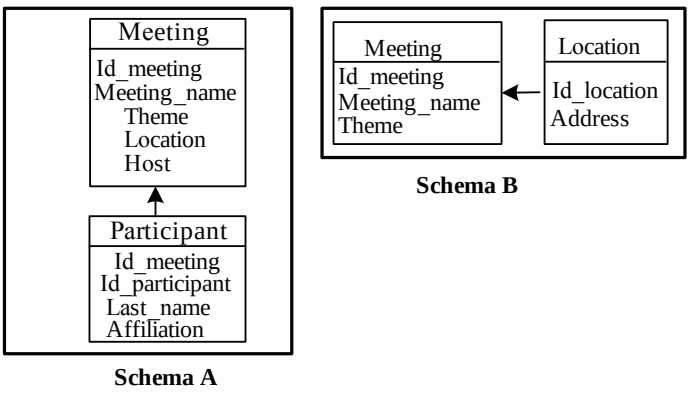
Various versioning methods exist in the literature. We cite the most important ones in this paper.

- Versioning-by-copy: a new version of a database component is created by making a copy of the component, assigning a version number to the copy and updating references to its predecessors. This technique consumes more disk space, but provides a form of backup.

We illustrate how the versioning-by-copy works in the example of the meeting in Figure 1. The database component in this case study to be versioned is the whole schema of the database (a schema version).

In the example, there are two schemas, **A** and **B**. Schema A is the original schema whereas the second schema B is a version-by-copy of schema A. Schema B is different from **A** because some of its components have undergone changes: the table **Meeting** is the same in both schemas. The table **Participant** has been changed in schema **B** and the table **Location** is created in schema **B**.

Figure 2. The versioning-by-difference (schema B is a version by difference of schema A)



Schemas **A** and **B** are both accessible to the users and to the existing and new applications running on top of them. Although the meeting table has been changed, there is no loss of data. It is still possible to access the schema and data before and after the evolution operation.

- Versioning-by-difference, also known as delta-versioning, requires that the differences between the new version and its predecessors be maintained. Although this technique saves disk space, it increases the overhead, as the DBMS needs to rebuild the particular version each time it is accessed.

We consider again the example of the Meeting in figure 2. In this example, there are two schemas, **A** and **B**. Schema **A** is the original schema whereas the second schema, **B** is a version- by-difference of schema **A**. Schema **B** contains only the components that have undergone changes. The table **Meeting** is not present in schema B because it has not changed. The table **Participant** has been changed, therefore it is present in schema **B** and the table **Location** is created in schema **B** because it is a new table.

Munch (1995) presented a set of versioning methods:

- Revisions (Sequential): the repetitive creation of a new version by modifying the most recent, previously new version. In the end, the versions form sequentially a single, linked list called a *revision chain* (Munch, 1995). Each version in the chain represents an evolution of the previous one. This method is used to version schemas. Because many copies of the database are created as schema versions, this technique is not very effective. It is adopted in the Orion System in which complete database schemas are created and versioned (Banerjee and Kim, 1986).
- Variants (Parallel): mean changing the relationships from one-to-one to many-to-one, so that many versions may have the same relationship with a common “ancestor" (Munch, 1995). A variant does not replace another, as does a revision, but is, instead, an alternative to the current version. This versioning method is adapted differently from one database to another, depending on the unit (a schema, a type or a view) to be versioned. For instance, Encore System (Skarra and Zdonik, 1986) uses type versioning.

Revisions and variants are usually combined in a common structure, called the *version graph*.

- **Merging:** consists of combining two or several variants into one (Munch, 1995).
- **Change Sets:** instead of having each object versioned individually, in this method the user collects a set of changes to any number of objects, and registers them as one unit of change to the database. This collection is termed a change set, or `cset.`
- Attribution is used to distinguish versions by the values of certain attributes, such as a date or status.

According to (Kim and Lochovsky, 1989), a version can be a transient version, a working version or a released version as follows:

- **Transient versions:** S transient version is considered unstable and can be updated and deleted. Therefore it is stored in its creator’s private workspace.
- **Working versions:** A working version is considered stable and cannot be updated, but can be deleted by its creator. It is also stored in its creator’s private workspace.
- **Released versions:** A released version is considered stable and cannot be updated or deleted.

Other approaches provide additional features:

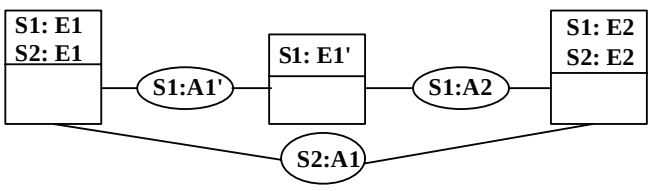
1. Semantics to versions such as Franconi et al. (2000) provide more semantics on versions and define reasoning tasks with Description Logic.
2. Refinement of the versioning model by proposing a new combined versioning approach with other approaches for schema evolution. An example is Benatallah (1996), who proposes an approach that combines modification with the versioning to benefit from their positive aspects for better management of the database schema evolution. In his approach, the researcher uses either the modification or the versioning approach, depending on the type of changes required in the schema. If the schema loses properties, such as the case of a suppression operation, a new version of the schema is generated. Otherwise, the schema is only modified.
3. Multi-representation, another way to implement the versioning approach for the schema evolution (Bounif et al., 2006). The multi-representation strategy uses stamps to have a schema that contains, at the same time, different representations for the modelling of the same universe of discourse (i.e., the modelling of the same real-world). For instance, in the following simple example, we have defined a stamp S =<S1, S2> in which the conceptual schema contains the entities E1, E1', E2 and the relationships A1', A2, according to the element S1 of the stamp S. However, according to the element S2 of the stamp S, the conceptual schema contains the entities E1 and E2 and one relationship A1. This is illustrated in Figure 3.

VERSIONED DATABASE FUNCTIONALITY

As mentioned in the previous section, the versioning is built using different methods and approaches. Consequently, several DBMSs support the versioning and the functionality of versioned database in different ways as well. Some commercial DBMSs are Ontos, Versant, ObjectStore, Objectivity/DB. The essential operations that a DBMS (Data Base Management System) provides to allow the versioning process are (Munch, 1995):

- creating new versions;
- accessing specific versions by a version selection mechanism;
- adding user-defined names or identifiers to versions;
- deleting versions (some restrictions to which versions may be deleted, if any, are created);

Figure 3. A simple example using multi-representation based on stamping on ER schema



- maintaining special relationships between versions, e.g., `revision _ of`, or `variant _ of`;
- changing existing versions, if possible;
- "freezing" certain versions so that they *cannot* be changed, if change is possible;
- merging variants, automatically and/or manually;
- attaching status values or other attributes to versions.

The user does not manage the versions. To gain non-versioned access to data stored in a versioned database, there are several alternative solutions (Munch, 1995):

1. Data is *checked-out* to a workspace, where applications can access it freely, and then *checked-in* to the database when it is no longer being used. This technique is most commonly used when dealing with files.
2. A layer is built on top of the versioned database, which provides a standard interface and conceals the details of the database, as well as the versioning. This technique is also useful for non-versioned databases.
3. Version selection is not part of the normal access functions of the database. Versioning is set up at the start and, possibly, the end of a change, or for a long transaction.

FUTURE TRENDS

The versioning approach allows the evolution of the schema over time but at the same time, presents drawbacks due to 1) the problems undergone with the choice of the unit to be versioned, 2) the problem of checking the consistency with the number of generated versions, 3) the problem of performance because the versioning is expensive and requires a great deal of memory space as the database evolves which is difficult to support in the long-term. Therefore, current research on versioning investigates the way to:

- Combine the units to be versioned for example in the object model, with a mix of object version, a class version and schema version.
- Combine the versioning approach with other approaches for schema evolution such as the Versioning-view approach
- Decrease the number of versions
- Speed up the access to data through the different versions.
- Concentrate on evolution based on an ontology versioning instead of database schema versioning because ontology contains more semantics and includes the contents of the schema.

CONCLUSION

Database evolution has been extensively addressed in the past. As presented in this paper, a variety of techniques have been proposed to execute change on the schema in the smoothest way to preserve data and maintain application functionalities.

The versioning approach is one of the solutions approved. The versioning principles can be applied universally to many different forms of data, such as text files, relational or object databases. Despite certain negative points that lead us to consider improvements within this approach such as combining the units to be versioned and combining the versioning approach with other approaches for schema evolution. Versioning is gaining ground each day and the work in schema versioning is expanding.

REFERENCES

Banerjee, J., Kim, H.-J., et al. (1986). Schema Evolution in Object-Oriented Persistent Data-

bases. *XP/7.52 Workshop on Database Theory*, University of Texas at Austin, TX, USA.

Bellahsene, Z. (1996). View Mechanism for Schema Evolution. *Advances in Databases, 14th British National Conferenc on Databases, BNCOD 14*, Edinburgh, UK, July 3-5, 1996. Edinburgh, UK: Springer.

Bellahsene, Z. (2002). Schema Evolution in Data Warehouses. *Knowledge and Information Systems, 4*(3).

Benatallah, B. (1996). Un *Compromis: Modification et Versionnement du Schéma. 12 èmes Journées Bases de Données Avancées,* 27-30 Août 1996, France INRIA.

Bounif, H., & Spaccapietra, S. et al. (2006). Requirements Ontology and Multi-representation Strategy for Database Schema Evolution., Seoul, Korea, 2006 *VLDB Workshop on Ontologies-based techniques for DataBases and Information Systems.*

Borgida, A., & Williamson, K. E. (1985). Accommodating Exceptions in Databases, and Refining the Schema by Learning from them. *11th International Conference on Very Large Data Bases VLDB'85*, Stockholm, Sweden, Morgan Kaufmann

Chen, J. L., Mcleod, D. et al. (1995). Domain-Knowledge-Guided Schema Evolution for Accounting Database Systems. *Expert Systems with Applications, 9*(4), 491-501.

Christodoulakis, D., Soupos, P. et al. (1989). Adaptive DB schema evolution via constrained relationships. *EEE International Workshop on-Tools for Artificial Intelligence,* 1989. Architectures, Languages and Algorithms, Fairfax, VA , USA, IEEE.

Fontana, E., & Dennebouy, Y. (1995). Schema Evolution by using Timestamped Versions and Lazy Strategy. *Onzièmes Journées Bases de Données Avancées*, 29 Août - 1er Septembre 1995, Nancy, France INRIA.

Franconi E., Grandi F., & Mandreoli F. (2000). *Schema Evolution and Versioning: A logical and Computational Characterisation at the 9th International Workshop on Foundations of Models and Languages for Data and Objects,* FoMLaDO/DEMM2000, Dagstuhl Castle, Germany, September 2000.

Galante, R. D. M., Edelweiss, N., et al. (2002). Change Management for aTemporal Versioned

Object-Oriented Database. *Conceptual Modeling - ER 2002, 21st International Conference on Conceptual Modeling,* Tampere, Finland, October 7-11, 2002, Proceedings, Tampere, Finland, Springer.

Jensen, O. G., & Bohlen, M. H. (2004). Lossless Conditional Schema Evolution. Conceptual Modeling - ER 2004. *23rd International Conference on Conceptual Modeling*, Shanghai, China, Springer.

Karahasanovic, A. (2001). SEMT-A Tool for finding Impacts of Schema Changes. NWPER'2000. *Ninth Nordic Workshop on Programming Environment Research*, Lillehammer, Norway.

Kim, W., & Lochovsky, F. H. (Eds.) (1989). *Object-Oriented Concepts, Databases, and Applications.* ACM Press and Addison-Wesley.

Kupfer, A., Eckstein, S. et al. (2006). *Handling Changes of of Database Schemas and Correspondin Ontologies Advances in Conceptual Modeling - Theory and Practice, ER 2006 Workshops BP-UML,* CoMoGIS, COSS, ECDM, OIS, QoIS, SemWAT, November 6-9, Tucson, AZ, USA, Springer.

Lautemann, S.-E. (1996). An Introduction to Schema Versioning in OODBMS. *Object-Oriented Databases 2 workshop,* Zurich, Switzerland, IEEE-CS Press.

Lerner, B. S., & Habermann, A. N. (1990). *Beyond schema evolution to database reorganisation. Conference on Object-Oriented Programming Systems, Languages, and Applications / European Conference on Object-Oriented Programming*, Ottawa, Canada.

Liu, C.-T., Chang, S.-K. et al. (1994). Database Schema Evolution using EVER Diagrams. *AVI '94, Proceedings of the Workshop on Advanced Visual Interfaces*, June 1-4, 1994., Bari, Italy, ACM.

Loomis, M. E. S., & Chaudhri, A. B. (1997). *Object Databases in Practice*. Prentice-Hall.

Lu, J., Dong, C. et al. (1999). Equivalent Object-Oriented Schema Evolution Approach Using the Path-Independence Language. *31st International Conference on Technology of Object-Oriented Languages and Systems*, Nanjing, China, IEEE Computer Society.

McKenzie, E., & Snodgrass, R. (1990). Schema Evolution and the Relational Algebra. *Inf. Syst., 15*(2), 207-232.

Munch, B. P. (1995). *Versioning in a software Engineering Database – the Change Oriented Way Division of Computer Systems and Telematics*. The Norwegian Institute of Technology.

Osborn, S. (1989). The role of Polymorphism in schema evolution in an object-oriented database. *IEEE Transactions on Knowledge Data Engineering, 1*(3).

Perez-Schofield, J. B. G., Rosello, E. G. et al. (2002). Managing Schema evolution in a container-based persistent system. *Softw., Pract. Exper., 32*(14), 1395-1410.

Peters, R. J., & Özsu, M. T. (1997). An Axiomatic Model of Dynamic Schema Evolution in Object-based Systems. *ACM TODS, 22*(1), 75-114

Rahm, E., & Bernstein, P. A. (2006). An Online Bibliography on Schema Evolution. *Sigmod Record, 35*, 30-31.

Rashid, A., & Sawyer, P. (2000). Object Database Evolution Using Separation of Concerns. *SIGMOD Record, 29*(4): 26-33.

Roddick, J. F. (1995). A survey of schema versioning issues for database systems. *Information and software Technology, 37*(7), 383-393.

Skarra, A. H., & Zdonik, S. B. (1986). The Management of Changing Types in an Object-Oriented Database. *Conference on Object-Oriented Programming Systems, Languages, and Applications* (OOPSLA'86), Portland, Oregon, USA.

Takahashi, J. (1990). Hybrid relations for database schema evolution. *14th Annual International Computer Software and Applications Conference*, Chicago, Ilinois, USA.

Verelst, J. (2004). The Influence of the level of Abstraction on the Evolvability of conceptual models of information systems. *International Symposium on Empirical Software Engineering (ISESE 2004)*, Redondo Beach, CA, USA, IEEE Computer Society

Wedemeijer, L. (2000). Defining Metrics for Conceptual Schema Evolution. *9th International Workshop on Foundations of Models and Languages for Data and Objects, FoMLaDO/DEMM 2000*, Dagstuhl Castle, Germany, Springer.

Wienberg, A., Ernst, M. et al. (2002). Content Schema Evolution in the CoreMedia Content Application Platform CAP. Advances in Database Technology - EDBT 2002. *8th International Conference on Extending Database Technology*, Prague, Czech Republic, March 25-27, Proceedings, Prague, Czech Republic, Springer.

Yu, C., & Popa, L. (2005). Semantic Adaptation of Schema Mappings when Schemas Evolve.

Proceedings of the 31st International Conference on Very Large Data Bases, Trondheim, Norway, ACM.

KEY TERMS

Multi-Representation: A multi-representation strategy that enables one to create two or more points of views of the same real-world and to combine them. The unit to be versioned can be one or several schema components or the whole schema.

Schema Evolution: Is the ability of the database schema to change over time without loss of stored data.

Stamp: A stamp S is defined as a vector $S=<s_1, s_2, ..., s_n>$ where each component *si of the S* represents the *i th* representation of the real-world.

Version Derivation: Is a direct acyclic graph (DAG) with which all the relationships between all versions of an object are specified

Version of an Object: Is a snapshot of this object taken at a certain period of time

Version: Is a unity that has a unique and immutable identity as well as an internal structure

Versioning-by-Difference: Is a delta compression that contains only the components of the schema that have been changed from one schema version to the next.

Versioning-View: Is a combined approach that implies the alternative use of the versioning and the view mechanisms to realize the evolution of the schema. The views are used to make the minor changes and the versioning to complete the complex ones.

Chapter XII
Evolutionary Database: State of the Art and Issues

Vincenzo Deufemia
Università di Salerno, Italy

Giuseppe Polese
Università di Salerno, Italy

Mario Vacca
Università di Salerno, Italy

INTRODUCTION

Waterfall methodologies can poorly cope with changes, making maintenance considerably an expensive process. For this reason, incremental and iterative methodologies were introduced (Larman & Basili, 2003). They view system development as a step-by-step process, with the introduction of new functionalities to meet user needs. The main problem arising in both paradigms is the complexity in facing changes. Therefore, an increased automated support in this task would result in a reduction of efforts and costs, especially in incremental methodologies, because it would make them more systematic.

Changes are often necessary to reflect the continuous evolution of the real world, which causes frequent changes in functional requirements. This entails frequent modifications to the software, yielding a gradual decay of its overall quality. For this reason, many researchers in this field have developed software refactoring techniques (Mens & Tourwé, 2004). Software refactoring is intended as the restructuring of an existing body of code, aiming to alter its internal structure without changing its external behavior. It consists of a

series of small behavior preserving transformations, which altogether can produce a significant software structural change. Moreover, system modifications resulting in changes to database structure are also relatively frequent (Roddick, 1995). These changes are particularly critical, since they affect not only the data, but also the application programs relying on them (Ambler & Sadalage, 2006; Karahasanovic, 2001).

Several disciplines have faced the problem of managing the effects of database schema changes. The interest in this topic has consistently grown, as shown by several surveys and bibliographies recently published (Roddick, 1992; Roddick, 1995; Li, 1999; Rahm & Bernstein, 2006). In particular, *schema modification* has faced the problem of changing the schema of a populated database. In addition to this, *schema evolution* pursues the same goal, but it tries to avoid loss of data. Alternatively, *schema versioning* performs modifications of the schema, but it keeps old versions to preserve existing queries and application programs running on it. Although schema versioning faces the problem of query and application programs preservation, it considerably increases the complexity and the overhead of the underlying DBMS. Finally, *database refactoring* aims to modify the database schema, and to change the corresponding application programs accordingly. In other words, the database refactoring is the process of slowly growing a database, modifying the schema by small steps, and propagating the changes to the queries. This is considered the major technique in the development of evolutionary database (Ambler & Sadalage, 2006).

With the introduction of evolutionary databases methodologies, the research area of schema evolution and versioning has been naturally broadening to embody new and more challenging research problems. In this chapter, borrowing the term from Ambler et al. (2006), we call this new research area *Evolutionary Database*.

This chapter discusses the main issues concerning evolutionary database and then we survey several models and tools proposed for their solution.

BACKGROUND

Schema evolution and schema versioning have been pioneer research areas facing problems due to the introduction of changes in database schemas.

Generally accepted definitions of schema evolution and schema versioning are provided in a paper by Roddick (Roddick, 1995), which can be considered as a sort of manifesto for this research topic. According to these definitions, the aim of schema evolution is to facilitate changes in the database schema without losing existing data. If these changes are seen as producing different versions, then the need to store the data of all the versions naturally arises. Therefore, the goal of schema versioning is to allow users to access the data of old versions (retrospective access) as well as those of new ones (prospective access).

Nowadays, researchers of the database area agree that schema evolution and versioning yield two main issues: *semantics of changes* and *change propagation* (see, for example, Peters, et al. 1997 or Franconi, et al., 2000). The first problem requires determining the effects of changes on the schema, whereas the second deals with the effects on the data.

Example 1

Let us consider the database of the employees of a University described by the schema **S** represented by the relation

R(*Employee_ID, Last Name, First name, Department_ID, Salary, Address*)

where *Employee_ID* is the primary key. Let us suppose that it is required that the database be in third normal form (Elmasri & Navathe, 2006). The addition of the functional dependency

f: *Department_ID* → *Address*

violates the third normal form. In order to preserve the normalization properties, it is necessary to transform the schema **S** in a schema **S'**, splitting **R** into two relations $\mathbf{R}_1$ and $\mathbf{R}_2$:

$\mathbf{R}_1$(*Employee_ID*, *Surname*, *Name*, *Salary*, *Department_ID*)

$\mathbf{R}_2$(*Department_ID*, *Address*)

This example shows how the addition of a functional dependency (a kind of schema change operation[1]) makes the schema **S** evolve in a new schema **S'** (**S** and **S'** are examples of versions). The splitting of **R** also requires transferring the existing data from **R** to $\mathbf{R}_1$ and $\mathbf{R}_2$ (change propagation).

The problems of *semantics of changes* and *change propagation* yield other main issues in the research areas of schema evolution and versioning, such as the definition of the aspects of the schema that can be changed, the way a schema is changed, the way these changes are propagated, the problem of maintaining schema consistency after a schema change, and the extent to which a database is still usable by application programs (Li, 2002).

According to Li (2002), the main schema-evolution research areas facing these problems are *schema evolvability*, *semantic integrity*, and *application compatibility*.

Schema evolvability deals with both changes to the structure of the schema (structural evolution), and to the application programs (behavioral evolution). The tasks of schema evolvability are: to study algebras or theoretical models for changes, to predispose models for checking schema consistency after changes, and to model architectures enabling the access to the different versions of a schema. The area of *semantic integrity* deals with the quality of the schema, verifying, for example, that referential integrity and constraint consistency are satisfied. Finally, the *application compatibility* studies the problems related to the compatibility between schema and applications after the changes.

MODELS AND SYSTEMS OF SCHEMA EVOLUTION AND VERSIONING

Many models, along with the systems embodying them, have been proposed to overcome the problems outlined in the previous section.

A pioneering attempt to model schema evolution was due to Lakshmanan *et al.* (1993). In their approach, schema evolution is interpreted as a special case of schema integration, being each stage of the evolution seen as a different database schema, which is linked to another one through a higher order logic language.

The theoretical models and related systems can be classified in two categories: *invariant and rule based models* (Banerjee, *et al.*, 1987; Nguyen & Rieu, 1988), and *formal* or *axiomatic models* (Peters and Özsu, 1997; Franconi, *et al.*, 2000). These models differ in the management of the effects of changes on the schema. Systems belonging to the former category are, for example, ORION (Banerjee, *et al.*, 1987), and Sherpa (Nguyen, *et al.*, 1988), while a system based on a formal model is Tigukat (Özsu, *et al.*, 1995).

The ORION model is structured into three components: a set of properties of the schema (*invariants*), a set of *schema changes*, and a set of *rules*. The invariants state the properties of the schema (for example, classes are arranged in a lattice structure), whereas rules help detecting the most meaningful way of preserving the invariants when the schema changes. *Schema changes* are organized into categories, like changes to the contents of a class. This model of schema evolution yields two important issues: completeness and soundness of the schema evolution taxonomy. Both of them have been proved only for a subset

of the schema change operations (Banerjee et al., 1987).

The *axiomatic* model was introduced by Peters *et al.* (1997). The aim of this model is to support the *dynamic schema evolution*, i.e., the evolution applied to a working system. This model has three basic components: *terms*, *axioms*, and *changes*. The basic concept underlying this model is the type (analogous to the class of ORION), which is in turn characterized by the terms. In fact, examples of terms are the lattice *T* of all types in the system, and the supertypes of a type *t*. The role of *axioms* is to state the properties of terms and their relations. For instance, they state that the lattice has no cycles. Changes on the schema are accomplished by means of three basic change operations: *add*, *drop*, and *modify*. The process of schema evolution is realized by elementary changes, whereas the problem of the effects of changes is solved by re-computing the entire lattice by using axioms. The model satisfies the properties of soundness and completeness. The system Tigukat embodies the axiomatic method illustrated above.

Another formal approach is provided in (Franconi, et al., 2000). It models schema versioning of object oriented databases from a logical and computational point of view, using a formal semantic framework based on Description Logic (Baader, 2003). The basic elements of the model are: classes and their attributes, schema, and elementary schema change operators. This framework allows broadening the number of consistencies that are considered as reasoning problems, according to the style of Description Logic, and checked formally. Finally, all the consistency problems taken into consideration have been proved to be decidable.

A serious and still partially unsolved problem is the management of compound changes (Lerner, 2000). Compound changes, like for example the movement of an attribute from a class to another one, are problematic as they cannot always be realized by a sequence of local changes. For example, a move change is not equivalent to a sequence of deletion and insertion, as the deletion causes a loss of data. In order to consider the possibility to operate compound changes avoiding the explosion of change operations allowed by the system, Lerner proposed a solution based both on giving the administrator the possibility to edit a desired schema, and on algorithms detecting changes in the schema and attempting to infer how types have changed. This approach has been implemented in the Tess system (Lerner, 2000).

The approaches based on the two models mentioned above present two main limitations. The former regards the explosion of rules when facing more general schema changes, whereas the second regards the fact that they are all suited to the object-oriented data model.

DATABASE REFACTORING

Analogously to software development, also database development approaches can be classified in traditional and evolutionary. Both try to cope with suddenly arising changes, but in very different ways. In fact, the formers try to minimize changes by foreseeing them during the design phase, whereas the second ones use changes as the underlying paradigm for developing the system.

Evolutionary database approaches are based on the assumption that a system is built incrementally and iteratively, according to the incoming requirements (Ambler & Sadalage, 2006). There are both advantages and disadvantages in applying these methodologies. An advantage is the possibility to facilitate the maintenance phase, since the system is always working. A drawback is the lack of tools providing automated support (Ambler, *et al.*, 2006).

Evolutionary methodologies are based on several techniques, among which the most important one is *database refactoring.* A database refactoring is a change to a database system that

improves its design without modifying both its informational and behavioral semantics (Ambler, *et al*., 2006).

Example 2

Let us consider the relation **R** of example 1. Merging *Last Name* and *First Name* attributes in a unique attribute, named *Identity*, is an example of database refactoring. Relation **R** would become

$\mathbf{R_1}$(*Employee_ID, Identity, Department_ID, Salary, Address*)

Application programs using **R** have to be updated for accessing $\mathbf{R_1}$. To this aim, a *transition period* (*deprecation period*), in which both the new and the old schema exist, will be used in the software system. At the end of this period only $\mathbf{R_1}$ will remain. This is an important *strategy* of database refactoring.

There are two kinds of changes: those that do not significantly affect the semantic of the schema (e.g., the changes of data format) and those that significantly affect the database schema (named *transformations*). As this distinction seems little meaningful to us, in the following with will use the term *database refactoring* to refer to any kind of change.

Example 3

Let us consider the database schema of example 1, and a query *Q* asking for all the employees working in the Computer Science Department (identified by label "CS"):

(query Q)
select *
from R
where *Department_ID* = "CS"

If the transformation consisting in the addition of a functional dependency *f* is applied to **S**, leading to the schema **S'**, also *Q* needs to be updated as follows:

(query Q')
select *Employee_ID, Last Name, First Name, Salary, Department_ID, Address*
from $\mathbf{R_1}$,$\mathbf{R_2}$
where ($\mathbf{R_1}$.*Department_ID* = $\mathbf{R_2}$.*Department_ID*) and (*Department_ID* = "CS")

The problem of database refactoring is very difficult since it requires modifying both schema and queries. As pointed out by Ambler *et al.* (2006), the level of difficulty depends on the degree of *coupling* between data and queries. Coupling indicates the measure of the dependence between two items, and in our case can be used to denote the dependence between applications and data. The higher is the coupling between applications and data, the more difficult is to change the schema. For example, a database with only one application program has a coupling lower than one with many of them.

So far database refactoring has been a task accomplished by database administrators, almost manually. In order to make this process easier, several categories of database refactoring have been studied, such as structural, data quality, referential integrity, architectural (aiming to improve the interaction between application programs and the database) (Ambler, et al., 2006).

It is worth to note that even if the problem of refactoring is strongly connected with schema evolution and versioning, it significantly differs from them. The main difference lies in the treatment of application programs and queries. In fact, schema evolution deals with the extent to which a schema change preserves application programs (*consistency* between schema and queries). The aim of schema versioning is more general with respect to that of schema evolution, as it allows to access both old and new versions of the schema, but the correctness of applications on the new schema is not guaranteed. Recently, Karahasanovic (2001) took in account the problem of the impact of schema changes on applications, proposing a tool for analyzing it a posteriori.

Therefore, if compared with schema evolution and versioning, database refactoring appears to be a more complete and complex task, since it combines different areas like schema evolution and query reformulation (Deutsch, 2006).

FUTURE TRENDS

So far the research on database refactoring has led to the definition of several methodologies (Ambler, *et al.*, 2006). However, no significative contribution has been provided towards the automation of this process. This is mainly due to the lack of formal approaches, like those developed for schema versioning and schema evolution (Banerjee, et al., 1987; Franconi, *et al.*, 2000; Lakshmanan, et al., 1993; Peters & Özsu, 1997). Nevertheless, these approaches use models that do not consider queries, hence they do not analyze the impact of schema changes on queries and application programs.

In the future formal approaches should be defined for analyzing both changes to the database schema and their impact on queries and application programs. They should enable us facing two important problems, which cannot be managed through the formal models described above: the variability of schema properties, and the propagation of changes into queries.

Formal approaches will provide the basis for tools capable to perform changes to database systems triggered upon the detection of database schema anomalies, and consequently, they will reduce the designer effort, providing the basis for automating the database refactoring process. These tools should also be communicative, in order to base their decisions also on user suggestions. For example, adding a functional dependency is a serious decision, and it would be desirable having the tools ask for user support.

CONCLUSION

In this article, we have presented a brief survey on the state of art of database evolutionary models and tools. We have examined representative works and related approaches to the problems of schema evolution and versioning, and of database refactoring. Finally, we have showed that if the problem of database refactoring is seen in connection with schema evolution and versioning, it originates a new research area, enriching previous ones with both theoretical (*models*) and application (*methodologies*) open problems.

REFERENCES

Ambler, S. W., & Sadalage, P. J. (2006), *Refactoring databases: Evolutionary database design*. Addison Wesley Professional.

Baader, F., Calvanese, D., McGuinness, D. L., Nardi, D., & Patel-Schneider, P. F. (2003). *The description logic handbook: Theory, implementation, and applications.* Cambridge University Press.

Banerjee, J., Kim, W., Kim, H.-J., & Korth, H.F. (1987). Semantics and implementation of schema evolution in object-oriented databases. *Proceedings of the 1987 ACM SIGMOD International Conference on Management of data,* 311-322. San Francisco, CA.

Deutsch, A., Popa, L., & Tannen, V. (2006). Query reformulation with constraints. *SIGMOD Record, 35*(1), 65-73.

Elmasri, R., & Navathe, S. B. (2006). *Fundamentals of database systems,* 5th edition. Addison-Wesley.

Franconi, E., Grandi, F., & Mandreoli, F. (2000). A general framework for evolving schemata support. *Proceedings of SEBD 2000,*.371-384.

Karahasanovic, A. (2001). Identifying impacts of database schema changes on application. *Proceedings of the 8th Doctoral Consortium at the CAiSE, 01*, 93-104.

Kuhn, T. S. (1970). *The structure of scientific revolutions*, 2nd Edition. University of Chicago Press.

Lakshmanan, L. V. S., Sadri, F., & Subramanian, I. N. (1993). On the logical foundations of schema integration and evolution in heterogeneous database systems. *Proceedings of Third International Conference of Deductive and Object-Oriented Databases* (*DOOD'03*), 81-100.

Larman, C., & Basili, V. R. (2003). Iterative and incremental development: A brief history. *IEEE Computer, 36*(6), 47-56.

Lerner, B. S. (2000). A model for compound type changes encountered in schema evolution. *ACM Transactions on Database Systems, 25*(1), 83-127.

Li, X. (1999). A survey of schema evolution in object-oriented databases. *Proceedings of TOOLS'31*, 362-371.

Mens, T., & Tourwé, T. (2004). A survey of software refactoring. *IEEE Transactions on Software Engineering, 30*(2), 126-139.

Nguyen, G., & Rieu, D. (1988). Schema evolution in object-oriented database systems. *Rapports de Recherche, 947.*

Özsu, M. T., Peters, R. J., Szafron, D., Irani, B., Lipka, A, & Muñoz, A. (1995). TIGUKAT: A uniform behavioral objectbase management system. *VLDB Journal, 4*(3), 445-492.

Peters, R. J., & Özsu, M. T. (1997). An axiomatic model of dynamic schema evolution in objectbase systems. *ACM Trans. Database Syst., 22*(1), 75-114.

Rahm, E., & Bernstein, P. A. (2006). An online bibliography on schema evolution. *Sigmod Record, 35*(4), 30-31.

Roddick, J. F. (1992). Schema evolution in database systems: An annotated bibliography. *ACM SIGMOD Record, 21*(4), 35-40.

Roddick, J. F., Craske, N.G., & Richards, T. J. (1993). A taxonomy for schema versioning based on the relational and entity relationship models. *Proceedings of the 12th International Conference on the Entity-Relationship Approach*, 137-148.

Roddick, J. F. (1995). A survey of schema versioning issues for database systems. *Information and Software Technology, 37*(7), 383-393.

KEY TERMS

Database Refactoring: It indicates little changes in the database schema which preserve both the meaning of the data and the behaviors of the applications. These changes improve the quality of the design.

Evolutionary Data Modeling: Methodologies to iteratively and incrementally model database systems so that schema and applications evolve in a parallel way.

Query Reformulation: Given two schemas S_1 and S_2 and a set of constraints. Let Q_1 a query on S_1 and Q_2 a query on S_2. Q_1 is a reformulation equivalent to Q_2 if Q_2 produces the same answers of Q_1 on the instances of the joint schema satisfying the constraints.

Schema Changes: They are operations which allow database scheme to evolve according to the changes of the external world.

Schema Evolution: It indicates the property of a database system which allows changes in the database schema without losing the existing data.

Schema Modification: It indicates the property of a database system which allows changes in the database schema of populated databases.

Schema Versioning: Given different schema versions, schema versioning indicates the capability of a database system to access to the data of all the versions both retrospectively and prospectively.

ENDNOTE

[1] A list of change operations in relational databases can be found in (Roddick, *et al*., 1993), whereas a list of change operations in object oriented databases can be found in (Banerjee, *et al*., 1987).

Chapter XIII
Interrogative Agents for Data Modeling

Vincenzo Deufemia
Università di Salerno, Italy

Giuseppe Polese
Università di Salerno, Italy

Mario Vacca
Università di Salerno, Italy

INTRODUCTION

The problem of changes in software development is a complex one, and it is almost impossible to avoid it. Indeed, the continuous evolution of the real world causes frequent changes in functional requirements, which entail frequent modifications to the software, yielding a gradual decay of its overall quality. To tackle this problem, two methodologies have been proposed: waterfall methodologies, and incremental/iterative methodologies. The formers try to prevent changes, whereas the second ones consider system development as a step by step process.

The concept of software refactoring is at the base of iterative and incremental methodologies. According to Fowler software refactoring is "*…a disciplined technique for restructuring an existing body of code, altering its internal structure without changing its external behavior.*" (M. Fowler, http://www.refactoring.com/).

Analogously, changes to the database structure are also relatively frequent (Roddick, 1995). They are particularly critical, since they affect not

only the data, but also the application programs accessing them (Ambler & Sadalage, 2006; Karahasanovic, 2001). Therefore, similarly to software refactoring, database refactoring aims to modify the database schema, and to change the corresponding application programs accordingly.

Ambler & Sadalage (2006) gave the definition of database refactoring: It "*is a simple change to a database schema that improves its design while retaining both its behavioural and informational semantics.*". The database refactoring is the basis of evolutionary data modeling methodologies, which are the database analogous of the iterative and incremental ones. Ambler & Sadalage (2006) also observed that a disadvantage in the application of refactoring is the lack of mature supporting tools.

In this paper we deal with the problem of developing tools supporting the evolutionary data modeling process. First of all, we observe that the characteristics of the problem can be naturally framed in the agent paradigm, because the evolutionary data modeling can be seen as a process in active databases able to change their beliefs and structure. Moreover, the evolutionary data modeling can be compared to the design of an agent acting in an open environment: the environment can be represented by the user needs and requirements (which change in an unforeseeable way), while the database development process is represented by the evolution of a reactive agent. Then, by following the *AOSE* (Agent-Oriented Software Engineering) view, we show that the use of tools and techniques from AI (Artificial Intelligence) can help facing the problem of developing supporting tools to automate evolutionary data modeling. To this end, after a brief introduction to the basic concepts in agent theory, and the highlighting of relationships among agents, software engineering, and databases, we point out the correspondence between agents and data modeling by showing a suitable architecture based on the logic of interrogation (Hintikka *et al.*, 2002).

BACKGROUND

Before dealing with the problem of database refactoring, along with the more general data modeling, and the role of agents in solving it, we need to briefly introduce the concept of agent, highlighting advantages of agent technology.

The term agent has not reached a universally accepted definition, as pointed out by Franklin & Graesser (1996), but the essence of being an agent can be summarized as follows:

*"An **autonomous agent** is a system situated within and a part of an environment that senses that environment and acts on it, over time, in pursuit of its own agenda and so as to effect what it senses in the future."* (Franklin & Graesser, 1996).

Luck *et al.* (2004) proposed to divide applications of agents in three main categories:

- *Assistant agents*, which replace humans in the execution of some task (e.g., agents for hotel reservation);
- *Multi-agent decision systems*, where the agents in the system make some joint decisions;
- *Multi-agent simulation systems*, used to simulate real-world domains like biological populations.

In this paper we are mainly concerned with particular applications of the first type, in which agents operate on databases (see, for example, de Carvalho Costa *et al.*, 2003; Magnanelli & Norrie, 2000). In particular, we are concerned with reactivity (the ability of the agent to respond in an appropriate way to the environment changes), and with the design of reactive systems needing to interact in open system environments[a].

Different architectures have been designed to realize the features of agents, the most famous of

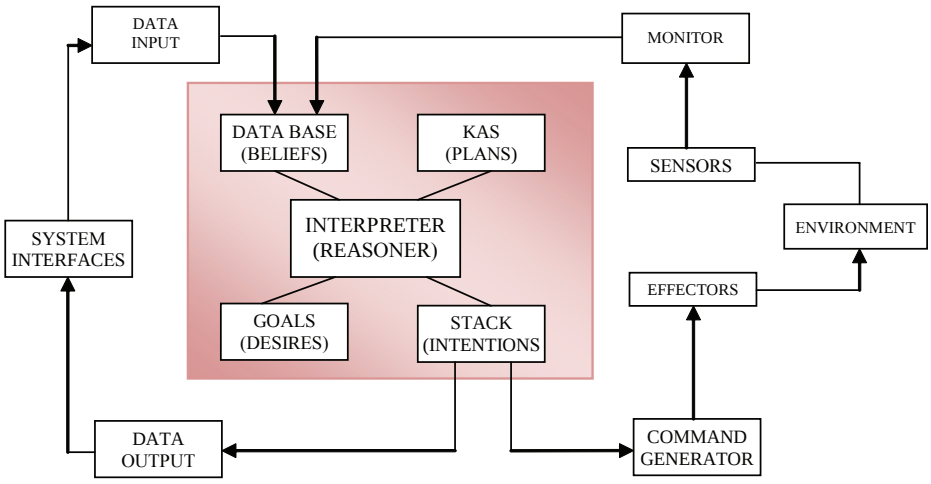
which is the so called *Belief-Desires-Intentions* (*BDI*, for short), based on the Georgeff & Lansky (1987) theory (see Figure 1). The BDI architecture is composed of four data structures: *beliefs*, *desires*, *intentions*, and *plans*, which are managed by an interpreter. The agent has two kinds of knowledge: *beliefs*, representing its knowledge about the world it interacts with, and *plans*, courses of actions representing its know-how. *Desires* are the tasks assigned to the agent, whereas *intentions* represent desires that the agent has committed to achieve. The *agent cycle* is realized by the interpreter operating on the data structures. The interpreter updates beliefs (after observations or interactions with the external world), generates new desires (due to the beliefs update), selects the current intentions from the desires and the plans to reach them.

Many logics for agent architectures have been developed (see, for example, Meyer 2004). In this paper we will refer to the interrogative model proposed in (Deufemia *et al.*, 2007), which is based on of the Hintikka *et al.* (2002) logic.

The theory of agents is not only a cognitive theory and a part of the AI, but agents can be considered a real technology, fruitfully applicable to other areas. In this paper we are interested in two areas to which agents can be applied: software and database engineering.

The agent technology and the field of software engineering are closely related; in fact, as Wooldridge & Ciancarini claimed "*Software engineers continually strive to develop tools and techniques to manage the complexity that is inherent in software systems. In this article, we argue that intelligent agents and multi-agent systems are just such tools.*" (Wooldridge & Ciancarini, 2001). The marriage between agents and software engineering produced a new branch of software engineering, Agent-Oriented Software Engineering (AOSE, for short)[b], whose aim is to exploit agents in the design of complex systems (for a survey of this discipline see, for example, Tveit 2001; Wooldridge & Ciancarini, 2001; Zambonelli & Omicini, 2004). AOSE has the merit to renew the relationship between AI and software engineering. In fact, Zambonelli & Omicini (2004) observed that the increasing complexity of problems calls for more intelligent abilities in systems.

Figure 1. The BDI architecture (Georgeff & Lansky, 1987)



The relationship between agents and databases has been studied from different points of view:

- active databases;
- DBMS functions;
- data management;
- distributed databases.

J. Bailey *et al.* (1995) showed that active databases and agents are very similar to each other; still, from the this point of view, van den Akker & Siebes, (1997), showed that extending the reasoning capability of the database can give the possibility of using different strategies to maintain a single constraint). de Carvalho Costa *et al.* (2003) studied the possibility of applying agents to accomplish DBMS functions.

Magnanelli & Norrie (2000) dealt with the benefits deriving from a sort of symbiosis between web agents and databases: the web agent uses the data stored in the database to search for information on the web, whereas the database, in turn, is updated by the data found by the agent.

Recently, Lockemann & Witte (2005) studied the relationship from the point of view of the semantic consistency problem in a peer-to-peer distributed database. They observed that it is appropriate to interpret information in the database in terms of beliefs, and to apply the techniques of belief representation and revision.

All these examples show some benefits or advantages deriving from some kinds of relationships between agents and databases. In the following we deal with another example of an effective relationship between agents and databases: the relation between agents and evolutionary data modeling.

THE AGENT BASED DATA MODELING

The refactoring of databases (and the more general evolutionary data modeling) requires that the system changes coherently both its own knowledge and its behavior after a request of change occurred. The process always starts with a *schema change request* that the administrator poses and the system tries to react to.

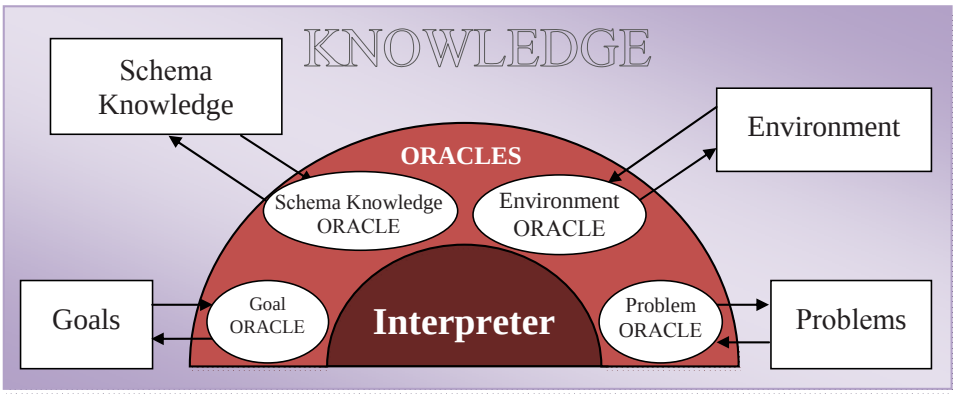
Data modeling can be seen as a *knowledge revision process*. In fact, schema change requests can involve objects in the database schema other than those directly involved in the request. This is due to the fact that schema change requests can generate inconsistencies which, in turn, can generate problems. For example, the *addition* of an attribute in a table can provoke the lacking of some functional dependency. Therefore, it becomes natural to see a support system as an agent based process aiming to operate on the schema in order to perform the required changes, and trying to preserve original properties in terms of knowledge and queries (Chang *et al.*, 2007).

A system can support the data modeling process to different extents, like checking the schema consistency, supporting database administrators during the process, or applying suitable consistency maintenance changes automatically. These examples of support systems can be accomplished by three kinds of agents each one characterized by some specific goals and abilities. For instance, the first agent needs only deductive capabilities, the second one requires communicative abilities, whereas the last needs to know how to use some problem solving heuristics.

In particular, a *data modeling interrogative agent* is constituted by an *interpreter* dialoguing with three knowledge sources, *Schema*, *Environment* and *Problem*, each dialogue being ruled by a specific goal (belonging to a set of goals). The whole process is based on the logic of interrogation[c] and the corresponding architecture is showed in Figure 2.

To make clear how the process works, let us consider a database storing data about employees of a university, and having a query for retrieving all employees of the Computer Science Department.

Figure 2. The architecture of the interrogative agent model



The *Schema* knowledge is constituted by information about the structure and the properties both of a general schema and of the Employee Database one. For instance, the functions and predicates in the following table may be useful in describing general schema properties:

Using these functions and predicates it is possible to express, in a general way, properties of database schema, such as "every table must have a primary key":

∀r.(**table**(r) => ∃k ⊆ **attr**(r) such that **primary_key**(r, k)).

The structure and the properties of the Employee Database are the following:

The *Environment* realizes the interface between the administrator and the support system; then, it manages information about the operations the administrator might perform on the database structure (*change operations*)[d]. For instance, the splitting of a table *t* into two tables *t'* and *t''*, due to the introduction of a new functional dependency *f*, can be described in Equation 1.

The *agent way of working* is as follows.

The Administrator gives a command, for example **split_table**(*R*, *R'*, *R''*, *f5*: Department_ID → Address), to the *Environment*, which, in turn,

Table 1. Examples general of information in the Schema source

PROPERTIES	**prop**(*p*, *S*) tests if *p* is a property of the schema *S*
ATTRIBUTES	**attribute**(*A*) tests if *A* is an attribute
TABLES	**table**(*R*) tests if *R* is a relation **attr**(*R*) returns the set of attributes of table *R*
QUERIES	**query**(*Q*) tests if *Q* is a query **body**(*Q*) returns the body of a query *Q* **var**(*Q*) returns the set of variables of a query *Q*
FUNCTIONAL DEPENDENCIES	**FD**(*f*) is true when *f* is a functional dependency **LHS**(*f*) returns the set of attributes on the left of the functional dependency *f* . **RHS**(*f*) returns the set of attributes on the right hand side of the functional dependency *f*.

Equation 1.

split_table(*t, t', t'', f*)	← (A' = A ∧ T' = (T − {*t*}) ∪ {*t', t''*} ∧ F' = F ∪ {*f*} ∧ Q' = {*q'*| **var**(*q'*) = **var**(*q*), **body**(*q'*) = ρ(**body**(*q*), *t, t'* ∩ *t''*)}[1] ∧ **attr**(*t'*) = **attr**(*t*) − **RHS**(*f*) ∧ **attr**(*t''*) = **RHS**(*f*))

Table 2. Examples of specific information in the Schema source

PROPERTIES **P**	**primary_key**(R, Employee_ID) (Every relation must have a primary key)
ATTRIBUTES **A**	Employee_ID, Name, Department_ID, Salary, Address
TABLES **T**	R(Employee_ID, Name, Department_ID, Salary, Address)
QUERIES **Q**	q(x, y, w, z) ≡ R(x, y, "CS", w, z)
FUNCTIONAL DEPENDENCIES **F**	f1 : Employee_ID → Name; f2 : Employee_ID → Department_ID f3 : Employee_ID → Salary f4 : Employee_ID → Address

passes it to the *Interpreter*, which has to decide what to do. The interpreter interacts with the *Goal* source to set the appropriate goal (see table 3 for a list of goals and their associated problems). In particular, the interpreter asks the *Goal* source for the problem ?**Consistent**(**split_table**(*R, R', R'', f5*)) which is the starting point of the process. The answer to the ?**Consistent**(*change-request*) is yes when each property in the set of properties obtained by the application of the change-request is true.

At this point the game starts. The *interpreter* interacts with the *Schema* in order to answer the question ?**Consistent**(**split_table**(*R, R', R'', f5*)).

A system with only the task to check for consistency will communicate the answer to the environment terminating the process, while a more sophisticated system will communicate the answer to the *Goal* source in order to know what is the new problem to solve.

In our case, if the answer to ?**Consistent**(**split_table**(*R, R', R'', f5*)) is negative, it will be necessary to solve the problem trying to apply some heuristic, and the *Goal* will select the new goals to be achieved, as showed in Table 4.

It is worth to note that, as the final answer is obtained by using heuristics, the *Interpreter* must ask the administrator to detect the possibility to apply it.

FUTURE TRENDS

In the future we think that it is necessary to implement support tools for evolutionary data modeling. To this end, we think that it is worth to investigate the possibility of developing a software tool based on the model presented here and using visual language based tools capable of supporting the data modeling process directly on

Table 3. Goals and associated problems in the Goal source

Goal	Associated problem
Checking the schema for the consistency	*Is the schema resulting from the required change still consistent?* Example: ?**Consistent**(**split_table**(R, R', R'', f5)
Searching for the inconsistencies	*Which are the false propositions?* Example: ∃p. **prop**(p, *Employees*) ∧ **not** p
Searching for the causes of the inconsistency	*Why is (are) the proposition(s) false?* *What makes the proposition(s) false?* Example: ?∃p'. **not prop**(p', *Employees*) => **not** p (Is the lacking of some property in the *Employees* schema to provoke the inconsistency?)
Trying to fix the inconsistency	*Is there a change operation that applied to the schema makes the inconsistency disappear?* Example: ?∃ ε' ∀p. (**prop**(p, ε'(*Employees*)) => p) (is there an operation ε' making true each property of the *Employees* schema?)

Table 4. Example of steps in the process of database refactoring

Goal	Problem	Answer
Checking the schema for the consistency	?**Consistent**(**split_table**(R,R',R'', f5))	NO
Searching for the inconsistency	?∃p. **prop**(p, *Employees*) ∧ **not** p	p ≡ **not** ∀r ∈ T ∃k ⊆ Attr(r) such that **primary_key**(r, k)
Searching for the causes of the inconsistency	?∃p'. **not prop**(p', *Employees*) => **not** p	p' ≡ p**rimary_key**(R'',Department_ID)
Trying to fix the inconsistency	?∃ ε'∀p. (**prop**(p, ε'(*Employees*)) => p)	It can't be solved
Asking the Problem source for an heuristic: Specialization	?∃ ε' **prop**(**primary_key**(R'',Department_ID), *Employees*)	**add**(**primary_key**(R'',Department_ID), *Employees*)
Checking the schema for the consistency	?**Consistent**(**add**(**primary_key**(R'',Department_ID), *Employees*)	YES
Consulting the Environment for the application	? **add**(**primary_key**(R'',Department_ID), *Employees*)	The DB administrator will decide

the database schema by means of special gesture operators.

CONCLUSION

In this article, we explored a new relationship between agents, databases and software engineering, which enforces both the thesis of Bailey *et al.* (1995) and of van den Akker & Siebes (1997), by stating that enriched active databases are able to face more complex problems and extending to single databases the claim of Lockemann & Witte (2005) about the use of beliefs. The framework presented uses an agent to discover and resolve inconsistencies during the process of data modeling. The proposed process is a *trial and error* one based on the Hintikka *et al.* (2002) view of logic, in which the agent tries to find a solution to a given problem by deducing or consulting other information sources.

REFERENCES

Ambler, S. W., & Sadalage, P. J. (2006). *Refactoring Databases: Evolutionary Database Design*. Addison Wesley Professional.

Bailey, J., Georgeff, M. P., Kemp, D. B., Kinny, D., & Ramamohanarao, K. (1995). Active Databases and Agent Systems - A Comparison. *Rules in Database Systems*, (pp. 342-356).

Chang, S. K., Deufemia, V., Polese, G., & Vacca, M. (2007). A Logic Framework to Support Database Refactoring. In *Proc. of 18th International Conference on Database and Expert Systems Applications, LNCS, 4653*, 509-51.

de Carvalho Costa, R. L., Lifschitz, S., & Vaz Salles, M. A. (2003). Index Self-tuning with Agent-based Databases. *CLEI Electronic Journal, 6*(1).

Deufemia, V., Polese, G., Tortora, G., & Vacca, M. (2007). Conceptual Foundations of Interrogative Agents. In *Proc. of Workshop from Objects to Agents* (*WOA'07*), (pp. 26-33).

Franklin, S., & Graesser, A. (1996). Is it an Agent, or just a Program? A Taxonomy for Autonomous Agents. *Proceedings of ATAL'96*, (pp. 21-35).

Georgeff, M. P., & Lansky, A. L. (1987). Reactive Reasoning and Planning. In *Proc. of the Sixth National Conference on Artificial Intelligence* (*AAAI-87*), (pp. 677-682).

Hewitt, C., & de Jong, P. (1982). Open Systems. *On Conceptual Modelling (Intervale)*, (pp. 147-164).

Hintikka, J., Halonen, I., and Mutanen, A. (2002). Interrogative Logic as a General Theory of Reasoning', in *Handbook of the logic of argument and inference. The turn towards the practical, 1*, 295-337.

Karahasanovic, A. (2001). Identifying Impacts of Database Schema Changes on Application, in *Proc. of the 8th Doctoral Consortium at the CAiSE*01*, (pp. 93-104).

Lockemann, P. C., & Witte, R. (2005). Agents and Databases: Friends or Foes? *Proceedings of IDEAS'05*, (pp. 137-147).

Li, X. (1999). A Survey of Schema Evolution in Object-Oriented Databases. *TOOLS, 31*, 362-371.

Luck, M., McBurney, P., & Preist, C. (2004). A Manifesto for Agent Technology: Towards Next Generation Computing. *Autonomous Agents and Multi-Agent Systems, 9*(3), 203-252.

Magnanelli, M., & Norrie, M. C. (2000). Databases for Agents and Agents for Databases. *Proc. 2nd Intl. Bi-Conference Workshop on Agent-Oriented Information Systems* (AOIS-2000).

Meyer, J-J. Ch. (2004). Intelligent Agents: Issues and Logics. *Logics for Emerging Applications of Databases*, (pp. 131-165).

Roddick, J. F. (1995). A Survey of Schema Versioning Issues for Database Systems, *Information and Software Technology, 37*(7), 383-393.

Tveit, A. (2001). A Survey of Agent-Oriented Software Engineering. *Proceedings of the First NTNU Computer Science Graduate Student Conference*.

van den Akker, J., & Siebes, A. (1997). Enriching Active Databases with Agent Technology. *Proceedings of CIA 1997,* (pp. 116-125).

Wooldridge, M., & Ciancarini, P. (2001). Agent-Oriented Software Engineering: The State of the Art. *LNCS, 1957,* 1-28.

Zambonelli, F., & Omicini, A. (2004). Challenges and Research Directions in Agent-Oriented Software Engineering. *Autonomous Agents and Multi-Agent Systems, 9*(3), 253-283.

KEY TERMS

Agent: Agent is a hardware or software system able to act without human intervention (*autonomy*) both by itself (*pro-activeness*) and reacting to the environment changes (*reactivity*), also interacting with other agents (*social ability*).

Database Refactoring: It indicates little changes in the database schema which preserve both the meaning of the data and the behaviors of the applications. These changes improve the quality of the design.

Evolutionary Data Modeling: Methodologies to iteratively and incrementally model database systems so that schema and applications evolve in a parallel way.

Functional Dependency: Given two sets of attributes *X* and *Y*, a functional dependency between them is denoted by $X \rightarrow Y$. The constraint says that, for any two tuples t_1 and t_2 having $t_1[X] = t_2[X]$, then $t_1[Y] = t_2[Y]$. More precisely, given a table *R*, $X \rightarrow Y \Leftrightarrow \forall t_1, t_2 \in R.(t_1[X] = t_2[X] \Rightarrow t_1[Y] = t_2[Y])$.

Schema Changes: They are operations which allow database schema to evolve according to the changes of the external world.

Schema Level Consistency: A schema is consistent if it is "well-formed (i.e. a schema with some desired properties)." (Li (1999), p.365).

Schema Property: A schema property is a proposition about the constituents of the schema. An example of schema property is the first normal form.

Schema: the database schema is constituted by attributes (fields), tables (of fields), and the relationships between attributes (functional dependencies) and between tables.

ENDNOTES

[a] Open systems were born in the 80s (Hewitt & de Jong, 1982), and are characterized by the fact that their structure is capable of changing dynamically.

[b] Note that the acronym AOSE is also widely used for Aspect-Oriented Software Engineering.

[c] For a detailed description of interrogative agents see Deufemia *et al.* (2007).

[d] In this paper we denote a change request (operation) with the Greek letter ε. The schema resulting by its application to a given schema *S* is ε(*S*).

[e] The queries are transformed using substitutions: in this case an intersection of tables $t' \cap t''$ is substituted to the initial table *t*.

Chapter XIV
Schema Evolution Models and Languages for Multidimensional Data Warehouses

Edgard Benítez-Guerrero
Laboratorio Nacional de Informática Avanzada, Mexico

Ericka-Janet Rechy-Ramírez
Laboratorio Nacional de Informática Avanzada, Mexico

INTRODUCTION

A Data Warehouse (DW) is a collection of historical data, built by gathering and integrating data from several sources, which supports decision-making processes (Inmon, 1992). On-Line Analytical Processing (OLAP) applications provide users with a multidimensional view of the DW and the tools to manipulate it (Codd, 1993). In this view, a DW is seen as a set of dimensions and cubes (Torlone, 2003). A dimension represents a business perspective under which data analysis is performed and organized in a hierarchy of levels that correspond to different ways to group its elements (e.g., the Time dimension is organized as a hierarchy involving days at the lower level and months and years at higher levels). A cube represents factual data on which the analysis is focused and associates measures (e.g., in a store chain, a measure is the quantity of products sold) with coordinates defined over a set of dimension levels (e.g., product, store, and day of sale). Interrogation is then aimed at aggregating measures at various levels. DWs are often implemented using multidimensional or relational DBMSs. Multidimensional systems directly support the multidimensional data model, while a relational implementation typically employs star schemas

(or variations thereof), where a fact table containing the measures references a set of dimension tables.

One important problem that remains open in the Data Warehouse context is schema evolution, the dynamic modification of the schema of a database (Banerjee, Kim, Kim & Korth, 1987, Roddick, 1995). Early works assumed that there was no reason to evolve the schema of a DW. However, practitioners recognized several reasons later (Zurek & Sinnwell, 1999): (i) it is important to support incremental approaches to schema design; (ii) it is necessary to improve the design or to fix errors; and (iii) new user requirements arise. If these changes are not integrated, the DW becomes incomplete and users' information needs are not satisfied anymore.

Schema evolution is a problem that has been mainly studied for relational and object-oriented (OO) databases. However, existing solutions (e.g., Banerjee et al., 1987; McKenzie, & Snodgrass, 1990; Roddick, 1995) are only partially suited for the DW context because they do not consider the multidimensional nature of data. In response to these deficiencies, several works have proposed conceptual evolution models for DWs that consider those features.

The objective of this entry is to introduce the problem of DW schema evolution, explaining current solutions and identifying open problems. It is organized as follows. First, background information covering traditional solutions to the schema evolution problem will be introduced. Then, research on conceptual evolution models and languages will be presented, comparing reference works to others. Open issues will be introduced. Finally, this entry will conclude with a summary.

BACKGROUND

Schema evolution is a subject that has been studied for years. This section briefly introduces traditional works and introduces the problem in the DW context.

Schema Evolution on Relational and Object-Oriented Databases

The problem of schema evolution has been mainly studied for relational and OO databases. In the relational case, few concepts (relations and attributes) are used to describe a database schema. Thus, possible changes are limited to (McKenzie & Snodgrass, 1990) addition/suppression of an attribute or modification of its type, addition/suppression of a relation, and so forth. In the OO context, the model (classes, attributes, methods, is-a relationships) is richer, and then schemas are more complex. Possible changes are (Banerjee et al., 1987) addition, suppression or modification of attributes and methods in a class definition, or changes in superclass/subclass relationships.

Schema Evolution Management

Evolving a schema raises problems at schema, instance, and application levels. At *schema* level, a modification in a part of a schema can cause inconsistencies with other parts. At *instance* level, data can become inconsistent with respect to their schema. At *application* level, applications can become incompatible with the schema after a change. Existing approaches to manage schema evolution try to minimize those problems. These approaches are schema modification, usage of views, and versioning.

Schema Modification

In the schema modification approach, the database schema has only one definition, shared by all applications. So each time the schema needs to be changed, it is updated, and its associated instances are modified accordingly (Penney & Stein, 1987). Therefore, information can be lost if changes imply the removal or modification of

schema elements, generating the incompatibility of old applications.

Views

Some authors (e.g., Bellahsène, 1996) suggest using views to simulate changes. This approach enables schema changes without having to restructure its instances and preserves applications from incompatibilities due to information loss. However, it is not possible to add schema elements (e.g., attributes) unless they are derived from other existing elements.

Versioning

In the schema versioning approach, a schema evolution is represented by the derivation of a new schema version or of a part of it (Roddick, 1995). The main motivation of this approach is to enable applications to access data from several schema versions. Old applications can continue to work after a schema evolution because the old version of the schema always exists, which is crucial when the number of affected applications is large or it is impossible to modify them. Information loss is then avoided, but the number of versions can grow rapidly, complicating their management.

Schema Evolution in the Data Warehouse Context

Schema evolution in the DW context was first tackled by practitioners. Particularly, the addition of new attributes to a table as a way of handling changes to dimensions was proposed in Kimball (1996). This work is important because it proposes a practical solution to a real problem. However, this solution is completely implementation-dependent. The research community saw immediately the need of proposing conceptual evolution models independent of any particular implementation. Existing conceptual DW evolution models are based on the modification and versioning approaches, as explained in the following sections.

SCHEMA MODIFICATION MODELS AND LANGUAGES

This section introduces conceptual evolution models for DWs based on schema modification. First, a reference evolution model (and its associated language) will be introduced, and then existing proposals will be compared with respect to this model.

The WHEM Model

This subsection briefly describes the WareHouse Evolution Model (WHEM) composed by a multidimensional data model and a set of schema evolution operators. For details, see (Benítez-Guerrero, Collet & Adiba, 2003).

Multidimensional Data Model. The main concepts of the multidimensional data model of WHEM are *dimension* and *cube*, and for each of them, the terms *schema* and *instance* are defined. A dimension schema has a name and a set of levels. They represent data domains at different granularities and are organized into a hierarchy by a roll-up relationship. Associated to a level, there is a (possibly empty) set of properties. A dimension instance (or simply, a dimension) is a set of roll-up functions relating values (members) of different levels. A roll-up function associates a member *e* of a level *l* with a member *e'* of a level *l'* such that *l* rolls-up to *l'* in the schema of the dimension. Figure 1(a) shows the schema of the *Product* dimension, which has as levels *code*, *category*, *department*, and a special top level (T). Associated with the *code* level, one can find as property the *name* of the product.

A cube schema has a name, a set of axes (where an axis is a dimension level), and a set of measures. Figure 1(b) shows the schema of the

Sales cube. Its axes are *code* (level of the *Product* dimension) and *day* (level of the *Time* dimension), and its measure is *quantity*. A cube instance (or simply, a cube) is a set of cells (represented as tuples) associating a collection of members, each one belonging to an axis of the cube schema, with one or more measure values.

The collection of cubes and dimensions forms a multidimensional database. Its schema is composed by the collection of schemas of its dimensions and cubes.

Evolution Operators

A set of multidimensional schema evolution operators have been defined. This set includes operators to create dimensions and cubes, to add/delete levels to/from a dimension, to add/delete properties to/from a level, and to add/delete axes and measures to/from a cube. Each of them takes as input and produces as output a multidimensional database. Figure 2(a) shows the addition of the *brand* level to the *Product* dimension (note that now the *code* level rolls up to this level). Figure 2(b) shows the addition of the *city* level (of a *Store* dimension) as an axis of the *Sales* cube.

Associated to this model, a schema evolution language called *WareHouse Evolution Language* (WHEL) has been defined. WHEL is explained next.

The WHEL Language

WHEL provides expressions to define and modify multidimensional schemas according to the set of evolution operators that have been defined.

Figure 3(a) and (b) show the expressions to create the *Product* dimension and the *Sales* cube, respectively. Figure 3(c) shows the expression to add the *brand* level to the schema of the *Product* dimension. Note that the *code* level rolls up to this new level and that the default member of the *brand* level is *Brand1*. Figure 3(d) shows the expression to add the *city* axis to the *Sales* cube. The default member of this axis is *City*1.

Existing Evolution Models and Languages Based on Schema Modification

(Hurtado, Mendelzon, and Vaisman (1999) propose operators to modify and update dimensions

Figure 1. Schemas of (a) the Product dimension and (b) the Sales cube

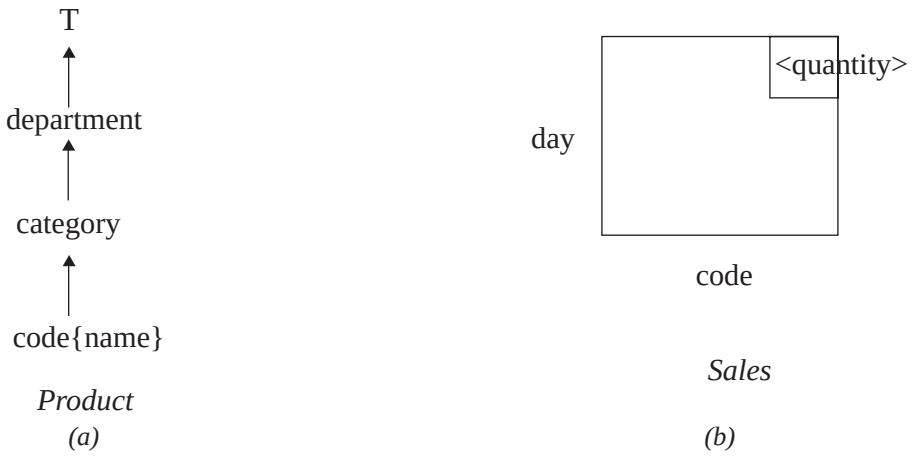
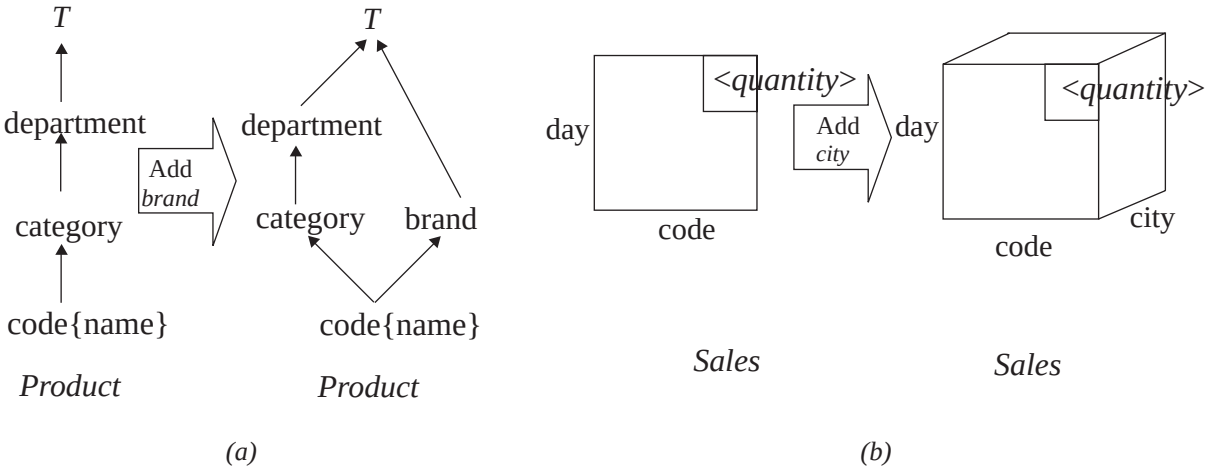


Figure 2 (a). Addition of a level to the Product dimension; (b) Addition of an axis to the Sales cube



at schema and instance levels. At the schema level, they propose operators to add and delete levels (similar to those presented here), and operators to modify roll-up relationships. At the instance level, they propose operators to add, eliminate, split, merge, and update members. This model does not consider cubes evolution. Blaschka, Sapia, and Höfling (1999) also propose a set of schema evolution operators. The difference with the operators presented here is the "granularity" of definition; theirs can be considered as ''low-level'' operators in the sense that each of them adds or removes specific elements (level, property, measure, etc.) of a schema. The problem is that schema consistency may be lost between two successive evolutions so an additional mechanism (transactions) is needed to avoid that problem. The operators described here are "high-level" operators ensuring schema consistency between successive evolutions.

Regarding languages for evolving multidimensional schemas, Hurtado, et al. (1999) propose an algebraic language. Blaschka, et al. (1999) propose a visual language called ME/R, adapting the entity-relationship model to include multidimensional concepts such as dimensions, measures, and so forth. While ME/R can be used in CASE environments, WHEL (as a SQL-like language) can be embedded into data warehousing applications.

VERSION-BASED MODELS AND LANGUAGES

This section introduces conceptual evolution models based on versioning. First, a reference evolution model and its associated language will be introduced. Then existing proposals will be compared with respect to this model.

THE WHEM-V MODEL

This subsection briefly introduces the WareHouse Evolution Model based on versions (WHEM-V) that extend the WHEM model. WHEM-V is based on the notion of the multidimensional database version, and it proposes a set of evolution operators adapted to work with this structure. For details, see Rechy-Ramírez and Benítez-Guerrero (2006).

Multidimensional Database Version

In WHEM-V, the granularity of versioning is the multidimensional database version. It is a DW

Figure 3. Examples of WHEL expressions

```
create dimension Product
level code: char(10)
(property name: char(30))
level category: char(40)
level department: char(40)
rollup code, category
rollup catergory, department
```

(a)

```
create cube Sales
axis code
axis day
measure quantiy: interger
```

(b)

```
alter dimension Product
add level brand: char(20)
   default 'Brand1'
rollup code, brand
```

(c)

```
alter cube Sales
add axis city
default 'City1'
```

(d)

state (a multidimensional schema and its data) that has associated a temporal pertinence, indicating its intervals of valid and transaction times ($[start, end]_{vt}$ and $[start, end]_{tt}$, respectively)[1]. In order to distinguish unequivocally a version, its temporal pertinence needs to be referenced. Consider, for instance the TV_1 version in Figure 4. It contains the schema and an instance of the Product dimension, and its temporal pertinence is $[1, \infty]_{vt}$ and $[0,2]_{tt}$.

Evolution Operators

The schema evolution operators of WHEM were adapted to work on versions. As their nontemporal counterparts, they modify the schema of a multidimensional database and adapt the associated instance. The main difference is that each operator is applied to the current version (the one that has as transaction time the interval $[st, now]_{tt}$, where *st* is the start of the transaction time interval and *now* is the current time) and creates a new version (with its own temporal pertinence) incorporating the change. Consider, for instance, Figure 4. If the *Sales* cube with valid time $[0,\infty]_{vt}$ is added at transaction time *now=3*, a new version TV_2 is generated from TV_1. This new version contains the schemas of the *Product* dimension and the *Sales* cube, and their corresponding instances (particularly a new, empty instance for the *Sales* cube). Its temporal pertinence is $[0, \infty]_{vt}$ and $[3, \infty]_{tt}$.

Associated to the WHEM-V model, the WHEL-V evolution language is defined. It is presented next.

The WHEL-V Language

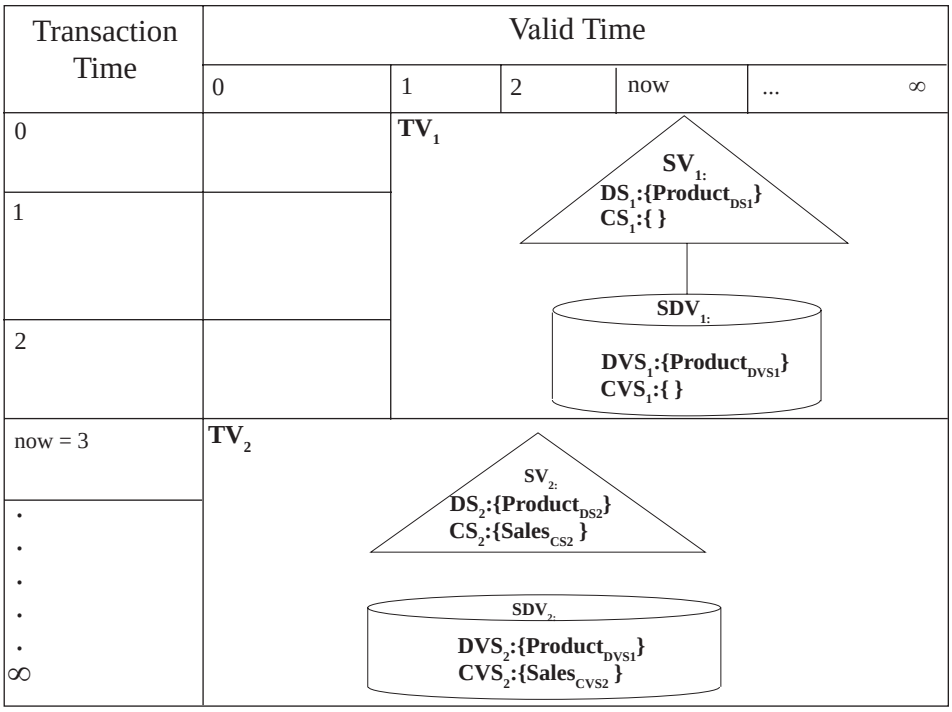
WHEL-V is an adaptation of WHEL to work with versions of multidimensional schemas. The basic expressions of WHEL-V are syntactically the same as in WHEL, but they are applied to the current version *TV*, being unnecessary to specify the transaction time explicitly. In WHEL-V, expressions for creating and modifying versions are introduced.

Figure 5(a) shows a WHEL-V expression to create a new schema version. It is composed by the schemas of the Product dimension and the Sales cube, and its valid time is $[02/05/06, \infty]_{vt}$, while its transaction time is $[now, \infty]_{tt}$. Figure 5(b) shows an expression to modify a version. The result is a new version that contains the schema of the *Product* dimension with the new *brand* level. Its valid time is $[22/05/06, \infty]_{vt}$ while its transaction time is $[now, \infty]_{tt}$.

Existing Evolution Models and Languages Based on Schema Versioning

The COMET model (Eder & Koncilia, 2001) tackles the problem of the evolution of dimension instances. It is based on the concept of *version structure*, which is a dimension instance (members and their roll-up

Figure 4. Multidimensional database versions



relationships) that is valid during a given time interval. Associated to this concept, three operators to insert, update, and delete members (with their corresponding valid time) have been proposed. In COMET, cube evolution is not explicitly considered, as cubes are modeled as degenerated dimensions. COMET is unable to handle two version structures with the same valid time since transaction time is not considered. More recently, in Koncilia (2003), a bitemporal extension of COMET is proposed, but evolution operators are not defined, and the management of the transaction time is only outlined. Associated to this model, there is a visual language that enables the user to create and alter version structures.

The evolution model proposed in Malinowski and Zimányi (2006) distinguishes valid and transaction times of data coming from sources and load time at the DWm which we call transaction time. This model considers temporal and nontemporal levels, members, and roll-up relationships. It is focused on dimension evolution, handling three kinds of changes: changes in the members of a level without affecting the roll-up relationships, changes in the members of a level by affecting the roll-up relationships, and temporal changes in roll-up relationships. Associated to this model is a language that uses a graphical notation to represent levels of a dimension, hierarchies, members, and attributes.

In contrast to other models, WHEM-V uses both transaction and valid times to avoid information loss, allowing the user to have two versions with the same valid time but with different transaction times. In addition, it also considers the evolution of cube schemas.

Figure 5. WHEL-V expressions to (a) create a new schema version and (b) modify a version

```
CREATE TEMPORAL SCHEMA
    create dimension Product
        level article: char(20)
        level brand: char(20)
        level company: char(20)
        rollup article, brand
        rollup brand, company;
    create cube Sales
        axis article
        axis day
        measure quantity: interger;
VALID PERIOD [02/05/06, forever]
```

```
ALTER TEMPORAL SCHEMA
    alter dimension Product
        add level brand: char(20)
        default 'Brand1'
            rollup code, brand
VALID PERIOD [22/05/06, forever]
```

FUTURE TRENDS

Some questions related to the implementation of evolution models and the impact of schema evolution in DW interrogation remain to be answered. The implementation of a conceptual evolution model deals with the particularities of the database model and system chosen to implement the DW. The multidimensional schema is mapped to the implementation one (a relational starlike schema or a DBMS-specific multidimensional schema), and then changes to the multidimensional schema are propagated to its implementation counterpart, adapting the associated data as needed. In the modification approach, the implementation schema is updated, and the stored data are adapted to the new schema. In the version-based approach, the implementation schema (and its data) can be also versioned (Serna-Encinas & Adiba, 2005). Research is still needed to propose efficient implementation methods.

Regarding DW interrogation, the problem that arises in a schema modification setting is that schema elements can be deleted, and then queries over those elements will fail. This problem is not present in the schema versioning approach, but the problem is how to answer cross-version queries (i.e., queries spanning multiple schema versions). In Golfarelli, Lechtenborger, Rizzi, and Vossen (2006), a method to process this kind of queries, based on computing a schema (from the intersection of different schema versions) under which all data involved can be queried uniformly, is proposed. Still, efficient mechanisms for query processing are needed.

CONCLUSION

This chapter presented the two main approaches that have been adopted for conceptual DW schema evolution. In the schema modification approach, there is only one multidimensional schema defined, and whenever a change is needed, this schema is modified and the associated data are adapted. In the version-based approach, the multidimensional schema can have several versions, each one with a corresponding temporal pertinence. Two corresponding evolution models, each one with its own set of evolution operators and its associated evolution language, were explained.

REFERENCES

Banerjee, J., Kim, W., Kim, H.J., & Korth, H.F. (1987). *Semantics and implementation of schema evolution in object-oriented databases.* Proceedings of the SIGMOD'87, San Francisco, California, 311–322.

Bellahsène, Z. (1996). *View mechanism for schema evolution.* Proceedings of the BNCOD-14. LNCS, Edinburgh, UK, 1094, 18–35.

Benítez-Guerrero, E., Collet, C., & Adiba, M. (2003). The WHES approach to data warehouse evolution. *Digital Journal e-Gnosis,* 1665–5745. Retrieved from http://www.e-gnosis.udg.mx

Blaschka, M., Sapia, C., & Höfling, G. (1999). *On schema evolution in multidimensional databases.* Proceedings of the DaWak'99, Florence, Italy, 153–164.

Codd, E. (1993). *Providing OLAP (on-line analytical processing) to users-analysts: An IT mandate* [white paper]. E.F. Codd and Associates.

Eder, J., & Koncilia, C. (2001). *Changes of dimension data in temporal data warehouses.* Proceedings of the DaWak'01, Munich, Germany, 284–293.

Golfarelli, M., Lechtenborger, J., Rizzi, S., & Vossen, G. (2006). Schema versioning in data-warehouses: Enabling cross-version querying via schema augmentation. *Data and Knowledge Engineering, 59*(2), 435–459.

Hurtado, C.A., Mendelzon, A.O., & Vaisman, A.A. (1999). *Updating OLAP dimensions.* Proceedings of the DOLAP'99, Kansas City, Missouri, 60–66.

Inmon, W. (1992). *Building the data warehouse.* Wellesley, Massachusetts: QED Technical Publishing Group.

Kimball, R. (1996). Slowly changing dimensions. *DBMS* and *Internet Systems.*

Koncilia, C. (2003). A bi-temporal data warehouse model [CAiSE'03 short paper]. Proceedings of the CEUR Workshop, 74, 77–80.

Malinowski, E., & Zimányi, E. (2006). *A conceptual solution for representing time in data warehouse dimensions.* Proceedings of the 3rd Asia-Pacific Conference on Conceptual Modeling, Hobart, Australia, 45–54.

McKenzie, L.E., & Snodgrass, R.T. (1990). Schema evolution and the relational algebra. *Information Systems Journal, 15*(2), 207–232.

Penney, D.J., & Stein, J. (1987). *Class modification in the gemstone object-oriented DBMS.* Proceedings of the OOPSLA'87, 111–117.

Rechy-Ramírez, E.-J., & Benítez-Guerrero, E. (2006). *A model and language for bitemporal schema versioning in data warehouses.* Proceedings of the 15th International Conference on Computing (CIC'06), Mexico City, Mexico.

Roddick, J.F. (1995). A survey of schema versioning issues for database systems. *Information and Software Technology, 37*(7), 383–393.

Serna-Encinas, M-T, & Adiba, M. (2005). *Exploiting bitemporal schema versions for managing an historical medical data warehouse: A case study.* Proceedings of the ENC'05, 88–95.

Torlone, R. (2003). Conceptual multidimensional models. In *Multidimensional databases: Problems and solutions* (pp. 69–90). Hershey, PA: Idea Group.

Zurek, T., & Sinnwell, M. (1999). *Data warehousing has more colours than just black and white.* Proceedings of the VLDB'99, Edinburgh, Scotland, 726–728.

KEY TERMS

Cube: Set of cells associating a collection of members, each one belonging to a dimension level, with one or more measure values.

Database Schema: Description of the structure of the database, defined as a collection of data types.

Data Warehouse: Collection of historical data, built by gathering and integrating data from several sources, which supports decision-making processes.

Dimension: A set of values (members) organized in a hierarchy of levels.

Multidimensional Database: A collection of cubes and dimensions.

Multidimensional Database Version: State of a DW with an associated temporal pertinence.

Schema Evolution: The dynamic modification of the schema of a database.

Temporal Pertinence: Element of the Cartesian product of the domains of valid and transaction times.

Transaction Time: Indicates the moment when a fact was stored in the database.

Valid Time: Represents the moment when a fact exists in reality.

ENDNOTE

[1] In Temporal Databases, *valid time* is the time when a fact is true in reality, while *transaction time* is the time when a fact is stored in the database.

Chapter XV
A Survey of Data Warehouse Model Evolution

Cécile Favre
University of Lyon (ERIC Lyon 2), France

Fadila Bentayeb
University of Lyon (ERIC Lyon 2), France

Omar Boussaid
University of Lyon (ERIC Lyon 2), France

INTRODUCTION

A data warehouse allows the integration of heterogeneous data sources for analysis purposes. One of the key points for the success of the data warehousing process is the design of the model according to the available data sources and the analysis needs (Nabli, Soussi, Feki, Ben-Abdallah & Gargouri, 2005).

However, as the business environment evolves, several changes in the content and structure of the underlying data sources may occur. In addition to these changes, analysis needs may also evolve, requiring an adaptation to the existing data warehouse's model.

In this chapter, we provide an overall view of the state of the art in data warehouse model evolution. We present a set of comparison criteria and compare the various works. Moreover, we discuss the future trends in data warehouse model evolution.

BACKGROUND

Schema and Data Evolution in Data Warehouses: The Coherence Problem

The main objective of a data warehouse is to provide an analysis support for decision-making.

The analysis possibilities of a data warehouse mainly depend on its schema. The analysis results depend on the data. Following the evolution of sources and analysis needs, the data warehouse can undergo evolution on the level of its schema and its data at the same time.

From the schema evolution point of view, the following evolutions can be envisaged:

- dimension (adding/deletion)
- measure (adding/deletion)
- hierarchy structural updating (level adding/deletion)

These evolutions enrich or deteriorate the analysis possibilities of data warehouses. However, they do not induce erroneous analysis as evolution of the data does.

In regard to data evolution, we have identified three operations: insertion, deletion, and updating of data in the data warehouse. These operations can be performed on either the fact table or the dimension tables, and depending on the case, do not have the same impact on analysis coherence. The insertion (in the fact table or in the dimension tables) corresponds to the usual data-loading process of the data warehouse.

However, since data warehouses contain historical and nonvolatile data, records should not be updated or deleted. However, as Rizzi and Golfarelli (2006) point out, updates in the fact table could be required in order to correct errors or to reflect the evolution of the events.

Furthermore, the usual assumption in data warehouse modeling is the independency of the dimensions. Thus, defining a dimension to characterize time induces that other dimensions are independent of the time dimension. In other words, these dimensions are supposed to be time-invariant. However, this case is extremely rare. Thus, in order to ensure correct analysis, these dimensions have to evolve in a consistent way (Letz, Henn & Vossen, 2002).

Kimball (1996) introduced three types of "slowly changing dimensions" that consist in three possible ways of handling changes in dimensions. The basic hypothesis is that an identifier cannot change, but the descriptors can. The first way consists of updating the value of the attribute. In this case, the historization of changes is not available. Thus, this solution has consequences on analysis coherence only if this updated attribute is used to carry out the analysis. The second type allows keeping all the versions of the attribute's value by creating another record valid for a time period. The drawback of this approach is the loss of comparisons throughout versions. This is due to the fact that the links between evolutions are not kept even if evolutions are preserved. The last type consists of creating another descriptor to keep track of the old value in the same record. Thus, we keep the link between the two versions. However, if there are several evolutions, there is a problem to consider the different versions with changes on several attributes that do not occur at the same time.

As Body, Miquel, Bédard, and Tchounikine (2002) summed it up; the study of Kimball takes into account most users' needs and points out the necessity of keeping track of both history and links between transitions. Indeed, the main objective of a data warehouse is to support correct analysis in the course of time and ensure good decisions.

This objective mainly depends on the capacity of the data warehouse to be a mirror of reality. From our point of view, the model evolution problem must not be separated from the problem of analysis coherence. Thus, we think we have to identify when the evolution induces incoherence of analysis. Data historization and, more precisely, dimension historization are required for descriptors that are involved in the analysis process. Note that for analysis purposes, it is necessary to be able to translate facts by getting data in a consistent time.

In order to take into account these evolutions, we can distinguish in the literature two types of approaches: model updating and temporal mod-

eling. In the next section, we will present works falling in these two categories.

Model Updating

Concerning model updating, only one model is supported, and the trace of the evolutions is not preserved. The evolutions are thus applied to define a new model. We distinguish three types of approaches.

The first stream (labeled "evolution operators") consists of providing formal evolution operators that allow an evolution of the model. Hurtado, Mendelzon, and Vaisman (1999) propose a formal model and operators that allow the updating of dimensions and their hierarchies, such as adding a granularity level at the end of a hierarchy. They also study the effect of these updates on the materialized views and propose a way to maintain them. Blaschka, Sapia, and Höfling (1999) propose the evolution not only of dimension, but also of facts. They propose an algebra with different operators that can be combined to allow complex evolutions on a conceptual level. Benitez-Guerrero, Collet, and Adiba (2004) exploit these works to propose a data warehouse framework to allow the creation and evolution of the schema, independently of the storage method (i.e., relational, etc.).

The second stream (labeled "hierarchies enrichment") is inspired by the previous works but focuses on the creation of a new granularity level in a dimension hierarchy. The objective is to focus on how to create this level and not on how to represent this operation. The two approaches we present consist of updating the model without questioning the coherence of the existing data. Mazón and Trujillo (2006) propose to automatically enrich dimension hierarchies. The hypothesis is that a dimension hierarchy corresponds to a set of semantic relations between values. Then they propose to use hyperonym ("is-a-kind-of") and meronym ("is-a-part-of") relations of WordNet (Fellbaum, 1998) to build new granularity levels at the end of hierarchies. The approach we proposed allows enriching dimension hierarchies according to users' knowledge, providing analysis personalization (Favre, Bentayeb & Boussaïd, 2007). This knowledge is represented on one hand by an aggregation metarule that represents the structure of the aggregation link, and on the other hand by if-then rules that represent links between instances. The created granularity levels can be added at the end of hierarchies or inserted.

The last stream (labeled "view maintenance") is based on the hypothesis that a data warehouse is a set of materialized views defined on data sources (Bellahsene, 2002). Hurtado, et al. (1999) were interested in materialized views maintenance to propagate data warehouse model evolution on data cubes that are under the form of views. On the contrary, Bellahsene (2002) is interested in propagating data sources evolution on the data warehouse model that is represented by views, in a relational context. To take into account analysis needs evolution, it is possible to add new attributes.

Temporal Modeling

Contrary to model updating, temporal modeling makes it possible to keep track of the evolutions by using temporal validity labels.

The first stream (labeled "temporal instances") consists of using temporal validity labels for dimension members (Bliujute, Saltenis, Slivinskas & Jensen, 1998); that is, a temporal star schema is proposed there to represent the temporal validity of facts and dimension members. The data are represented in a consistent time. In this approach, time is not a separate, independent dimension table but rather a dimension of all tables represented via one or more time-valued attributes in all tables.

The second stream (labeled "temporal aggregation links") proposes to manage the temporality of aggregation links within dimension hierarchies. Thus, Mendelzon and Vaisman (2000) propose a multidimensional model in which an aggregation path within dimension hierarchies

can evolve. They also propose a language that supports this model, allowing temporal queries; namely, TOLAP.

The last stream (labeled "temporal versions") is the versioning. It consists of managing several versions of the same data warehouse. There are many works in this field. For space limitations, we present only a part of them.

From the model point of view, we can cite the model proposed by Eder and Koncilia (2000) that allows the use of mapping functions between different versions. These functions are based on the knowledge about the evolutions applied. Then Eder, Koncilia, and Morzy (2002) propose a metamodel named COMET to manage temporal data warehouses. More recently, Ravat, Teste, and Zurfluh (2006) propose a multidimensional model in consistent time, characterized by the fact that it allows evolutions on a constellation model.

From the analysis point of view, Body, et al. (2002, 2003) propose an approach that allows users to get analysis according to their needs. Indeed, users can choose in which version to carry out the analysis (in consistent time, in the old version, in the current version). Versions are also used to provide an answer to "what-if analysis" by creating alternative versions to simulate changes (Bebel, Eder, Koncilia, Morzy & Wrembel, 2004). Furthermore, various works stress analyzing data throughout versions in order to achieve the main objective of data warehousing: analyzing data in the course of time, as in Morzy and Wrembel (2004) and Golfarelli, Lechtenborger, Rizzi, and Vossen (2006).

In all the works following temporal modeling, an extension to a traditional SQL language is required to take into account the particularities of the approaches for analysis or data loading. These approaches have to be deployed during the design step.

DISCUSSION

This section deals with the comparison between existing works on data warehouse model evolution.

Comparison Criteria

We have defined three groups of criteria: functionalities offered, deployment criteria, and performance criteria.

The criteria corresponding to the functionalities offered by the approaches are:

- Dimension historization
- Analysis coherence
- User-driven approach

These criteria tell us whether an approach allows for time invariant dimensions, whether the analysis made after the schema evolution is coherent, and whether the approach is user-centric in regard to the decision process. More precisely, this last point is linked to personalization. Data warehouse personalization is a relatively new approach in the data warehouse community. These works concern data visualization based on user preferences and profile. For example, Bellatreche, Giacometti, Marcel, Mouloudi, and Laurent (2005) refine user queries based on user preferences to show only relevant data.

The criteria about the deployment of approaches are:

- The need for deploying the solution during the design phase
- The simplicity of deployment

The need for deploying the solution during the design phase considers whether the approach has to be applied during the design phase of the data warehouse. The simplicity of the deployment measures whether the existing tools have to be adapted.

Finally, the performance criteria are:

- The volume of stored data
- The time for answering analysis queries

Data warehouses are meant for online analysis; therefore, performance is a crucial aspect in regard to response time and storage capacity.

Comparative Study

Table 1 summarizes a comparison of the various approaches described previously according to the selected criteria. The + (- respectively) represents whether the approach satisfies (or does not satisfy, respectively) the criterion.

For the dimension historization criteria, we find that the approaches based on temporal modeling ensure the data historization, whereas those based on model updating do not satisfy the criteria. Nevertheless, granularity level addition does not have any impact on analysis coherence. Therefore, approaches aiming at hierarchical enrichment and those based on temporal modelization do not have any impact on analysis coherence.

The studied approaches deals with user implication in different ways. For instance, Favre, et al. (2007) consider that model evolution is triggered by the user, while others consider that the evolution is triggered by the data warehouse administrator. The temporal modeling is not user-centred, except for some works following a version approach that allow users to choose the data warehouse version used for the analysis.

The deployment of temporal modeling approaches may be complex due to the need to plan their deployment during the design phase and their need of specific tools for data loading and data analysis. Therefore, the deployment of such an approach may be very difficult.

Regarding performance, temporal approaches are storage-space intensive. Indeed, they need extra storage space for the storing of temporal labels, versions, and metadata. Moreover, response

Table 1.Comparative study of approaches for data warehouse model evolution

		Model Updating			**Temporal Modeling**		
		Evolution Operators	Hierarchies Enrichment	View Maintenance	Temporal Instances	Temporal Aggregation Links	Temporal Versions
Characteristics	Dimension Historization	-	-	-	+	+	+
	Analysis Coherence	-	+	-	+	+	+
	User-Driven Approach	-	+	-	-	-	+ / -
Deployment	During Conception	+	+	+	-	-	-
	Simplicity	+	+	+	-	-	-
Performances	Storage	+	+	+	-	-	-
	Response Time	+	+	+	-	-	-

time is longer, and query rewriting is needed to take into account different versions of the data warehouse. These performance criteria are rarely discussed, except by Bliujute, et al. (1998).

FUTURE TRENDS

Different future trends are envisaged. However, we detail here the one that seems to be the most important. The stake is to get a general framework to manage the data warehouse evolution. Our idea is to extend the workflow paradigm of Bouzeghoub, Fabret, and Matulovic-Broqué (1999), who only deal with the data warehouse refreshment. Our idea is to consider the whole evolution of the data warehouse as a workflow, in which the start point would be the data sources evolution or the analysis needs evolution. Moreover, by data warehouse evolution, we mean not only the propagation of evolutions on the model but also the propagation on the different components of the data warehouse (e.g., datamarts, materialized views, indexes, etc.).

CONCLUSION

To conclude, we would like to point out the fact that the use of temporal approaches in practice is not generalized, but the research field is very active through the works on versioning. Some commercial systems already allow tracking changes in data and effectively querying data according to different temporal scenarios. For instance, SAP-BW allows the user to choose the version of the hierarchies for an analysis (SAP, 2000). However, schema versioning has only partially been explored, and no dedicated commercial tools are available to the designer. Moreover, as we mentioned earlier, the deployment of temporal approaches needs to be considered at the design stage, and they are not applicable to already deployed data warehouses. Finally, it seems important to apply these research approaches to the business field since data warehouses emerged and became widely accepted in most companies.

REFERENCES

Bebel, B., Eder, J., Koncilia, C., Morzy, T., & Wrembel, R. (2004). *Creation and management of versions in multiversion data warehouse*. Proceedings of the 19th ACM Symposium on Applied Computing (SAC 04), Nicosia, Cyprus, 717–723.

Bellahsene, Z. (2002). Schema evolution in data warehouses. *Knowledge and Information Systems, 4*(3), 283–304.

Bellatreche, L., Giacometti, A., Marcel, P., Mouloudi, H., & Laurent, D. (2005). *A personalization framework for OLAP queries.* Proceedings of the 8th ACM International Workshop on Data Warehousing and OLAP (DOLAP 05), Bremen, Germany, 9–18.

Benitez-Guerrero, E.I., Collet, C., & Adiba, M. (2004). The WHES approach to data warehouse evolution. *Electronic Journal e-Gnosis, 2.*

Blaschka, M., Sapia, C., & Höfling, G. (1999). *On schema evolution in multidimensional Databases.* Proceedings of the 1st International Conference on Data Warehousing and Knowledge Discovery (DaWaK 99), Florence, Italy, 1676, 153–164.

Bliujute, R., Saltenis, S., Slivinskas, G., & Jensen, C. (1998). *Systematic change management in dimensional data warehousing.* Proceedings of the 3rd International Baltic Workshop on Databases and Information Systems, Riga, Latvia, 27–41.

Body, M., Miquel, M., Bédard, Y., & Tchounikine, A. (2002). *A multidimensional and multiversion structure for OLAP applications.* Proceedings of the 5th ACM International Workshop on Data Warehousing and OLAP (DOLAP 02), McLean, Virginia, 1–6.

Body, M., Miquel, M., Bédard, Y., & Tchounikine, A. (2003). *Handling evolutions in multidimensional structures.* Proceedings of the 19th International Conference on Data Engineering (ICDE 03), Bangalore, India, 581–591.

Bouzeghoub, M., Fabret, F., & Matulovic-Broqué, M. (1999). *Modeling the data warehouse refreshment process as a workflow application*. Proceedings of the International Workshop on Design and Management of Data Warehouses (DMDW 99), Heidelberg, Germany, 19, 6.

Eder, J., & Koncilia, C. (2000). *Evolution of dimension data in temporal data warehouses* [technical report]. Austria: University of Klagenfurt, Austria.

Eder, J., Koncilia, C., & Morzy, T. (2002). *The COMET metamodel for temporal data warehouses.* Proceedings of the 14th International Conference on Advanced Information Systems Engineering (CAiSE 02), Toronto, Canada, 2348, 83–99.

Favre, C., Bentayeb, F., & Boussaïd, O. (2007). *Dimension hierarchies updates in data warehouses: A user-driven approach*. Proceedings of the 9th International Conference on Enterprise Information Systems (ICEIS 07), Funchal, Madeira, Portugal.

Fellbaum, C. (1998). *WordNet: An electronic lexical database (language, speech, and communication).* The MIT Press.

Golfarelli, M., Lechtenborger, J., Rizzi, S., & Vossen, G. (2006). Schema versioning in data-warehouses: Enabling cross-version querying via schema augmentation. *Data and Knowledge Engineering, 59*(2), 435–459.

Hurtado, C.A., Mendelzon, A.O., & Vaisman, A.A. (1999). *Maintaining data cubes under dimension updates.* Proceedings of the 15th International Conference on Data Engineering (ICDE 99), Sydney, Australia, 346–355.

Kimball, R. (1996). *The data warehouse toolkit.* John Wiley & Sons.

Letz, C., Henn, E.T., & Vossen, G. (2002). *Consistency in data warehouse dimensions.* Proceedings of the International Symposium on Database Engineering & Applications (IDEAS 02), Edmonton, Canada, 224–232.

Mazón, J.N., & Trujillo, J. (2006). *Enriching datawarehouse dimension hierarchies by using semantic relations.* Proceedings of the 23rd British National Conference on Databases (BNCOD 06), Belfast, Northern Ireland, 4042, 278–281.

Mendelzon, A.O., & Vaisman, A.A. (2000). *Temporal queries in OLAP.* Proceedings of the 26th International Conference on Very Large Data Bases (VLDB 00), Cairo, Egypt, 242–253.

Morzy, T., & Wrembel, R. (2004). *On querying versions of multiversion data warehouse*. Proceedings of the 7th ACM International Workshop on Data Warehousing and OLAP (DOLAP 04), Washington, DC, 92–101.

Nabli, A., Soussi, A., Feki, J., Ben-Abdallah, H., & Gargouri, F. (2005). *Towards an automatic data mart design*. Proceedings of the 7th International Conference on Enterprise Information Systems (ICEIS 05), Miami, Florida, 226–231.

Ravat, F., Teste, O., & Zurfluh, G. (2006). *A multiversion-based multidimensional model.* Proceedings of the 8th International Conference on Data Warehousing and Knowledge Discovery (DaWaK06), Krakow, Poland, 4081, 65–74.

Rizzi, S., & Golfarelli, M. (2006). *What time is it in the data warehouse?* Proceedings of the 8th International Conference on Data Warehousing and Knowledge Discovery (DaWaK 06), Krakow, Poland, 4081, 134–144.

SAP. (2000). *Multi-dimensional modelling with BW: ASAP for BW accelerator* [technical report]. SAP America Inc. and SAP AG.

KEY TERMS

Analysis in a Consistent Time: Results are provided by taking into account the moment when a fact exists in the reality.

Data Warehouse: Collection of historical data, built by gathering and integrating data from several data sources, structured in a multidimensional way to support decisional queries.

Data Warehouse Model: The data warehouse model includes its schema and its data.

Data Warehouse Schema: Designs the structuration of the data in the data warehouse; measures representing facts are analyzed according to dimensions.

Model Updating: Making the same model evolve without keeping track of its evolution history; thus, the model corresponds to its current version.

Model Versioning: Building several versions of a model where each new version corresponds to a schema evolution or an evolution of data valid for a given period.

Temporal Modeling: Providing temporal extensions to keep track of the model history.

Temporal Validity Label: A temporal validity label corresponds to a timestamp used to denote the valid time.

Chapter XVI
Document Versioning and XML in Digital Libraries

M. Mercedes Martínez-González
Universidad de Valladolid, Spain

INTRODUCTION

Digital libraries are systems that contain organized collections of objects, serving in their most basic functions as a mirror of the traditional library that contains paper documents. Most of the information contained in the collections of a digital library consists of documents, which can evolve with time. That is, a document can be modified to obtain a new document, and digital library users may want access to any of those versions. This introduces in digital libraries the problem of versioning, a problem that is also of interest for the hypertext community and the Semantic Web community. Some domains in which document evolution is a very important issue are the legislative domain (Arnold-Moore, 1997; Martínez González, de la Fuente, Derniame & Pedrero, 2003a; Vitali, 1999), the management of errata made to scientific articles (Poworotznek, 2003), software construction (Conradi & Westfechtel, 1998), and collaborative e-learning (Brooks, Cooke & Vassileva, 2003).

In the legislative domain, rules suffer amendments that result in new versions of the amended rules. Access to all versions of a document is an important facility for their users; for example, to understand a tribunal sentence it is necessary to get access to the text of involved rules, as they were valid at the moment the sentence was made. Errata to scientific articles are somewhat similar. The errata are posterior to the original article and they are published together with the reference to the part of the article to be changed by the modification. In software construction, different versions of program files are available at the same time, and the composition of software has to assemble adequate versions in order to obtain the correct version of the software. In e-learning frameworks, the problem comes from the updates

made to content objects, or the reordering of these contents.

In recent years, the spread of XML as the metalanguage for document modelling has been accompanied by a strong interest in XML document versioning. The interesting issue is that XML documents are no longer considered as atomic items that can be substituted or not, but composed of document nodes (elements) that can themselves be versioned. Besides, there have been several initiatives that propose using XML as the ideal format to represent metadata related with changes.

Next, we revise the issues related with document versioning, the main approaches proposed and the issues that each approach favours. Issues related with XML will receive special attention in this updated chapter[1]. Versioning a document impacts not only the document itself but also other items, such as references from and to the versioned document, or the indexes created for information retrieval operations.

BACKGROUND

As for the issues of interest related to document versions, we distinguish seven categories:

1. What can be versioned?

This question can be considered from two perspectives. The first perspective considers objects stored in the system as atomic units of information, which cannot suffer partial changes. This is the typical situation in the Web and hypertext environments. Hypertext nodes (documents, files, others) can change (be substituted, deleted, inserted), and the hypertext structure can also change (objects may vary their location, some of them may disappear, others may change their references to other objects), but each document is considered an atomic item which is not subdivided in other objects: changes always concern the whole document. The evolution considered in the second perspective is the one of the documents used by digital library users --these documents may or may not match unidirectionally any of the objects stored in the digital library (Arms, 1997)—and with XML documents. Changes in this case can be related with any component of a document: its content, part of it (e.g., some nodes in XML documents), the internal structure of documents, or references (citations within documents, that are part of a document).

2. Detecting changes

Sometimes it is necessary to recognise two documents as versions of the same work, or to find the changes that have to be applied to a document version to obtain another one. There are two possible ways to do this: extracting references from document content (Thistlewaite, 1997; Martínez, de la Fuente, Derniame & Pedrero, 2003a), or comparing versions (Chawathe, Rajaraman, García-Molina & Widow, 1996; Lim & Ng, 2001; Cobena, Abiteboul & Marian, 2002).

3. Representing changes

The information about versions and the changes between them has to be stored somehow in the system. This is dealt with in the version control model used, or in the data model, as happens with XML documents. Besides, metadata that describe the items can also be used to represent the versioning information. In summary, the main possibilities are:

- To store the versions caused by a change, and/or the corresponding differences (deltas). This is the classical approach in version management, and corresponds to solution 1 of versioning management approaches presented in the next section.
- To represent changes as annotations (attributes) to the versioned items. In this case

the history of an item is described in its annotations. This is solution 3 for version management.

- To stamp the items changed with some mark that indicates its time of validity (timestamp) or version validity (versionstamp). This temporal validity attribute is used during the reconstruction of versions to retrieve the valid nodes of XML documents (Arévalo, Polo & Fernández, 2006; Chien, Tsotras, Zaniolo & Zhang, 2006; Grandi, Mandreoli & Tiberio, 2005).
- To model modification relationships as links between the modifiers and the target of the modification (Martínez, 2003a). This solution considers the semantic relationship that is behind a change.

4. Querying changes

This consists of answering questions about a document evolution, such as ''What are the documents that modify this one?'' or ''What are the modifications of this document over the last month?''. The way to operate here is dependent on the choice made for representing changes. Besides, there are some proposals that consider how query languages can support querying temporal information. In some cases, the proposals try to extend existing standards to support querying multiversion documents (Chien, 2006; Norvag, 2005).

5. Access to any version

The access to any version can be provided either by storing all of them or by composing them on demand. It depends on the approach chosen for version management. It can also be seen as a query in which only the valid items of a document make part of the result set that will be used in the composition of the requested version.

6. Dealing with the propagation (effects) of versioning some items on related items.

Changes to a document can affect other information items, such as references or links that reach the document. This is the well-known problem of dangling links, so common in the Web, or, in a more general definition, the 'referential integrity' issue (Ashman, 2000). Other possible impacts are on indexes used for information retrieval operations.

7. Inferring the composition rules of new versions

In certain situations (for example, the legislative domain) the new versions are virtual (no copy or direct way to obtain it is provided), and the structure of the new version has to be inferred from the information provided about modifications. Humans commonly assume this task, but an automatic inference can also be tackled (Martínez, de la Fuente & Derniame, 2003b).

8. Validating updates

This issue appears in XML document versioning, as XML documents usually conform to some schema that describes the structure rules that the logical structure of an XML document must follow. In such cases, it is possible that some change (addition of new nodes, reordering, ...) makes the new version invalid. Hence, the goal is to avoid such 'no valid' changes (Kane & Rundernsteiner, 2002).

APPROACHES TO VERSION MANAGEMENT

Different approaches can be distinguished. Here, they are listed ordered temporally: the ones with a longer tradition appear first and the more recent ones are at the end of the list.

1. Maintaining simultaneously all versions in digital library collections and linking related versions, or storing one version (the first one,

or the more recent one) and the deltas that permit other versions of the same document to be obtained. This approach facilitates access to any version, but does not consider queries about the evolution of versioned items. The propagation of versioning effects is considered, but there are no general solutions and this is still a difficult issue to deal with. This approach and variations of it have received a good amount of attention in the version control area (Hicks, Legget, Nürberg & Schnase, 1998; Chien, 2006). It is also used with XML documents (Marian, Abiteboul, Cobéna & Mignet, 2001).

2. To consider different stamps of the database and to compare them in order to detect changes that reflect the fact that an object has been versioned (Cellary & Jomier, 1992). This solution is used with object databases, and therefore can be considered when modelling documents as objects (Abiteboul, Cluet, Christophides, Milo, Moerkotte & Simeon, 1997). In this approach changes are represented indirectly as the difference between two database states.
3. Modelling modifications as attributes and storing this information with documents. This approach comes from the area of semi-structured data, which includes structured documents and XML documents. Changes are represented as annotations (attributes) to the affected nodes, facilitating queries about nodes' 'history'. The detection of versions is done by tree comparisons (Chawathe, 1996; Cobena, 2002). The document structure is considered, thereby associating changes to document fragments instead of to whole documents. It is possible to know the document has been changed and where to find the changes; however, it is up to the user to obtain the versions if this is his/her wish. For the same reason, it does not facilitate the automatic composition of versions.
4. Automatically composing versions of documents. This can be done by following the rules that allow the generation of versions of documents. This is the option named *intentional versioning* (Conradi & Westfechtel, 1998) and has been used for document management (Arnold-Moore, 1997). It is also possible to compose versions by querying meta-data (attributes, links) stored in the system databases (Hemrich, 2002; Martínez, 2003a). Another possibility is to stamp (XML) document nodes with validity attributes that permit one to know what the valid nodes for each version are (Arevalo, 2006; Chien, 2006; Grandi, 2005; Kane, 2002; Iksal & Garlatti, 2002). These solutions deal well with access to versions and they are in a good position to treat queries about version evolution. Their main weakness is in dealing with the propagation of versioning effects on information retrieval operations.

XML DOCUMENT VERSIONING

The spread of XML has renewed the interest in document versioning. The novelty is that documents are no longer considered atomic items, but items composed of nodes that are organised through ordered inclusions which produce a document structure called 'logical structure'.

Versioning XML documents produces what is sometimes called 'multiversion' XML documents, that is, XML documents in which a new dimension appears: each node can have several branches that correspond to its different versions, so that each branch should be used only to compose the document version in which it is valid. Normally, each node version is stamped with some validity attribute that is used during version reconstruction (Arévalo, 2006; Chien, 2006; Grandi, 2005).

However, there are also solutions that adapt the classical version control models to represent changes in XML document warehouses (Marian, 2001).

MANNERS OF IMPLEMENTING VERSIONING SOLUTIONS

The manners to implement solutions to manipulate versions have evolved with time. Version control servers, which provide general mechanisms for version control issues, are the first generalized solution (Hicks, 1998). As for the manner to represent and store the versioning information, some solutions using HTML elements (Vitali, 1999) for modelling changes as attributes were proposed. They were later superseded by solutions based on XML. If the documents versioned are themselves XML, the data models are extended to include XML attributes that model XML nodes' validity (Arévalo, 2006; Chien, 2006; Grandi, 2005; Kane, 2002; Iksal, 2002).

In fact XML supposed a change in the manners of implementing versioning. Solutions based on automatically composing versions appeared with this standard, as it facilitates automatic document composition: document markup and standards such as XSLT and XQuery facilitate it (Arnold-Moore, 1997; Chien, 2006; Grandi, 2005; Hemrich, 2002; Iksal, 2002; Martínez-González, 2001; Norvag, 2002).

XML is also been used to store metadata about document evolution (Brooks, 2003; Wang, 2006). In (Kitchaorensakkul, 2001) they use RDF implemented on top of XML. XML has also opened up the possibility of validating updates against schemas (Grün, Karlinger & Schrefl, 2006; Kane, 2002). XSLT and XQuery are the two standards that support the dynamic part of these XML based implementations.

FUTURE TRENDS

It is clear that XML is the standard that will support most versioning solutions in digital libraries in the future. Querying changes, treating change information at a finer granularity level than document level (nodes), and automatically composing versions are already possible. Nevertheless, work will continue to improve these solutions. XML query optimization and efficiency is an open research area that can provide benefits to document versioning in digital libraries.

Now that solutions for control version are well advanced, digital libraries would be great beneficiaries of solutions that permit the semantics of versioning to be modelled and queried. Of course, this issue is highly related with the Semantic Web. Solutions for this type of issue would be a step forward in permitting digital libraries to offer semantic information (in addition to documents and objects) to their users.

CONCLUSION

Document evolution demands solutions to manipulate versions and to satisfy user requests from digital libraries. Several issues emerge related with versioning: accessing any version of a document, querying its history, managing the impact of versioning an item on related items, etc. The background on dealing with these problems is varied. The version control community has studied issues such as access to any version for a long time. However, this study area does not consider other issues such as querying the history of a document, or the impact of versioning on information retrieval operations such as indexing.

The approaches that compose versions automatically and infer the composition rules (structure) of versions from semantic information are implemented on top of XML. These solutions introduce dynamism and finer granularity in version management. Future improvements in

digital libraries should introduce semantic querying of changes.

REFERENCES

Abiteboul, S., Cluet, S., Christophides, V., Milo, T., Moerkotte, G., & Simeon, J. (1997). Querying documents in object databases. *International Journal on Digital Libraries, 1*(1), 5-19.

Arévalo, J. L., Polo, A., & Fernández, J. M. (2006). Representing Versions in XML documents using versionstamp. In *ER (Workshops) 2006, Lecture Notes in Computer Science, 4231*, 257-267.

Arms, W. Y., Blanchi, C., & Overly, E. A. (1997). An Architecture for Information in Digital Libraries. *D-Lib Magazine*, Feb 1997.

Arnold-Moore, T. (1997). Automatic Generation of Amendment Legislation. *In Sixth International Conference on Artificial Intelligence and Law, ICAIL'97* (Melbourne, Victoria, Australia, 1997), (pp. 56-62).

Ashman, H. (2000). Electronic document addressing: dealing with change. *ACM Computing Surveys, 32*(3), 201-212.

Brooks, C., Cooke, J., & Vassileva, J. (2003). *In Proceedings of the 3rd. IEEE International Conference on Advanced Learning Technologies (ICALT'03)* (pp. 296-297). Athens, Greece: IEEE Computer Society.

Cellary, W., & Jomier, G. (1992). *Building an object-oriented database system. The story of O_2.*, chapter Consistency of Versions in Object-Oriented Databases. Number 19 in The Morjgan Kaufmann Series in Data Management Systems. Morgan Kaufmann (pp. 447-462).

Chawathe, S., Rajaraman, A., Garcia-Molina, H., & Widom, J. (1996). Change detection in hierarchically structured information. *SIGMOD Record (ACM Special Interest Group on Management of Data), 25*(2), 493-504.

Chien, S.Y., Tsotras, V. J., Zaniolo, C., & Zhang, D. (2006). Supporting complex queries on multiversion XML documents. *ACM Transactions on Internet Technology, 6*(1), 53-84.

Cobena, G., Abiteboul, S., & M. A. (2002). Detecting Changes in XML Documents. *In Data Engineering 2002 (ICDE2002),* (pp. 41-52).

Conradi, R., & Westfechtel, B. (1998). Version models for software configuration management. *ACM Computing Surveys, 30*(2), 232-282.

Grandi, F., Mandreoli, F., & Tiberio, P. (2005). *Data and Knowledge Engineering, 54*(3), 327-354.

Grün, K., Karlinger, M., & Schrefl, M. (2006). Schema-aware labelling of XML documents for efficient query and update processing in SemCrypt. *International Journal of Computer Systems, Science, and Engineering, 21*(1), 65-82.

Hemrich, M. (2002). A New Face for Each Show: Make Up Your Content by Effective Variants Engineering. In *XML Europe 2002.* Available at http://www.idealliance.org/papers/xmle02/ (verified on September 7, 2004).

Iksal, S., & Garlatti, S. (2002). Revisiting and Versioning in Virtual Special Reports. *Lecture Notes in Computer Science, 2266*, 264-279.

Hicks, D. L., Leggett, J. J., Nürnberg, P. J., & Schnase, J. L. (1998). A Hypermedia Version Control Framework. *ACM Transactions on Information Systems, 16*(2), 127-160.

Kane, B., Su, H., & Rundernsteiner, A. E. (2002). Consistently updating XML documents using incremental constraint check queries. In Fourth ACM CIKM International *Workshop on Web Information and Data Management (WIDM'02)* (pp. 1-8). Virginia, USA: ACM.

Lim, S. J., & Ng, Y. K. (2001). An Automated Change-Detection Algorithm for HTML Documents Based on Semantic Hierarchies. In *The 17th International Conference on Data Engi-*

neering (ICDE 2001), (pp. 303-312), Heidelberg, Germany.

Marian, A., Abiteboul, S., Cobéna, G., & Mignet, L. (2001). Change-centric management of versions in an XML warehouse. *In 27th International Conference on Very Large Databases (VLDB'01)* (pp. 581-590). Roma, Italy: Morgan Kauffman.

Martínez González, M., de la Fuente, P., Derniame, J., & Pedrero, A. (2003a). Relationship-based dynamic versioning of evolving legal documents. In *Web-knowledge Management and Decision Support*, volume 2543 of *Lecture Notes on Artificial Intelligence* (pp. 298—314). Springer-Verlag.

Martínez González, M., de la Fuente, P., & Derniame, J.-C. (2003b). XML as a means to support information extraction from legal documents. *International Journal of Computer Systems Science and Engineering, 18*(5), 263-277.

Norvag, K. (2005). Query operators in XML databases. In L. C. Rivero, J. H. Doorn, & V. E. Ferraggine (Ed.), *Encyclopedia of Database Technologies and Applications 2005* (pp. 500-505). Idea Group.

Poworotznek, E. (2003). Linking of errata: current practices in online physical sciences journals. *Journal of the American Society for Information Science and Technology, 54*(12), 1153-1159.

Thistlewaite, P. (1997). Automatic Construction and Management of Large Open Webs. *Information Processing and Management, 33*(2), 161-173.

Vitali, F. (1999). Versioning hypermedia. *ACM Computing Surveys, 31*(4es), 24.

Wang, F., Zhou, X., & Zaniolo, C. (2006). Bridging relational database history and the web: the XML approach. *In Eight ACM International Workshop on Web Information and Data Management (WIDM'06)* (pp. 3-10). Arlington, Virginia, USA: ACM.

KEY TERMS

Digital Library: A set of electronic documents organized in collections, plus the system that provides access to them. They are the digital version of traditional libraries.

Hypertext: The organization of information units as a network of associations, which a user can choose to resolve. Hypertext links are the instances of such associations.

Multiversion XML Document: An XML document in which each node can have several branches that correspond to its different versions.

Referential Integrity: In hypertext, a measure of the reliability of a reference to its endpoints. A reference has the property of referential integrity if it is always possible to resolve it. When references are represented as links it is called 'link integrity'.

Versions: Variations of an object with a high degree of similarity. Document versions are never completely equal, but they are similar enough so as to be able to recognise them as the same document.

Version Control: Set of mechanisms that support object evolution in computer applications.

XML: Extensible Markup Language. Markup language for structured documents. Structure is represented with textual markup that intermixes with document content. XML is a recommendation from the World Wide Web Consortium (W3C).

XSLT: XSL Transformations. A transformation language for XML documents. It permits rules for transforming a source tree into a result tree to be expressed. It is a W3C Recommendation.

XQuery: XML Query Language. It is a W3C Recommendation.

ENDNOTE

[1] This chapter corresponds to the updating of the chapter 'Document versioning in Digital Libraries', published in *Rivero, L.C., Doorn, J.H. and Ferragine, V.E. (Ed.), Encyclopedia of Database Technologies and Applications 2005, Idea Group.*

Chapter XVII
MDD Approach for Maintaining Integrity Constraints in Databases

Harith T. Al-Jumaily
Carlos III University of Madrid, Spain

Dolores Cuadra
Carlos III University of Madrid, Spain

Paloma Martínez
Carlos III University of Madrid, Spain

INTRODUCTION

In the context of database, we believe that **MDD** (Model-Driven Development) (OMG, 2006) is a very ambitious task because we find that when applying database development methodologies such as (Elmasri, et al., 2007), there are processes devoted to transforming conceptual into logical schemata. In such processes, semantic losses are produced since logical elements are not coincident with conceptual elements. A correct constraints transformation is necessary to preserve the semantics that reflects the Universe of Discourse. The **multiplicity constraint,** also called cardinality constraint, is one of these constraints that can be established in a conceptual schema. It has dynamic aspects that are transformed into the logical model as certain conditions to verify the insertion, deletion, and update operations. The verification of these constraints is a serious and complex problem because currently database systems are not able to preserve the **multiplicity constraints** of their objects. To solve the modeling problem, **CASE tools** have been introduced to automate the life cycle of database development. These platforms try to help the database developers in different design phases. Nevertheless, these tools are frequently simple graphical interfaces and do not completely carryout the design methodology that they are should to support.

Therefore, in this work the **MDD** approach has been considered to enhance the **transformation**

rules of the conceptual schema into the relational schema. The relational model was considered in this work because most database methodologies are agreeing with it to transform the conceptual schema into a logical one. A tool was plugged into Rational Rose to ensure this task. This tool can automatically generate maintaining mechanisms for these constraints to a target DBMS; triggers system as maintaining mechanisms is used. These mechanisms can provide a fundamental base to obtain a very high level of knowledge independence (Paton, 1999). The triggers system is specified according to the recent **SQL:2003 standard** that revises all parts of SQL99 and adds new features (ISO Standard 2003).

Because triggers development is more complicated than traditional method, we have detected that generating triggers and plugging those into a given schema is an insufficient task because the behaviour of triggers is not clear; they could produce cycles in the execution and needs to be verified. Therefore, besides the transformation of integrity constraints into triggers, our plugged-in tool builds UML sequence diagrams to verify the interaction of these triggers with themselves first, then with the other elements in the schema. These contributions make our approach quite useful, practical, and intuitive to manage triggers.

The rest of this chapter is organized as the following. Related works are presented in the next section. Therefore, the adaptation of **MDD** to build our tool is discussed. In the Add-in Module Design section, we discuss how the integration of the active technology into UML schema is performed. And finally, some conclusions and future works are presented.

RELATED WORKS

When the active technology was introduced into database systems, the automatic transformation from constraints specification to **active rules** has been well considered in the literature. Different approaches were used to transform integrity constraints into **active rules**. Some of these approaches reject updates when the violation occurs, and the initial state before updates is restored (Decker, 2006). Other approach in which inconsistency states are detected first, but consistency is restored by issuing corrective actions that depend on the particular constraint violation (Ceri et al., 1997).

In some works, such as (Ceri et al., 1990) is described a general framework to transform constraints into production **active rules** for constraint maintaining. They define a general language for expressing integrity constraints, and **transformation rules** are used to convert integrity constraints into **active rules**. The contribution in (Türker et al., 2000) is very important to our work because they issued some simple rules that are independent of a particular commercial system. These rules are used for implementing triggers based on constraints specifications by using Tuple Relational Calculus (Elmasri, et al., 2007). In this work, we will specify the **transformation rules** based on constraints specifications by using **OCL** (Object Constraints Language). Our approach agrees with the proposal in (Olivé, 2003) that introduced **OCL** as a method to provide the definition of integrity constraints in conceptual schemas as constraint operations. In this proposal, constraints have been introduced in the conceptual schema as operations without defining any rules to specify the transformation of such operations to a logical schema.

On the other hand, working with **active rules**/triggers needs to ensure the execution termination. The verification of **active rules**/triggers execution is the major problem that makes applications development a difficult task. Because of **active rules** can act in such ways that lead to conflict and undesirable executions, database developers need additional effort to control rules behavior. The objective of this verification is to guarantee the termination of triggers execution.

Termination means that the execution of any set of **active rules** must terminate. This needs to

avoid cycling in the execution. A set of activated rules is confluent if the final state of database is independent on the order in which activated rules are executed (Baralis, et al., 2000). The non-termination state is the major problem that produces an error causing the execution of the transaction to abort.

Many works have been done in the area of static termination analysis. Most of these works such as (Paton, et al., 1999) used the concept of **Triggering Graph** (TG) as approach to detect the non-termination state. The **triggering graph** was introduced in (Baralis, et al., 2000) to detect non-termination state of a set of activated rules. The termination analysis themselves focus on identification and elimination rules whose cause the infinite execution in **triggering graph** (Hickey, 2000). Redefining this rule and reconstructing again the **triggering graph** is a good solution to verifying the termination state of the set of activated rules. Our approach to solve the termination problem is explained in (Al-Jumaily, et al., 2006).

Our main contribution in this work is to create a tool to follow completely the phases proposed in the **MDD** development. According to our approach these phases are the following: specifying a UML class diagram (PIM: platform independent model), transforming the PIM into (PSM: platform specific model) transforming the PSM related to relational database to SQL standard triggers, and finally converting these triggers to triggers of a target DBMS. This work is considered as a proposal to provide a plugged-in tool that will try to fill some of the emptiness that the commercial **CASE tools** leave during the database development.

APPLYING MDA TO GENERATE INTEGRITY MAINTAINING

The introduction of the **MDA** (**Model-Driven Architecture**) approach in Software Engineering has provided a good support and consolidations to automatic **codes generation** for applications development. MDA focuses on using models as approaches to cover the life cycle of software development. The heterogeneity and interoperability among systems with different implementation platforms to apply are good resolved with this approach.

The MDA concept is implemented by a set of tools and standards that can be used within a **MDD**. The MDA specifies two principles models: a platform independent model PIM focuses on the high-level business logic without considering features of the implementation technology of the system. A platform specific model PSM represents the detail of the use of a specific platform by a system.

In this section, we consider the **multiplicity constraints** because most **CASE tools** can define these constraints in the conceptual model (PIM), and in the (PSM). We start from a PIM (UML class diagram) which represents all the **multiplicity constraints** among their classes. The PSM show the transformation of the PIM to multiple models that describe the technology that will be used to build the application.

Applying MDA in our approach is shown in figure 1. It contains three phases; the first one is the PIM which represents the specification of all **multiplicity constraints** into the class diagram. The second is the PSM to describe the technology that will be used to build the application. In this case we use the **SQL:2003 standard** to describe the active technology in relational database. Finally, in the third phase, the SQL:2003 triggers are transformed to a target DBMS triggers **codes generation**.

The main advantage of our tool is to provide active mechanisms to verify the dynamics aspects such as the **multiplicity constraints** of the database. These active mechanisms are derived from the integrity specification in the conceptual model. The verification of these constraints is very difficult. Therefore, we believe that incorporating add-in modules is a good solution to solve

Figure 1. MDA adaptation for our approach

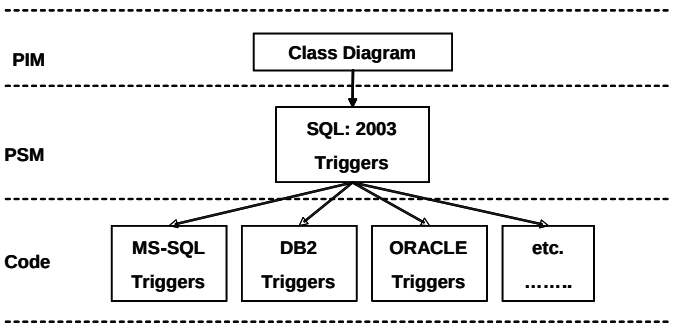
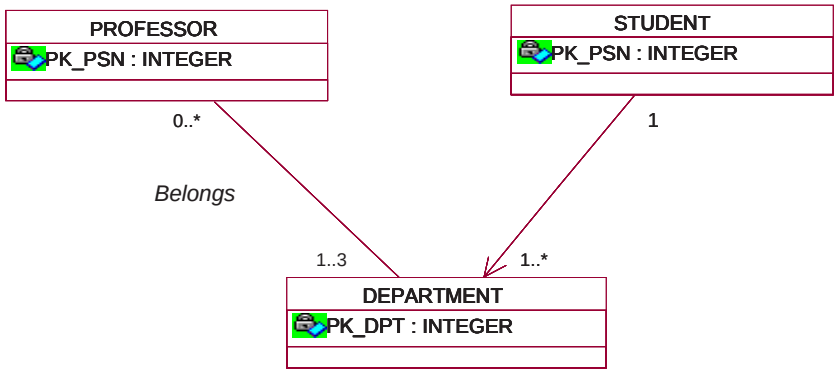


Figure 2. UML class diagram



some of the modeling problems, and to enhance the tasks performed by database designers. The three phases for the development are shown in the following:

PIM: UML Class Diagram

In this model, the **multiplicity constraints** are defined among the class diagram. Since its introduction by (Chen, 1976), the cardinality constraint is defined as the number of entity instances associated in a relationship. UML **multiplicity constraints** follow the Chen's style because to verify the cardinality constraints of one class, it is necessary to fix an object of the other class and to obtain how many objects are related to it (Cuadra, et al., 2002). The figure 2 shows an example of PIM (UML class schema).

In the following, the relationships that belong to the example are shown:

- One-to-many association between the two persistent classes, Student and Department.
- Many-to-many association that relates to the two persistent classes, Professor, and Department.

The **OCL** invariants, which explain the **multiplicity constraints** of the previous relationships are shown as follows:

IC1: Every object in Professor must have associations between 1..3 objects of Department.
Context PROFESSOR: inv
Self.DEPARTMENT->size >= 1 and Self.DEPARTMENT->size <=3

IC2: Every object in STUDENT must have association with only one object of DEPARTMENT.
Context STUDENT: inv
Self.DEPARTMENT->size =1

PSM: SQL:2003 Standard Triggers

Transformation rules are applied in this phase to obtain different logical schemata, although, our proposal only present the transformation to a relational schema.

The figure 3 shows the transformation of our example into relational database schema using the Rational Rose class diagram. The mapping of each class and each association into UML Rational Rose Data Model (RRDM) was done according to (Salo, et al., 2000) and (Vadaparty, 1999).

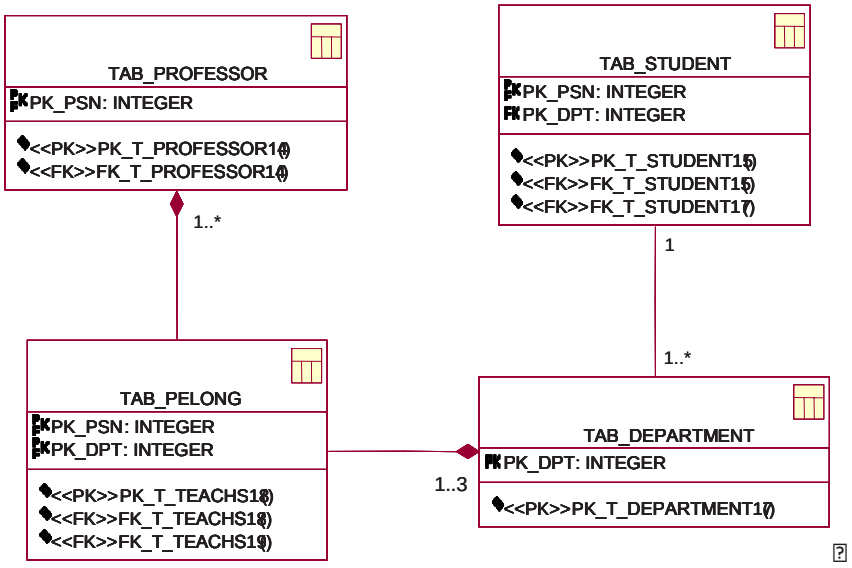
The one-to-many association is transformed into two tables, Tab_Student, Tab_Department, and a non-identifying relationship.

The many-to-many association relates the two persistent classes, Professor and Department. The mapping of this association into Rational Rose uses two tables Tab_professor, Tab_department, and an identifying relationship that is mapped to a third table Tab_Belong.

In the following, the transformation of the PSM (Relational Database schema) to **codes generation** is explained. Although, triggers are available in most DBMS, unfortunately the execution models of these triggers change from one DBMS to another. There are common components that are valid for all systems. These components usually are not changed. Therefore, to generate triggers for commercials DBMSs, first the **SQL: 2003 standard** triggers as common template, and secondly, generating triggers related to DBMSs from the specification of the SQL: 2003 triggers is used.

In general, transforming **OCL** invariants into a SQL trigger is a straightforward process. There are three basic components in any integrity constraints as same as the three basic components of a trigger. The first is an operation that modifies the database state such as DELETE, INSERT, and

Figure 3. Mapping UML class diagram to relational database schema



UPDATE. The second is a logical predicate that expresses a condition on one or several elements of a schema and needs to be fulfilled. The third is the reactions, which are carried out when the constraint is violated. In addition, we need to derive others two dynamic components of triggers. These components are: triggers activation times and triggers granularity. The activation times (BEFORE and AFTER) is used to define if the trigger execution must be produced before or after of the event. Triggers granularity are two levels, a statement-level trigger executes once for each triggering event, while a tuple-level trigger executes for each row that belongs to the modified rows set.

Codes Generation

Although, the almost triggers systems of the relational DBMS have same components as the triggers of SQL standard, the transformation of the standard SQL to the relational DBMS triggers should tack the characteristics of these DBMS into account. In most cases, the transformation is directly, that is to say, a trigger of the standard SQL is transformed into one trigger of a DBMS (1 to 1), while in other cases is needed to use two triggers of the DBMS to represent a SQL standard trigger (1 to 2). The major task associated to this transformation is to solve the problems of the triggers execution. One of these problems is related to limitations in some relational DBMS, such as Oracle that needs to solve the problem of mutating tables. Another problem is to avoid the non-termination state produced when the disjoint-total constraints of generalization is conserved. This problem is solved by using parameters to avoid or deactivate the trigger that may be executed twice in the same set of activated triggers.

ADD-IN MODULE DESIGN

We applied our approach on the Rational Rose CASE tool because it is able to easily add-in software tools to support the development needs. Add-Ins can install menus, help files, contents tab file, properties, executables, script files, and OLE servers. Our add-in module was developed using Basic Script Language Basic, (1996), and can be accessed from the Tools menu. As shown in the figure 4, the add-in module has an interface that shows some options that we need to consider before generating triggers.

The add-in module detects and presents all relationships that belong to the current scheme. The list box "Current-Model Relationships" presents all relationships that exist in the current scheme. According to the required semantics, the user can choose those relationships that he needs

Figure 4. Add-in interface design

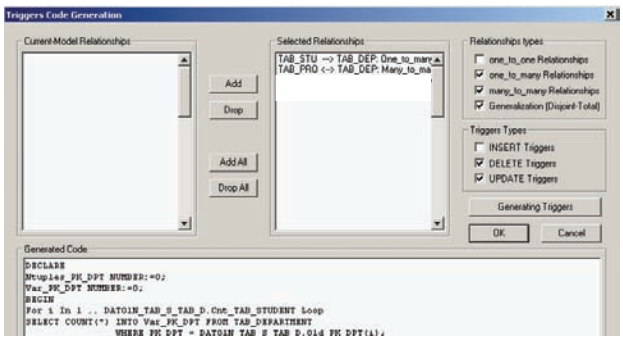
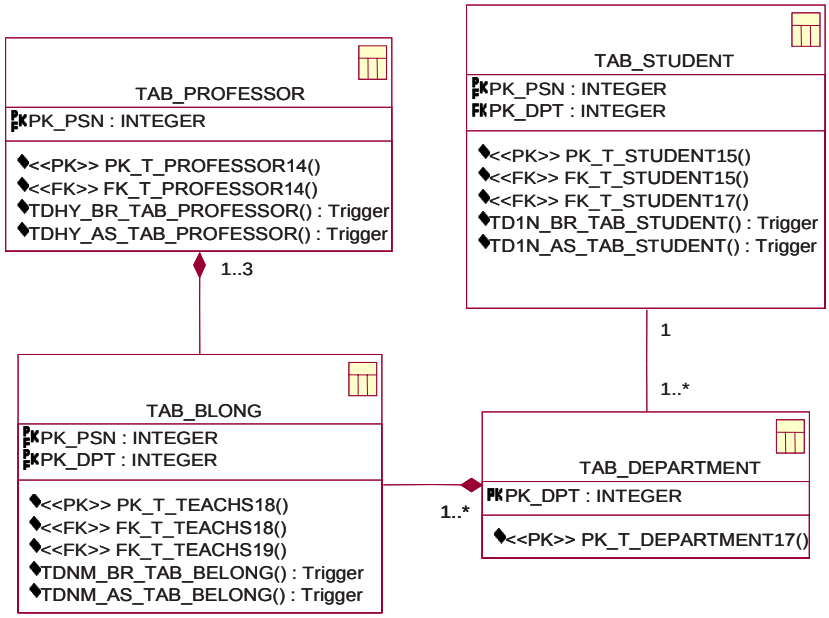


Figure 5. Plugged operations into the schema



to preserve. The list box "Selected Relationships" shows the selected relationships to be controlled. The user can choose one or more relationships to be controlled; as is shown in the figure we close all the relationships that belong to our example. According to our example, the add-in interface represents three type of relationships, one-to-many relationship (TAB_STU-->TAB_DEP) that associates tab_student with tab_department, many-to-many relationship (TAB_PRO<-->TAB_DEP) that associates tab_professor with tab_department, and generalization relationships (TAB_PER<>--TAB_PRO, TAB_STU) that associates the superclass tab_person with tab_student and tab_professor. The check boxes "Relationship Types" show the relationship types that the add-in module considers. The check boxes "Triggers Types" represent the types of triggers to be generated. The generated code that we obtain is saved in a SQL file that contains triggers and packages to define the global variables.

Because of ORACLE triggers system has not Old/New-Table referencing values, and due to mutating table problem, we used two triggers for controlling each event. Therefore, each trigger is transformed into two ORACLE triggers. The first (BEFORE/ROW) is used to save the identification keys, and the second (AFTER/STATEMENT) is used to verify the semantics.

The add-in module plugs these triggers into an UML schema as operations using:

Set theOperation = theClass.AddOperation (OperationName, OperationType)

Figure 5 shows the plugged operations that have been generated to control the deleted events. The triggers (TD1N_BR_TAB_STUDENT and TD1N_AS_TAB_STUDENT) are used for controlling the deletion of foreign keys that come from one-to-many relationship, while the triggers (TDNM_BR_TAB_BELONG, and TDNM_AS_TAB_BELONG) are used for controlling the many-to-many relationship.

CONCLUSION

Although the CASE database development tools have been developed to resolve the database-modelling problem and to provide automatic processes to develop all phases supported in a database methodology, the current situation of these tools shows that they provide conceptual models with more abstraction and are concerned to express more accurately the semantics of the real world. However, moving from conceptual level to logical level there is not any support and the generated code needs to be modified to complete the integrity constraints of the real world. One of these constraints is the **multiplicity constraints** that should be defined in the conceptual model nevertheless these constraints have not any support when the conceptual model is transformed to logical model.

Therefore, to fill some of the emptiness that the current **CASE tools** leave during the relational active database development in this work we present a tool, which follows completely the phases proposed in the MDA software development to transform the **multiplicity constraints** to triggers. These phases are the following: specifying **multiplicity constraints** in the UML class diagram, transforming the these constraints to **SQL:2003 standard** triggers, and transforming the standard triggers to a target DBMS triggers. Thus, this work joins the UML aspects that have widely accept and support by many **CASE tools** with the relational database aspects that have widely propagation in the commercial DBMS.

FUTURE WORKS

As a future work, we will propagate our approach to include other type of constraints and design some experiments to validate our tool. These experiments are focused on showing the utility of using it to facilitate maintenance and design tasks. Therefore, we propose two kinds of experimentations, one of them concern the utility to check the semantics with triggers. The other one is about the user interface to show triggers and sequence diagrams. We want to know if the designer understands the proposed diagrams and detects what does each one.

REFERENCES

Al-Jumaily, H. T., Pablo C., Cuadra D., & Martínez P. (2006). Using UML's sequence diagrams as termination analyzer for triggers-based executing. *In the 23rd British National Conference on Databases,* 18-20. Northern Ireland, Queen's University Belfast.

Baralis, E., & Widom, J. (2000, September). An algebraic approach to static analysis of active database rules. *ACM Transactions on Database Systems, 25*(3), 269-332.

Ceri, S., & Fraternali, P. (1997). Designing database applications with objects and rules: The IDEA Methodology. Addsion-Wesley. ISBN: 0201403692

Chen, P. (1976). The entity-relationship model – Toward a unified view of data. *ACM Transactions on Database Systems, 1*(1).

Cuadra, D., & Martínez, P. (2002). Preserving relationship cardinality constraints in relational schemata. *Database Integrity: Challenges and Solutions*, (Ed.). Idea Group Publishing.

Decker, H., Martinenghi, D., & Christiansen, H. (2006). Integrity checking and maintenance in relational and deductive databases and beyond. *In Intelligent Databases: Technologies and Applications,* 238-285. Idea Group.

Elmasri, R., & Navathe, S. (2007). Fundamentals of database systems, 5th Edition. Addison-Wesley. ISBN: 9780321415066

Hickey, T. (2000). Constraint-based termination analysis for cyclic active database rules. *Proc.*

DOOD'2000: 6th. International Conference on Rules and Objects in Databases, LNAI, 1861, 1121-1136.

ISO Standard (2003). Information technology – Database languages – SQL: 2003 International organization for standardization.

Olivé, A. (2003). Integrity constraints definition in object-oriented conceptual modeling languages. *Conceptual Modeling - ER 2003, 22nd Int. Conf. on Conceptual Modeling.* Chicago, IL, USA.

OMG. (2006). Object management group inc. http://www.omg.org/mda.

Paton, N., & Díaz, O. (1999). Active database systems. *ACM Computing Surveys, 31*(1).

Salo T., & Hill, J. (2000). Mapping objects to relational databases. *Journal of Object Oriented Programming, 13*(1).

Türker, C., & Gertz, M. (2000). *Semantic integrity support in SQL-99 and commercial (object) relational database management systems.* UC Davis Computer Science Technical Report CSE-2000-11, University of California.

Vadaparty, K. (1999). ODBMS - *Bridging the gap between objects and tables: Object and data models, 12*(2).

Basic, (1996). BasicScript 2.25 Language Reference, Summit Software. http://www.cardiff.com/CSdownload/misc/t1057.pdf.

KEY TERMS

Integrity Constraint: An integrity constraint is used to specify some restriction for maintaining the consistency of data. It can be defined in such a way that in a table (or more than one table) all qualified tuples must satisfy a logical predicate.

MDD (Model-Driven Software Development): Focuses on using models as approaches to cover the life cycle of software development. The main contribution of **MDD** is to give a solution to heterogeneity and interoperability among systems with different implementation platforms.

Multiplicity Constraints: Also called cardinality constraint, is one of these constraints that can be established in a conceptual schema. It has dynamic aspects that are transformed into the logical model as certain conditions to verify the insertion, deletion, and update operations.

Platform Independent Model (PIM): Focuses on the high-level business logic without considering features of the implementation technology of the system.

Platform Specific Model (PSM): Describes the technology that will be used to build the application.

Termination: The execution of any set of **active rules** must terminate. This needs to avoid cycling in the execution. A cycling means a non-termination problem is produced in the triggers execution.

Triggers: A trigger is a named event-condition-action rule that is activated by a database state transition. Once a trigger is activated and its condition is evaluated to true, the predefined actions are automatically executed.

Chapter XVIII
Artifacts for Collaborative Software Development

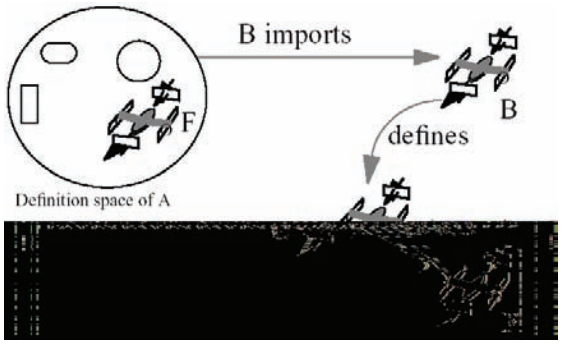
Pierre F. Tiako
Langston University, USA

INTRODUCTION

The development of software applications generally requires the following: hardware resources (computers, networks, peripherals, etc.), software resources (data, tools, etc.), human resources (individuals with various qualifications), and working methods. These resources are distributed in different autonomous software development environments. A single environment does not always have all the necessary resources to realize some large and/or complex projects. Therefore, collaboration between the environment in charge of the project (coordinator) and others (contractors) will be required to do the job.

While several research projects have contributed to various aspects of collaboration among software development environments during the past decade, little has been done on explicitly defining and modeling processes and environment artifacts involved in such partnerships. That is what this chapter is about. In the context of this study, environments work together by assigning tasks and sharing working methods. Tasks and working methods can be defined explicitly using process models. Process models, already the main focus in monolithic software development, will still be an important factor in our approach of collaborative software development. Because they are process-based, all software development environments considered here will be qualified in the continuation of Process-sensitive Software Engineering Environments (PSEEs).

Table 1. Brief process modeling history and limits



o automate software development processes, i.e. to build a software system based on process models to assist and guide software production. Several works contributed to this subject (Derniame & Gruhn, 1994), enabling it to reach a certain maturity.

BACKGROUND

The objective of developing large software projects can be reached more easily when the work to be done is divided into tasks (Smith, Hale & Parish, 2001) and assigned to various PSEEs. Tasks and working methods can be explicitly defined using process models. Table 1 proposes a brief history of process modeling and its limits.

Software processes are modeled before being assigned to PSEEs for performance. Collaborations to deal with these issues have also been discussed in information systems (Hahn, Jarke & Rose, 1991; Mookerjee & Chiang, 2002).

A simple solution toward this goal is to provide PSEEs with protocols that allow them to communicate in order to import or export process components as well as other PSEE artifacts. First investigations presented hereafter have been made in the subject by Oz (Ben-Shaul & Kaiser, 1995, 1998) and Federated PSEE (Basile, Calanna, Nitto, Fuggetta & Gemo, 1996) approaches.

Oz proposes to compose different instances of PSEEs where each of them is devoted to supporting the development process executed by a single organization. All these PSEEs run autonomously according to their own processes, but they can interact to accomplish common activities such as the integration test of software components that have been independently developed by various organizations. In Oz, the strategy through which a common activity is executed is called a summit. In a summit, one site acts as a coordinator. It receives from the other sites all data needed to execute the common activity, executes the activity, and sends the results back to the other sites. This behavior is obtained by directly implementing the summit protocol and procedures as a basic mechanism of the PSEEs.

Federated PSEEs takes Oz as a starting point and allows several interorganization policies to be implemented and combined. The solution of this issue is to provide a set of basic operations to specify any interorganization policy. Examples of operations are a search for the physical location of

a site and a request for the execution of one service at some physical location. Basic operations should be powerful enough to make the specification of any reasonable interorganization policy as simple as possible. The enactment of an interorganization process requires that sites can be located over the network, communication is enabled among sites, data can be safely exchanged, and some access control policies are executed.

These two approaches, original and complementary, are important contributions to address the federation problem. Both have in common to define a priori some interorganization policies. They limit the various types of interorganization policies that could be defined, established, and performed among PSEEs working together.

MAIN FOCUS OF THE CHAPTER

This chapter explores software processes and PSEE-related artifacts for collaborative software development. It also discusses how collaborations should be defined in order to support various types of interorganization policies that could be defined, established, and performed among PSEEs working together.

Process component is an encapsulation of process model properties and a behavior that allows handling these properties. A process component is defined to be performed across PSEEs (Table 2). It can have an unspecified size and be composed of a hierarchy of activities.

The concept of activity corresponds to a module for design and performance; it provides a support to progressively refine a process model. A process component under definition receives the name of its root activity. The concept of activity we use here is extensively presented in Tiako and Derniame (1999). A process component is defined around its root activity. All its root subactivities are introduced recursively into the component, as well as all entities linked to all the activities of a component. The hierarchies of root subactivities define its subcomponents. Activities can be instantiated or not before being attached to a process component under definition. In the first case, activities are ready to be performed without option of modification. In the second case, they can be instantiated later on prior to their performance. When defined, a process component is assigned a definition space. For instance, Table 2 shows the definition space of component *F*.

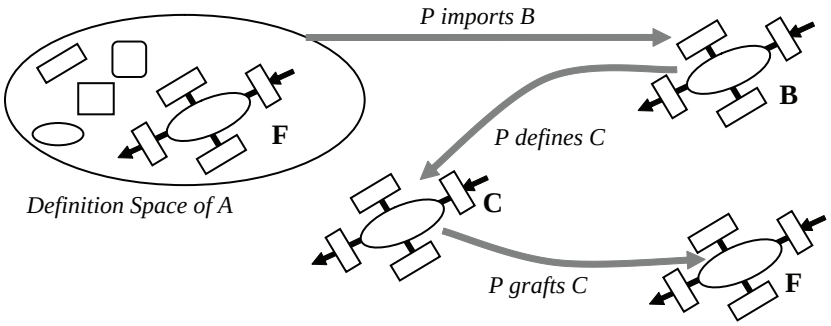
A process component can be exported or imported from the definition space of one PSEE to the definition space of another. That makes it possible to perform a process component across PSEEs.

These artifacts of the model of PSEE presentment in Table 3 serve as basis elements to support collaborative software development. A PSEE distinguishes active and passive entities. Active entities are those that manage others (Gorla & Lam 2004; Jiang, Klein & Pick, 2003). In the system presented, they are represented by "Participant" entity. Passive entities are those that are managed. They are composed by process components and their artifacts, as well as by other PSEE entities such as "Background" and "ModelType." The semantic of PSEE entities and relations among them are given in the following.

Participant. Participant is an active entity that allows to define and enact processes and to control processes during their performance. For instance, Table 2 shows how participant *P* is defining component *F*. *P* is doing so by first importing an existing component *B* by defining a new component *C* and by grafting *F* to *C* to form one of its subcomponents.

Background. Background defines experiences already acquired by a PSEE. It provides information on projects developed. For each project, it records its objective, duration, results obtained, and turnover carried out. It also saves information on PSEE partners as well as their skills and services provided in the collaboration.

Table 2. Process component modeling across PSEEs



ModelType. ModelType defines the types of process model a PSEE supports. Supporting a particular model type also means understanding the constraints to be satisfied and advisory guidance on carrying out software processes. Some types of process models might include statecharts, petrinets, and rule-based constraints.

Capability. Capability defines the services a PSEE can provide to others. The model of PSEE proposed distinguishes three classes of services: Factory, Perform, and Control. Factory expresses the capability of a PSEE to model various types of processes. Perform defines the capability of a PSEE to enact and perform various types of processes. Control expresses the capability to control various types of processes during their performance.

The capabilities of a PSEE are defined in terms of (1) roles that participants can play in the system, (2) tools likely to be used to perform processes, and (3) type of process models supported by a PSEE.

Ability. Ability defines the competence of a PSEE with regard to its capabilities and background. This allows evaluating the experiences already acquired by a PSEE.

FUTURE TRENDS

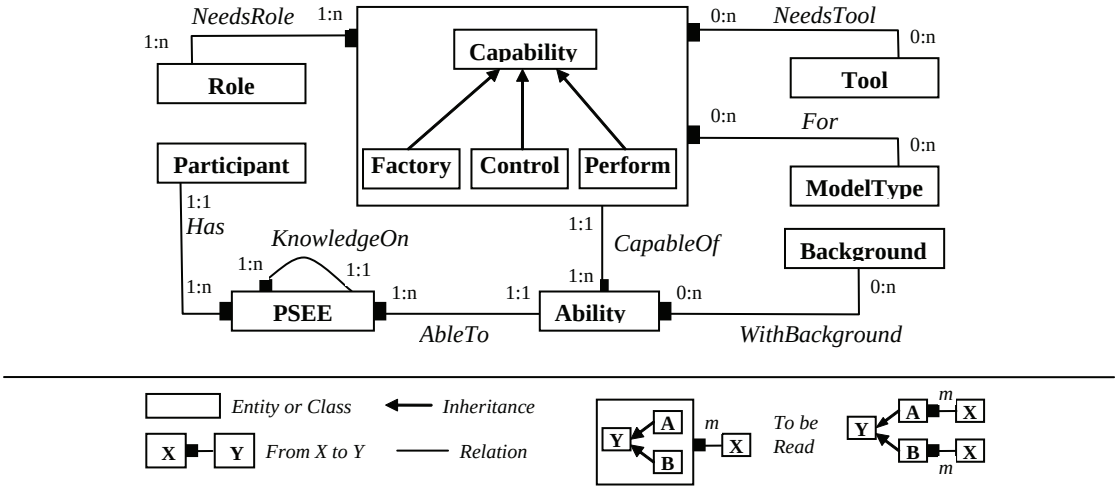
The main challenges will be to define various types of contracts to be established and performed among PSEEs willing to work together. We believe that the federation processes have to be modeled with the same level of abstraction and formalism always used to model software processes.

Emerging trends for supporting collaborative software development can be divided into three categories:

- Approaches based on interorganization policies or delegation
- Approaches based on contracts or negotiation
- Approaches based on agent collaboration or teamwork

Emerging trends should support mobile processes, mobile work, and model of kinds of collaboration for performing process components across PSEEs. It should be interesting in the future trends to study and implement the mobility policies of process components performed across PSSEs. Initial work on delegation (Tiako & Derniame, 1999) and other related works on process migration facility (Artsy & Finkel, 1989), and mobile work (Sørensen, 2005) should be a good starting

Table 3. Model of PSEE and its artifacts



point for the study. The implementation should discuss the advantages and inconveniences to be based on Object Management Group (Hailpern & Tarr 2006; O'Ryan, Schmidt & Noseworthy, 2002) or Extensible Markup Language (Abiteboul, Bonifati, Cobéna, Manolescu & Milo, 2003; McLaughlin & Edelson, 2006) infrastructures for distribution purposes.

CONCLUSION

The objective of developing large software projects can be reached more easily when the work to be done is divided into tasks and assigned to teams to implement them on independent, heterogeneous, and distributed PSEEs. The tasks are modeled as process components before being assigned and performed across PSEEs.

This chapter explores software process component and PSEE-related artifacts for collaborative software development. It also discusses how collaborations should be defined in order to support various types of interorganization policies that could be defined, established, and performed among PSEEs working together.

REFERENCES

Abiteboul, S., Bonifati, A., Cobéna, G., Manolescu, I., & Milo, T. (2003). *Dynamic XML documents with distribution and replication*. Proceedings of SIGMOD 2003, San Diego, California.

Armitage, J., & Kellner, M.A. (1994). *Conceptual schema for process definitions and models*. Proceedings of the Third International Conference on Software Process, Reston, Virginia, 153–165.

Artsy, Y., & Finkel, R. (1989). Designing a process migration facility. Computer, *22*(9), 47–56.

Basile, C., Calanna, S., Nitto, E., Fuggetta, A., & Gemo, M. (1996). *Mechanisms and policies for federated PSEEs: Basic concepts and open issues*. Proceedings of the 5th European Workshop on Software Process Technology, LNCS, 1149, 86–91.

Ben-Shaul, I.Z., & Kaiser, G.E. (1995). A paradigm for decentralized process modeling. Kluwer Academic Publisher.

Ben-Shaul, I.Z., & Kaiser, G.E. (1998). Federation of process-centered environments: The Oz experience. *Automated Software Engineering, 5*(1).

Derniame, J.-C., & Gruhn, V. (1994). Development of process-centered IPSEs in the ALF project. *Journal of Systems Integration, 4*(2), 127–150.

Gates, B. (2005). Building software that is interoperable by design [technical report]. Microsoft Corporation Web site. Retrieved December 2006, from http://www.microsoft.com/mscorp/execmail/2005/02-03interoperability.mspx.

Gorla, N., & Lam, Y.W. (2004). Who should work with whom? Building effective software project teams. *Communications of the ACM, 47*(6), 79–82.

Hahn, U., Jarke, M., & Rose, T. (1991). Teamwork support in a knowledge-based information systems environment. *Transactions on Software Engineering, 17*(5), 467–482.

Hailpern, B., & Tarr, P. (2006). Model-driven development: The good, the bad, and the ugly. *IBM System Journal, 45*(3).

Jiang, J.J., Klein, G., & Pick, R.A. (2003). The impact of IS Department organizational environments upon project team performances. *Information and Management, 40*(3), 213–220.

McLaughlin, B., & Edelson, J. (2006). *Java and XML*. Third Edition. O'Reilly.

Mookerjee, V.S., & Chiang, I.R. (2002). A dynamic coordination policy for software system construction. *IEEE Transactions on Software Engineering, 28*(7), 684–694.

O'Ryan, C., Schmidt, D.C., & Noseworthy, J.R. (2002). Patterns and performance of a CORBA event service for large-scale distributed interactive simulations. *International Journal of Computer Systems Science and Engineering, 17.*

Smith, R.K., Hale, J.E., & Parish, A.S. (2001). An empirical study using task assignment patterns to improve the accuracy of software effort estimation. *IEEE Transactions on Software Engineering, 27*(3), 264–271.

Sørensen, C.F. (2005). Adaptive mobile work processes in context-rich, heterogeneous environments [doctoral thesis]. Norwegian University for Science and Technology.

Tiako, P.F. (2005). *Collaborative approach for modeling and performing mobile software process components*. Proceedings of the 2005 International Symposium on Collaborative Technologies and Systems (CTS 2005), Austin, Texas.

Tiako, P., & Derniame, J.-C. (1999). *Toward process components mobility in federated PSEES*. Proceedings of the 5th International Conference on Information Systems, Analysis and Synthesis: ISAS'99, Orlando, Florida.

WfMC. (2006). Interoperability abstract specification. WfMC Document Number TC-1012, Version 2.0b. Retrieved December 2006, from http://www.wfmc.org/standards/docs/TC-1012_Nov_99.pdf

KEY TERMS

Activity: Any piece of work that must be done; an instance composed of an activity name and its relationships with product, direction, tool, role, and its subactivities. The activity must have been assigned to precisely one role. There must exist at least one agent that can perform this role.

Agent: A model entity that is able to perform roles and hence to carry out activities. An agent may be human, automated, or some combination of both.

Direction: A model entity that defines the objectives of the associated activities, defines any constraints to be respected, and may provide advisory guidance on carrying out the activities. Direction must provide instructions sufficient to complete the activity as required.

Federation Processes: Processes that allow modeling collaboration among distributed software development environments. Such processes allow dynamically building various contract types that can be established among environments.

Mobile Process Component: Process component likely to be routed on the network. It can thus be exported or imported for reuse or performance.

Process Component Delegation: Act that allows exporting a process component remotely for its performance and/or control.

Product: Can form an input to an activity, an output from an activity, or an intermediate result of an activity. In the latter case, it is accessible from the tools of the activity, from its subs, but not from outside.

Role: A model entity that identifies a skill requirement that an agent must satisfy in order to perform the role. During process definition, a number of activities can be assigned to a role, and a number of agents can be identified to be able to perform a role.

Tool: A model entity that is a piece of software. It can be employed in carrying out an activity. Any activity may employ an arbitrary number of tools.

Section II
Logical Modeling

Chapter XIX
Object–Relational Modeling

Jaroslav Zendulka
Brno University of Technology, Czech Republic

INTRODUCTION

Modeling techniques play an important role in the development of database applications. Well-known entity-relationship modeling and its extensions have become a widely-accepted approach for relational database conceptual design. An object-oriented approach has brought a new view of conceptual modeling. A class as a fundamental concept of the object-oriented approach encapsulates both data and behavior, whereas traditional relational databases are able to store only data. In the early 1990s, the difference between the relational and object-oriented (OO) technologies, which were, and are still used together to build complex software systems, was labeled the *object-relational impedance mismatch* (Ambler, 2003).

The object-oriented approach and the need of new application areas to store complex data have greatly influenced database technology since that time. Besides appearance of object-oriented database systems, which fully implement object-oriented paradigm in a database environment (Catell et al., 2003), traditional relational database management systems become object-relational (Stonebraker & Brown, 1999). The most recent versions of the SQL standard, SQL: 1999 (Melton & Simon (2001) and SQL: 2003 (Eisenberg et al., 2004), introduced object-relational features to the standard and leading database producers have already released packages which incorporate them.

Development of complex data intensive software systems involves a close working relationship between software and database developers. Currently, software developers deal with object-

oriented software development and use object-oriented modeling techniques to represent the main view of the application, whereas database developers model, design, build, and optimize the database. However, modeling techniques for relational databases do not support important features of object-relational databases. In addition, shared vision and clear communication of project details are necessary conditions for a software project to be successful. A common modeling language and supporting development tools can provide good conditions for it.

The Unified Modeling Language (UML) was adopted as an OMG (Object Management Group) standard for object modeling in 1997. Since that time, it has become popular and widely used. It provides several modeling techniques (diagrams) that visualize a system from different perspectives. From database design point of view, a class diagram is the most important diagram—it shows a set of structural elements and their static relationships. It can be used for conceptual modeling of persistent classes. But the UML does not contain direct support neither for capturing important features of relational databases, nor for specific features of object-relational databases that are necessary for modeling data stored in a relational database and objects stored in an object-relational database at design levels below the conceptual one. Therefore, it is necessary to add the ability to model features of this kind of databases in an effective and intelligible way. Fortunately, the UML provides an extensibility mechanism that allows doing it.

This article shows how an object-relational database schema can be modeled in UML. This technique will be referred to as object-relational modeling here.

BACKGROUND

In this section, we summarize important features of the SQL:1999 object-relational model (Melton & Simon 2001, Database Language SQL, 1999, Badia 2006) with some extensions available in SQL:2003 (Eisenberg et al., 2004, Database Language SQL, 2004). We also explain how the model is implemented by Oracle in their database servers. Then we describe extensibility mechanisms of the UML, which we exploit in the next sections.

The Object-Relational Data Model in Standard SQL and in Oracle SQL

We can say that since SQL:1999 the underlying model of the standard is object-relational and therefore the standard has also become a standard database language for object-relational databases. We only focus on the most important features of the model that represent object extensions here.

The model keeps the concept of tables but it removes the First Normal Form (1NF), which is the basic requirement of the relational model of data. The standard introduces *user-defined types* (UDTs), which mean types specified by users. There are two types of UDTs – distinct types and structured ones. The *structured types* are more interested from our point of view. A structured type can have an internal structure referred to as attributes and it can include methods that represent behavior of instances of the type. Similarly to classes, a structured type can inherit properties of another type. A structured type can represent a domain of a table column or it can specify the type of tupples in a table. There is an important difference concerning value identity between these two cases. Values in the former case do not have any identity. They are values of a composite attribute. The tupples in the latter case have identity known from the OO approach as an object identifier (OID). They represent objects with corresponding attributes and methods, and the table that stores them is said to be a *typed table*. The SQL refers the identifying values to as *REF values* and the relevant type as *REF type*. It is possible to specify the domain of

a table column or attribute as a REF type and to use REF values as links between objects instead of foreign – primary key pairs known from the relational model.

The structured UDTs are not the only extension that violates 1NF. Another one are *collection types* that can also be employed in table column or UDT attribute specifications. There are two types of collections available:

- arrays
- multisets (not available in SQL:1999, introduced in SQL:2003)

Both collection types are collections of data elements of the same type. An *array type* is ordered, whereas *a multiset type* is unordered. In addition, cardinality must be specified for arrays.

The SQL:1999 also introduced *typed views*, which are similar to typed tables, but they are virtual tables.

The object-relational model has been implemented in Oracle since Oracle8 version. It is compliant with the standard in many core features, but it uses a different terminology. User-defined types are termed *object types*, typed tables are called *object tables*, and typed views *object views*. The array type in Oracle (termed *VARRAY*) is bounded by maximum cardinality and multiset has the form of a *nested table*, which is unbounded. Moreover, inheritance was not supported in Oracle8, it was included later.

Extensibility Mechanisms in UML

The UML provides three extensibility mechanisms that make possible to extend the language in controlled ways (Booch, Rumbaugh & Jacobson, 1999, OMG, 2007):

- Stereotypes
- Tagged values
- Constraints.

A *stereotype* extends a vocabulary of the UML. It allows introducing a new model element derived from one existing in the UML metamodel. A *tagged value* extends the properties of UML model elements. It is a keyword-value pair element that may be attached to any kind of a model element. The keyword is called a tag. A *constraint* extends the semantics of a model block by means of specifying conditions and propositions that must be maintained as true otherwise the system being described by the model is invalid. There are some standard stereotypes, tagged values, and constraints predefined in the UML. One of them is a stereotype <<Table>>, which is a stereotype of the UML class element.

The main purpose of the extensibility mechanisms is to tailor the UML to the specific needs of a given application domain or target environment. It makes it possible to develop a predefined set of stereotypes, tagged values and constraints and notation icons that collectively specialize and tailor the UML for specific domain or process. Such a set is called a UML *profile*. Several profiles have already been accepted by OMG as standard profiles (OMG, 2007), but none of them is for data or object-relational modeling.

Several UML profiles for data modeling have been proposed (Using Rose Data Modeler, 2001, Marcos, Vela & Cavero, 2001, Ambler, 2003) and several proposals for object-relational modeling have been published too. The latter are discussed in the next section.

MAIN THRUST: UML PROFILES FOR OBJECT-RELATIONAL MODELING

Several works that propose a UML profile for object-relational modeling have been published (Rational 98i Using Rose Oracle8, 1998, Marcos, Vela & Cavero 2001, 2003). They have been proposed for SQL:1999 and/or the SQL dialect for Oracle8 because this DBMS had provided object

extensions before SQL:1999 was published. One of them is a proposal for a tool integrated by Rational Software Corporation in their Rational Rose®[1] (hereafter simply referred to as the Rose).

In this section we first describe the Rose UML profile for Oracle8 and then briefly discuss some other profiles.

The Rose Profile for Oracle8

The Rose is one of the best-known UML-oriented modeling tools. It provides support for several target programming and database environments, including Oracle. The Rose Oracle8 tool makes it possible both forward and backward engineering of Oracle8 object-relational schemas.

The profile (hereafter also referred to as the Rose profile) introduces several stereotypes, some constraints in the form of conventions, and tagged values. The tagged values have a form of schema generation properties attached to a project, class, operation, and attribute. They contain such values as the length of a VARRAY type, a WHERE clause of a view definition etc. Stereotypes of the profile are summarized in Table 1. It is evident that most of stereotypes are stereotypes of a UML class element. The stereotypes for a database and schema can be used on package or component diagrams. We do not employ them in our illustrative example.

Relationships between schema elements are modeled as associations, aggregations, and dependencies. Dependencies are used primarily to express relationships between data types and structures that store values of these types. Nesting of types is modeled by associations.

We use a simple *Purchase Order* example to show how the basic object extensions of Oracle8 are modeled in the profile. A conceptual class diagram of the example is displayed in Figure 1 on the next page. Only one operation (*sumLineItems()* of the *Order* class) is shown in the diagram. The *Order* and *Customer* classes contain a complex attribute of an *Address* type consisting of *street*, *city*, and *zip* attributes.

Table 1. Stereotypes in the Rose profile for the Oracle8

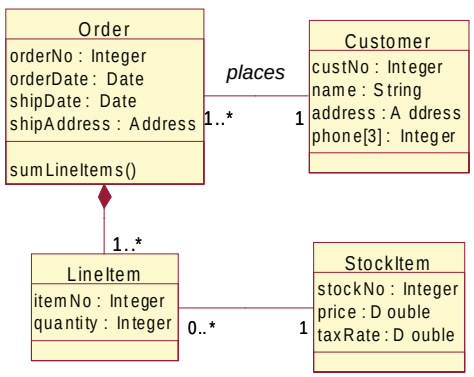
Stereotype	UML model element
<<Database Domain>>	Package
<<Schema>>	Component
<<ObjectType>>	Class
<<ObjectView>>	Class
<<ObjectTable>>	Class
<<NestedTable>>	Class
<<RelationalTable>>	Class
<<RelationalView>>	Class
<<VARRAY>>	Class
<<ObjectView>>	Dependency

Object Types

Object types are modeled by a stereotype <<*ObjectType*>>, which is a stereotype of class. Attributes of a given object type are modeled as attributes, methods as operations.

An object type can contain an attribute of another object type. This is modeled as a unidirectional association from the outer object type to the nested object type. A role name of the nested

Figure 1. A conceptual class diagram of a Purchase Order example



type is the name of the corresponding attribute. Association in the profile implies a relationship "*by value*". Methods of object types are modeled as operations of classes with some conventions.

VARRAYs

A VARRAY collection type is modeled by a stereotype <<*VARRAY*>>, which is a stereotype of class. All elements of the array must be of the same type, either scalar or object one. If the array is based on an object type, the relationship is modeled as dependency of the VARRAY type on a given object type. A VARRAY can be used as an attribute of an object type. Such nesting is modeled by association.

Let us assume that the *Customer* class from our example will be implemented by means of an object type *CUSTOMER_T* with a nested type *ADDRESS_T* and a *VARRAY* type attribute for a fixed-size array of phone numbers. The corresponding model is in Figure 2. The scalar type of the array element and its parameters are not displayed. This information is contained in tagged values.

Nested Tables

A nested table is modeled as a class with a stereotype <<*NestedTable*>>. If the nested table is based on an object type, the relationship is again modeled as dependency of the nested table on a given object type. A nested table can be used as an attribute of an object type, which is represented by association in the model.

Let us assume that the *Order* class from our example will be implemented by means of an object type *ORDER_T* with a nested type *ADDRESS_T* and a nested table containing all items of a given order (of a *LINEITEM_T* type). The corresponding model is in Figure 3.

REFs

A REF type is modeled by aggregation. In the profile, aggregation implies "by reference" containment. Figure 4 shows that the *ORDER_T* type contains an attribute *CUSTOMER* of the REF type, which references an object of the *CUSTOMER_T* type.

Figure 2. Modeling of a nested object type and an array

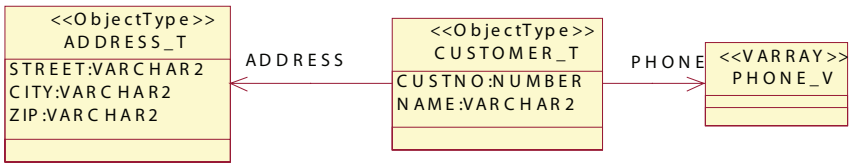


Figure 3. Modeling of a nested table

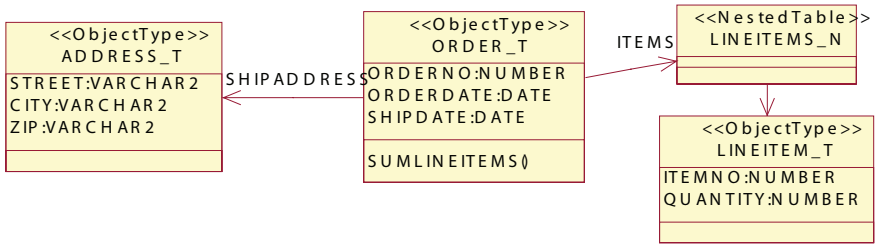
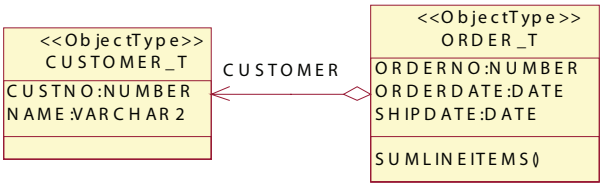


Figure 4. Modeling of a REF type



Object Tables

An object table is modeled as a class with a stereotype <<*ObjectTable*>>. Since object tables are built from underlying object types, this relationship is modeled as a dependency. An example can be found in Zendulka (2006).

Relational Tables

A relational table is modeled as a class with a stereotype <<*RelationalTable*>>. Attributes of the table that are of an object type, VARRAY, nested table, and REF are modeled in the same way as for object types.

Object Views

An object view is modeled as a class with a stereotype <<*ObjectView*>>. Links between source tables (object or relational) and the object view are modeled as dependencies. The relationship with the underlying object type of the view is modeled as a dependency with a stereotype <<*ObjectView*>>. An example can be found in Zendulka (2006).

Unfortunately, the Rose Oracle8 tool has not been adapted for later releases of Oracle servers. Therefore, it does not reflect their new features, even inheritance of object types.

Other Profiles

Marcos, Vela & Cavero (2001, 2003) proposed UML profiles both for SQL:1999 and Oracle 8i. Similarly to the Rose profile for Oracle8, the authors introduce stereotypes of class for structured types and typed tables. But there are important differences in modeling some other elements. First of all, they introduce a stereotype of association <<Knows>>, which models the relationship between a structured type and a typed table. In the Rose profile it is modeled as dependency, which may be a better choice.

Another difference is in modeling attributes of structured, collection and REF types. The authors apply the following criterion: If the type may be defined explicitly in the SQL schema, then it is modeled as a stereotyped class, otherwise as a stereotyped attribute. In the Rose profile, all these cases are modeled by means of associations and role names.

The profiles does not deal with typed and object views. On the other hand, they introduce stereotypes for a ROW type and redefined and deferred methods, which are features of SQL: 1999 not explained in this article.

FUTURE TRENDS

Currently, object-relational modeling is not used in practice very often. First of all, it is the conse-

quence of the fact that object features of object-relational databases and SQL: 1999 and 2003 has not been well accepted in the market place and adopted by database vendors yet, and that there is small experience with these features (Ambler, 2003). It is true for typical OLTP applications. Another reason is that despite of less impedance mismatch with object, there are not suitable standard interfaces for comfortable accessing complex database objects. In addition, only a few methods that generate an object-relational model from a conceptual model (Mok & Paper, 2001) and methodologies that guide designers in the object-relational database development have been published. Vara et al. (2007) propose such a methodology based on model-driven architecture (MDA) concepts. Grant, Cennamaneni & Reza (2006) use the formal transformations of the methodology in a comparative study.

On the other hand, we can say that application-oriented built-in support, such as Spatial or Intermedia in Oracle, is very often based on predefined structured UDTs, which creates conditions for object-relational features to be accepted by developers better.

CONCLUSION

The objective of this article was to introduce UML profiles for object-relational modeling. All such profiles exploit the extensibility mechanism of the UML. We have chosen the profile used by Rational Rose Oracle8 tool as a representative one and have described it. We have compared it with other approaches too. Such profiles can be useful not only for manual object-relational database schema modeling but also for automated object to object-relational transformations in the MDA approach.

REFERENCES

Ambler, S. W. (2003). *Agile database techniques.* Wiley.

Badia, A. (2006). Relational, object-oriented and object-relational data models. In: Rivero, L. C., Doorn, J. H., & Ferragine, V. E. (Eds.). *Encyclopedia of database technologies and applications* (pp. 530-535). Idea Group Reference.

Booch, G., Rumbaugh, J., & Jacobson, I. (1998). *The unified modeling language user guide.* Addison-Wesley Longman.

Catell, R. G., Barry, D. K., Berler, M., & Eastman, J. (2000). *The object data standard: ODMG 3.0.* Morgan Kaufmann Publishers.

Database Language SQL – Part 2: Foundation. (1999). *ISO/IEC, 9075*(2), *1999* standard.

Database Language SQL – Part 2: Foundation. (2003). *ISO/IEC, 9075*(2), *2003* standard.

Eisenberg, A., Melton, J., Kulkarni, K., Michels, J. E., & Zemke, F. (2004). SQL: 2003 has been published. *ACM SIGMOD Record* 33(1), 119-126.

Grant, E. S., Cennamaneni, R., & Reza, H. (2006). Towards analyzing UML class diagram models to object-relational database systems transformation. In *Proceedings of the 24th IASTED International Multi-Conference Databases and Applications* (pp. 129-134). Innsbruck: ACTA Press.

Mapping Objects to Data Models with the UML. (2001). Rational Software Corporation, Santa Clara, CA, USA.

Marcos, E., Vela, B. & Cavero J. M (2001). Extending UML for object-relational database design. In M. Gogolla & C. Kobryn (eds.). *UML 2001* (pp. 225-239). SpringerVerlag.

Marcos, E., Vela, B., & Cavero, J. M. (2003). A methodological approach for object-relational database design using UML. *Software and Systems Modeling, 2*(1), 59-72.

Melton, J., & Simon, A. (2001). SQL: 1999. *Understanding relational language components.* Morgan Kaufmann Publishers.

OMG Unified Modeling Language Specification. Version 2.1.1. (2007). Retrieved October 6, 2007, from http://www.omg.org/technology/documents/formal/uml.htm.

Oracle Database Application Developer's Guide - Object-Relational Features. (2005). Retrieved October 6, 2007, from http://download.oracle.com/docs/cd/B19306_01/appdev.102/b14260.pdf.

Rational 98i Using Rose Oracle8. (1998). Santa Clara, CA: Rational Software Corporation.

Rational 2000e Using Rose Oracle8. (2000). Santa Clara, CA: Rational Software Corporation.

Using Rose Data Modeler (2001). Retrieved October 6, 2007, from ftp://ftp.software.ibm.com/software/rational/docs/v2002/Rose_dm.pdf.

Vara, J. M., Vela, B., Cavero, J. M., & Esperanza, M. (2007). Model transformation for object-relational database development. In *Proceedings of the 2007 ACM symposium on Applied computing* (pp. 1012-1119). Seoul: ACM Press.

Zendulka, J. (2001). Object-relational modeling in UML. In *Proceedings of the 4th International Conference on Information Systems Modelling* (pp. 17-24). Ostrava: MARQ.

Zendulka, J. (2006). Object-Relational Modeling in UML. In L. C. Rivero, J. H. Doorn, & V. E. *Ferragine (Eds.). Encyclopedia of database technologies and applications* (pp. 421-426). Idea Group Reference.

KEY TERMS

Collection Type: A composite value comprising elements of the same data type. SQL:1999 supports arrays, which are ordered and bounded collections of elements. SQL:2003 added multisets, which are unordered and unbounded collections.

Nested Table: A collection type available in Oracle SQL which is a multiset from the SQL standard point of view.

Object-Relational Data Model: Extends the relational data model by providing a richer type system including complex data types and object orientation.

Object-Relational Modeling: Modeling of an object-relational database schema. It requires using model elements that are available neither in classic data models nor in object models.

REF: A data type value of which references a row in a referenceable table (or object in Oracle).

Referenceable Table: A table row of which can be referenced by REF type values. The table is based on a structured user-defined type and comprises one extra column containing row (or object) identifiers generated automatically when a new row is inserted into the table. In Oracle, such a table is called an object table.

SQL:1999: The most recent major release of the SQL standard published in 1999, which is based on an object-relational model. Several minor extensions were added to the model in SQL:2003.

UML Profile: A predefined set of stereotypes, tagged values and constraints and notation icons that collectively specialize and tailor the UML for specific domain or process.

UML Stereotype: One of UML extensibility mechanisms. It is an extension of the vocabulary of the UML that allows creating new kinds of building blocks that are derived from existing ones.

User-Defined Type: A named data type defined by a user. It can contain a list of attributes,

in which case it is said to be a structured type (or object type in Oracle). It is an abstraction of a real-world entity. It can also provide explicitly defined methods that implement operations with the entity.

ENDNOTE

[1] Now IBM Rational Software and IBM Rational Rose®, respectively.

Chapter XX
Concept-Oriented Model

Alexandr Savinov
University of Bonn, Germany

INTRODUCTION

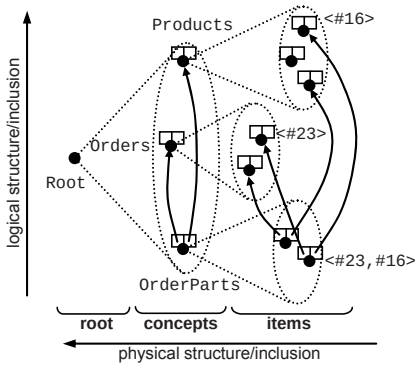
Background

The concept-oriented model (CoM) is a new approach to data modeling (Savinov, 2004) that is being developed along with concept-oriented programming (CoP) (Savinov, 2005a). Its major goal consists of providing simple and effective means for representing and manipulating multi-dimensional and hierarchical data while retaining the possibility to model how the data are represented physically. Thus, this model has two sides or flavors: logical and physical. From the point of view of logical structure, CoM belongs to a class of multidimensional models (Agrawal, Gupta, & Sarawagi, 1997; Gyssens & Lakshmanan, 1997; Li & Wang, 1996) and OLAP technologies (Berson & Smith, 1997). The main difference from the existing approaches is that CoM is based on the theory of ordered sets. Particularly, one source of inspiration when developing CoM was formal concept analysis (FCA) and lattice theory (Ganter & Wille, 1999).

Elements in the concept-oriented model are living among other elements within a multidimensional hierarchical structure (Savinov, 2005b). This means that any element has a number of parents and children. The direct and indirect neighbors determine its semantic properties while the element itself is thought of as an identifier. So the meaning of an element is distributed all over the model within the ordered structure and depends on its relative position among other elements. One important property of CoM that is absent in most other models is that it possesses canonical semantics. It makes many problem formulations and solutions much simpler because operations can be applied directly to the semantics of the whole model represented using primitive dimensions rather than to different local elements. In particular, it is very important for such a mechanism as grouping and aggregation (Savinov, 2006a), and constraint propagation and inference (Savinov, 2006b). In this sense, CoM is analogous to the universal relation model (URM) where all relations are assumed to be projections of a single relation (Fagin, Mendelzon, & Ullman, 1982; Kent, 1981; Maier, Ullman, & Vardi, 1984).

The multidimensional and hierarchical structure underlying the concept-oriented model can be used for navigational purposes (Savinov, 2005c). This means that data can be accessed by specifying a logical path rather than using joins. In this sense, CoM is similar to the functional data model (FDM; Gray, Kerschberg, King, &

Figure 1. Physical and logical structure of the model



Poulovassilis, 2004; Gray, King, & Kerschberg, 1999; Shipman, 1981). The difference is that the mechanism of logical navigation in CoM relies on the ordered structure of elements rather than using an arbitrary graph.

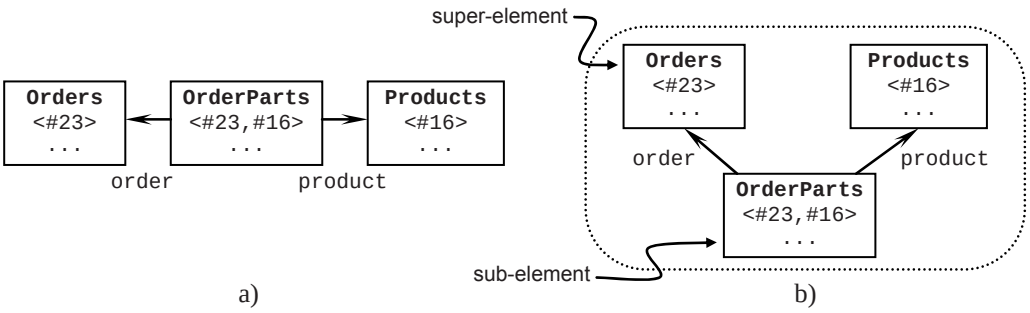
Duality

An important notion in the whole concept-oriented paradigm is that of duality. In CoM it exhibits itself via the existence of two structures called the physical (or hard) and the logical (or soft). In Figure 1, the physical structure spreads horizontally while the logical structure spreads vertically. Physical structure has a hierarchical form where each element has one permanent parent called also its physical context. For example, the model in Figure 1 has one root element (a database) that physically consists of three internal elements (tables), Orders, Products, and OrderParts, which in turn consist of their own internal elements (records). Logical structure has a multidimensional hierarchical form where each element has many parents that can change during their lifetime. In Figure 1 logical parents are denoted by arrows and a parent is positioned above its children. For example, element OrderParts belongs to two parents Orders and Products (while all the three elements physically belong to Root). Operations within physical structure mean creating or deleting elements while operations with elements within logical structure result in only a change of their properties. For data modeling it is important to understand that both of these structures can be used to represent information from the problem domain. Interestingly, physical structure is similar to the hierarchical data model while logical structure is similar to the network data model (with one important exception discussed in the next section being that its elements are ordered). In this sense, CoM can be viewed as uniting in one approach the two classical data models.

Order of Elements

The order of elements is of crucial importance in CoM; that is, it is precisely order that determines all its syntactic and semantic aspects. To define a concrete model using the concept-oriented approach, it is necessary to take a number of elements and then position them one under another without cycles. All other properties of the model can then be derived from this ordered structure.

Figure 2. Model as an arbitrary graph and as an ordered structure



In order to illustrate this property, let us consider a simple example consisting of two tables: Orders and Products. It is assumed that there is a many-to-many relationship between them where each order consists of many products and one product can be used in many orders. This relationship is stored in a separate table OrderParts, which has at least two columns pointing to related items from Orders and Products. Such a model can be represented as an entity-relationship diagram where tables are nodes (Figure 2a). However, elements of this diagram are not ordered and it is actually an arbitrary graph. In contrast, if we want to represent this model in the concept-oriented way, then these three tables have to be ordered as shown in Figure 2b. Here, table OrderParts references tables Orders and Products and hence it is positioned below both of them.

A position of each element in CoM determines its role and meaning. If this position changes, then the meaning of this element and of the whole model also changes. The relative position of elements has many interpretations. One of the most important of them allows us to bring the notion of object-attribute-value into the model (also referred to as subject-predicate-object). Namely, if one element is positioned above another element, then it is interpreted as a value while the second element is interpreted as an object (characterized by this value). The upward arrow between them is an attribute. Thus values reside always above the objects they characterize. Notice that to be a value or to be an object is a relative role. In other words, an element plays a role of value for lower level elements (subelements) and of an object for its higher level elements (superelements). This role is a consequence of the property of the structure to be ordered rather than a source assumption. In the first place, we assume that elements are ordered and only after that this order can be interpreted as object-attribute-value relationship among them (if needed and appropriate for the problem domain).

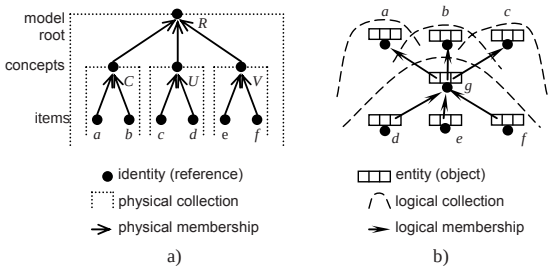
Another important interpretation of the order allows us to bring the notion of set into the model (more precisely, a logical collection). Namely, an element is interpreted as a collection or group of elements positioned below of it (subelements). In other words, each element in the model is interpreted as a collection of its subelements, and each element is a member of several collections represented by its superelements. The last statement combined with the previous interpretation as object-attribute-value has an interesting consequence: An attribute value is a collection consisting of all objects it characterizes. For example, one concrete product category is a collection consisting of all concrete products having this category.

MODEL DEFINITION

Syntax and Semantics

In the concept-oriented paradigm, it is assumed that all things and phenomena have two sides or flavors. For example, in CoP, the main programming construct, called a concept, consists of two classes: a reference class and an object class. In data modeling, this duality is used to distinguish identity modeling (how elements are represented and accessed) from entity modeling (how elements are characterized by other elements). In CoM, it is physical structure that is responsible for identity modeling while entity modeling is performed via logical structure. Another consequence of such a separation is that formally two types of element composition are distinguished: collection and combination. An element is then represented as consisting of a collection of other elements and a

Figure 3. An element is a pair consisting of one collection and one combination of other elements



combination of other elements from this model: $E = \{a, b, ...\}\langle c, d, ...\rangle$. Here $\{\}$ denotes a collection and $\langle\rangle$ denotes a combination. A collection can be viewed as a normal set with elements connected via logical *or* statements and identified by means of references (for example, tables with rows). A combination is analogous to fields of an object or columns of a table, which are identified by positions (offsets) and connected via logical *and* statements.

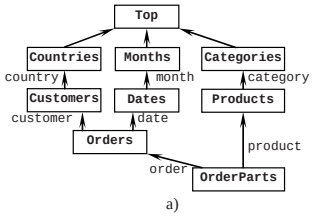
Using these terms, physical structure (Figure 3a) can be represented as a nested collection where any element has one parent: $R = \{C, U, V, ...\}\langle\rangle$, $C = \{a, b, ...\}\langle\rangle$, $U = \{c, d, ...\}\langle\rangle$, $V = \{e, f, ...\}\langle\rangle$. Physical structure can be easily produced by removing all properties (fields, columns, etc.) from elements. Physical inclusion can be thought of as inclusion by value; that is, we assume that any element has some physical position within its parent container. Physical inclusion can be used for grouping. For example, if $E = \{a, b, ...\}\langle\rangle$, then elements *a* and *b* physically belong to one group *E*. One property of physical structure is that it is immutable because elements cannot change their parent group and the only possibility to move an element between groups is to clone it. However, the main use of physical hierarchy consists of representing and accessing elements of the model while ignoring their semantic properties.

Logical structure (Figure 3b) arises when elements get some properties. The combinational part is not empty and elements are referencing other elements of the model. Such a referencing is a method of mutual characterization. For example, element $g = \{...\}\langle a, b, c, ...\rangle$ is referencing elements *a*, *b*, and *c* and we say that *g* is characterized by values *a*, *b*, and *c*. Logical structure provides the second (dual) method for grouping using the following principle: An element belongs to all elements it combines (references). In other words, properties of an object indicate groups in which it is a member. On the other hand, an object is a group for all other objects that reference it as the value of some attribute. For example, if $g = \{...\}\langle a, b, c\rangle$, then *g* belongs to three groups: *a*, *b*, and *c*. In contrast to physical grouping, logical grouping has two advantages: An element may belong to many groups simultaneously and this structure is not constant so that an element can change its parent groups by changing its attribute values.

From the point of view of physical structure, three types of the concept-oriented models are distinguished: (a) The one-level model has one root and a number of data items in it, (b) the two-level model has one root and a number of concepts in it, each of them having a number of data items, and (c) the multilevel model has an arbitrary number of levels in its physical hierarchy. In this article, only the two-level model is described, which is defined as consisting of the following elements:

- **[Root]** One root element *R* is a physical collection of *N* concepts: $R = \{C_1, C_2, ..., C_N\}$.
- **[Syntax]** Each concept is (a) a physical collection of data items (or concept instances) and (b) a combination of other concepts called superconcepts (while this concept is called a subconcept): $C = \{i_1, i_2, ...\}\langle C_1, C_2, ..., C_N\rangle \in R$.
- **[Semantics]** Each data item is (a) the empty physical collection, and (b) a combination

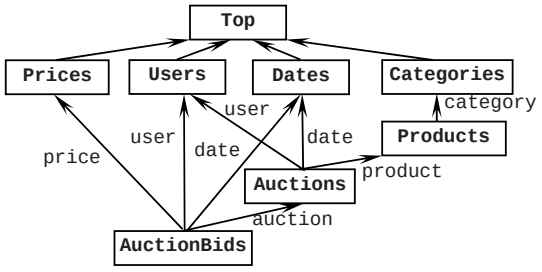
Figure 4. An example of the model syntactic structure and its primitive semantics



Country	Month	Category
Germany	July	Drinks
Germany	August	Cars
Argentina	July	Food
Argentina	April	Drinks
Japan	August	Food
Japan	June	Cars

b)

Figure 5. An auction example



of other data items from the model called superitems (while this item is called a subitem): $i = \{\}\langle i_1, i_2, ..., i_n\rangle \in C$.

- **[Syntactic Constraints]** Each data item from a concept may combine only items from its superconcepts: If $i = \{\}\ \langle i_1, i_2, ..., i_n\rangle \in C = \langle C_1, C_2, ..., C_N\rangle$, then $i_j \in C_j$, $j = 1, 2, ..., n$.
- **[Top and Bottom]** If a concept does not have a superconcept, then this role is played by one common top concept with its direct subconcepts called primitive concepts; and if a concept does not have a subconcept, then this role is played by one common bottom concept.
- **[Cycles]** Cycles in subconcept-superconcept relations and subitem-superitem relations are not allowed.

Figure 4a is an example of logical structure where each concept is a combination of its superconcepts. For example, the concept Orders is a combination of superconcepts Customers and Dates. It has one subconcept OrderParts, which is also the bottom concept of the model. In this case, an order item (instance of Orders) is logically a member of one customer item and one date item. At the same time, one order item logically includes a number of order parts that are its subitems. One order part is logically included into one order and one product. Figure 4b is an example of this model primitive semantics described in the next section.

Model Dimensionality

A named link from subconcept to a superconcept is referred to as a dimension. A dimension can be also viewed as a unique position of a superconcept in the definition of subconcept: $C = \langle x_1 : C_1, x_2 : C_2, ..., x_n : C_n\rangle$. Super-concepts $C_1, C_2, ..., C_n$ are called domains (or ranges) for dimensions $x_1, x_2, ..., x_n$, $C_j = \text{Dom}(x_j)$. The model syntactic structure can be represented by a directed acyclic graph where nodes are concepts and edges are dimensions. A dimension $x = x^1.x^2.\cdots.x^k$ of rank k is a sequence of k dimensions where each next dimension belongs to the domain of the previous one. Dimensions will be frequently prefixed by the first domain: $C.x^1.x^2.\cdots.x^k$. Each dimension is represented by one path in the concept graph. The number of edges in the path is the dimension rank. A dimension with the primitive domain is referred to as a primitive dimension.

For example, Auctions.product.category (Figure 5) is a primitive dimension of rank 2 from the source concept Auctions to its superconcept Categories. There may be several different dimensions (paths) between a concept and one domain (superconcept), which are called bounding dimensions. The number of different primitive dimensions of a concept is referred to as the concept primitive dimensionality. The length of the longest dimension of a concept is referred to as concept rank. The dimensionality and rank of the whole model are equal to that of the bottom concept. Thus, any concept-oriented model is characterized by two parameters: (a) the model rank describing its hierarchy (depth), and (b) the model dimensionality describing its multidimensionality (width). The models in Figure 4 and Figure 5 are three dimensional and six dimensional, respectively. However, both have rank 3.

Inverse dimension is a dimension with the opposite direction; that is, where a dimension starts, the inverse dimension has its domain, and where a dimension ends, the inverse dimension has its start. In a concept graph, inverse dimension is a path from a superconcept to one of its subconcepts. Inverse dimensions do not have their own identifiers. Instead, we apply an operator of inversion by enclosing the corresponding dimension into curly brackets. If $C.x^1.x^2.\cdots.x^k$ is a dimension of concept C with rank k, then $\{C.x^1.x^2.\cdots.x^k\}$ is the inverse dimension of concept $\text{Dom}(x^k)$ with the domain in C and the same rank k. An important thing is that any concept is characterized by a set of dimensions leading to superconcepts and a set of inverse dimensions leading to subconcepts. For example, an order (Figure 4) is characterized by two dimensions (date and customer) as well as one inverse dimension {OrderParts.order}. Such a duality is one of the distinguishing features of

CoM because it allows us to characterize items as a combination of (more general) superitems and a collection of (more specific) subitems.

Dimensions are very convenient in describing model semantics. We assume that subconcepts contain more specific information and inherit information from their superconcepts. Consequently, the bottom concept is assumed to contain the whole model semantics. If a set of items is represented using only the primitive dimensions of this concept, then such a representation is referred to as primitive semantics. For example, in Figure 4b, items from the bottom concept OrderParts are written using three primitive dimensions order. customer.country, order.date.month, and product. category. The first row in the table represents item <#23, #16> from OrderParts referencing order #23. This order #23 in turn references a customer from Germany and has some date in July. It also references product #16, which belongs to Drinks as a category. So we write <Germany, July, Drinks> in the table with the primitive semantics. Canonical semantics of the model contains the primitive semantics of its bottom concept as well as that of its superconcepts (recursively). Primitive and canonical semantics can be viewed as expanded or flat representations of model data.

Projection and Deprojection

Given an item or a set of items, it is possible to get related items by specifying some path in the concept graph. Informally, if we move up along a dimension to superitems, then it is thought of as an operation of projection. If we move down along an inverse dimension to subitems, then it is deprojection. To specify the path, dimension names and the operation of inversion are used.

If d is a dimension of C with the domain in superconcept $U = \text{Dom}(d)$, then operation $I \to d$, $I \subseteq C$ is referred to as a projection of items from I along dimension d. It returns a set of superitems referenced by items from I: $I \to d = \{u \in U \mid i.d = u, i \in I \subseteq C\}$. Each item from U can be included into the result collection (projection) only one time. If we need to include superitems as many times as they are referenced, then a double arrow has to be used; that is, $I \Rightarrow d$ includes all referenced superitems from U even if they occur more than once. The operation of projection (arrow) can be applied consecutively. For example, if A is a collection of today's auctions, then A->product->category will return a set of today's categories while A=>product=>category will return categories for all auctions in A (as many categories as we have auctions). If P is a subset of order parts, then projection P->order->customer->country is a set of countries (Figure 4).

If $\{d\}$ is an inverse dimension of C with the domain in subconcept $S = \text{Dom}(\{d\})$, then deprojection of I to S along $\{d\}$ consists of all subitems that reference items from I via dimension d: $I \to \{d\} = \{s \in S \mid s.d = i, i \in I \subseteq C\}$. For example, if C is a set of auction product categories, then C->{Auctions->product->category} is a set of all auctions with the products having these categories. Given month m, we can get all related orders by deprojecting it onto concept Orders: m->{Orders->date->month} (Figure 4).

MODEL USE

Logical Navigation

One of the main achievements of the relational model of data based on using sets and set operations is that it provided independence from physical data structure and thus successfully solved the problem of physical navigation. However, it failed to solve the problem of logical navigation. This means that even though a data item is not physically bound to some access path, we still need to encode a logical access path in each query that describes how this item can be reached. In order to get a set of records using SQL, it is necessary to include long and complex join conditions as part of each query. In contrast, if the mechanism of logical navigation was incorporated into a data model, then we would need only to specify table names and column names leading to the target records. All the rest would then be done

by the database itself. Currently, the mechanism of logical navigation is provided by the network model, object-oriented model, functional model, and some others. An important distinguishing feature of the concept-oriented model is that its logical navigation mechanism is based on ordering the elements.

The access path is a sequence of dimensions and inverse dimensions separated by single or double arrows where each next operation is applied to the result collection returned by the previous operation. An access path has a zigzag form in the concept graph where dimensions move up to a superconcept while inverse dimensions move down to a subconcept. Moving up means getting more general elements while moving down means getting more specific elements. For example, we can choose a product item p and then find its auctions, for which we can also find their users and finally find the bids of these users (Figure 5).

p->{Auctions.product} ->user ->{AuctionBids.user}

It is possible to restrict items that are returned by a deprojection operation by providing a condition that all items from the domain subconcept have to satisfy the following:

$I \rightarrow \{s : S \rightarrow d \mid f(s)\} = \{s \in S \mid (s.d = i) \wedge (f(s) = \text{true}), i \in I \subseteq C\}$.

Here, d is a bounding dimension from subconcept S to the source collection I; s is an instance variable that takes all values from set S, and the predicate f (separated by a bar) must be true for all items s included into the result collection (deprojection). The following access path is a modified version of the previous example.

Figure 6. Constraint propagation

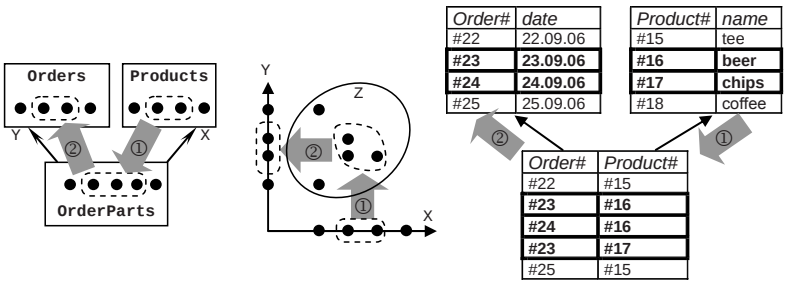
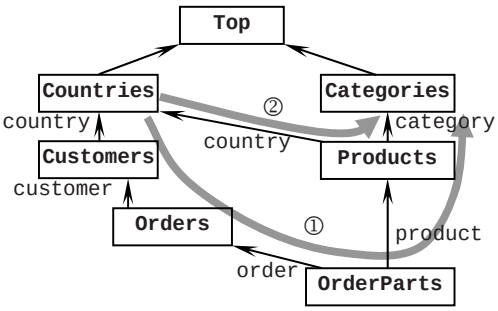


Figure 7. Multiple constraint propagation paths



p->{a in Auctions.product | a.date==today} ->user ->{AuctionBids.user}

It will return all today's auctions for the selected product item p.

Dimensions provide a simple method to logically navigate over the model using its structure. In many cases, it is necessary to have more complex properties such as aggregated characteristics of items computed from related items. This can be done by defining a derived property of a concept, which is a named definition of a query returning one or more items for each item from this concept. For example, we could define a derived property allBids of concept Auctions as follows.

Auctions.allBids = this->{ AuctionBids->auction }

(*this* is an instance variable referencing the current item of the concept)

Grouping and Aggregation

In CoM, any item is interpreted as a group consisting of its subitems, which are items that reference this item. In other words, a value is a group for all objects it characterizes. Subitems can be retrieved using inverse dimensions or derived properties, and this mechanism can be used to aggregate data by applying some aggregate function to the collection of data items. For example, the following query returns the maximum price.

```
Auctions.maxBid = max( this.allBids.price )
```

Here, we get a set of all bids by applying existing property allBids to the current item, then get their prices via a dot operation, and then find the maximum price. In the same way we can define a derived property for computing the mean price for 10 days for one product category.

```
Category.meanPriceForTenDays = avg({ab in
AuctionBids->auction->product->category |
ab.auction.date > today-10}.price)
```

The mechanism of grouping and aggregation can be generalized into a multidimensional form, which is very useful for the purpose of online analytical processing (Savinov, 2006a).

Automatic Constraint Propagation

Normally, data are retrieved by specifying precise criteria such as table names, column names, join conditions, and so forth. However, in many cases, it is enough to impose some constraints and indicate the type of result we want to get—all the rest will be done automatically. In particular, the source constraints will be automatically propagated over the model and only related data items will be returned. The mechanism of constraint propagation in CoM is based on the idea that if an item is removed, then all its subitems are also removed. For example, if some concrete product item is deleted, then all order parts from its subconcept have to be deleted. The whole procedure consists then of the following two steps.

1. Initial constraints imposed on concepts X_1, ..., X_n are propagated down to and imposed on the most specific concept Z using deprojection.
2. The constrained semantics of concept Z is propagated up to the target concept Y using projection.

An example of this procedure is shown in Figure 6. Constraints are imposed on the concept Products by selecting a number of product items such as beer and chips. In order to get related orders from the concept Orders, these products have to be propagated down to OrderParts using deprojection. All three selected order parts reference only the two specified products. On the second step, these three order parts are projected on the concept Orders. They reference only two orders, #23 and #24, which are the result of this query. This procedure is analogous to inference in multidimensional space (Figure 6, center). Constraints are imposed by selecting a subset of values along axis X. The available dependencies are represented as a set of points $Z \subset X \times Y$. Their intersection $X \cap Z$ is then projected on target axis Y and produces the result.

Notice that related orders can always be obtained by manually specifying an appropriate access path. However, an amazing property of the described procedure is that it does not require any access path and propagates constraints automatically. We need only to impose constraints and then indicate some target concept. The query for getting related orders might be written as follows using SQL-like syntax.

```
SELECT o.* FROM Orders o, Products p WHERE
p.name = 'beer' OR p.name = 'chips'
```

Notice that this query does not have any information on how the concepts are connected.

There exist models with many alternative constraint propagation paths between the source and target concepts. For example, if we select a country and want to get all related product categories (Figure 7), then there are two interpretations for this query: (a) Get all categories for products ordered from the selected countries, and (b) get all categories for products made in the selected countries. In this case, it is necessary to provide some hints that remove this ambiguity. For example, the query could provide an intermediate concept for constraint propagation in the keyword *via*.

```
SELECT cat.* FROM Categories cat, Countries c
WHERE c.name = 'Germany' VIA OrderParts
```

Another difficulty consists in the absence of an access path between the source and the target concepts. This problem can be solved using a zigzag access path or by imposing implicit constraints (Savinov, 2006b).

CONCLUSION

In this article, we described main properties of the concept-oriented data model and demonstrated how it can be used. This model has a number of advantages over the existing approaches, especially in the area of conceptual modeling and analytical data processing. It is an integrated full-featured model that can be applied to a wide range of tasks. At the same time it is a rather simple approach that uses only a few basic notions to derive many important data modeling mechanisms and manipulation techniques.

In the future, CoM will be more tightly integrated with CoP (Savinov, 2005a). In particular, the definitions of concept in CoM and CoP should be more compatible so that one construct can be used for both data modeling and programming. Another important task consists in defining a concept-oriented query language (CoQL) that also should be compatible with the principles of CoP.

REFERENCES

Agrawal, R., Gupta, A., & Sarawagi, S. (1997). Modeling multidimensional databases. *13th International Conference on Data Engineering (ICDE'97)* (pp. 232-243).

Berson, A., & Smith, S. J. (1997). *Data warehousing, data mining, and OLAP.* New York: McGraw-Hill.

Fagin, R., Mendelzon, A. O., & Ullman, J. D. (1982). A simplified universal relation assumption and its properties. *ACM Transactions on Database Systems, 7*(3), 343-360.

Ganter, B., & Wille, R. (1999). *Formal concept analysis: Mathematical foundations.* Springer.

Gray, P. M. D., Kerschberg, L., King, P., & Poulovassilis, A. (Eds.). (2004). *The functional approach to data management: Modeling, analyzing, and integrating heterogeneous data*. Heidelberg, Germany: Springer.

Gray, P. M. D., King, P. J. H., & Kerschberg, L. (Eds.). (1999). Functional approach to intelligent information systems. *Journal of Intelligent Information Systems, 12*, 107-111.

Gyssens, M., & Lakshmanan, L. V. S. (1997). A foundation for multi-dimensional databases. *VLDB'97* (pp. 106-115).

Kent, W. (1981). Consequences of assuming a universal relation. *ACM Transactions on Database Systems, 6*(4), 539-556.

Li, C., & Wang, X. S. (1996). A data model for supporting on-line analytical processing. *Proceedings of the Conference on Information and Knowledge Management* (pp. 81-88).

Maier, D., Ullman, J. D., & Vardi, M. Y. (1984). On the foundation of the universal relation model. *ACM Transactions on Database Systems (TODS), 9*(2), 283-308.

Savinov, A. (2004). *Principles of the concept-oriented data model.* Moldavian Academy of Sciences, Institute of Mathematics and Informatics.

Savinov, A. (2005a). Concept as a generalization of class and principles of the concept-oriented programming. *Computer Science Journal of Moldova, 13*(3), 292-335.

Savinov, A. (2005b). Hierarchical multidimensional modelling in the concept-oriented data model. *3rd International Conference on Concept Lattices and their Applications (CLA'05)* (pp. 123-134).

Savinov, A. (2005c). Logical navigation in the concept-oriented data model. *Journal of Conceptual Modeling, 36*. Retrieved from http://www.inconcept.com/jcm

Savinov, A. (2006a). Grouping and aggregation in the concept-oriented data model. *21st Annual ACM Symposium on Applied Computing (SAC'06)* (pp. 482-486).

Savinov, A. (2006b). Query by constraint propagation in the concept-oriented data model. *Computer Science Journal of Moldova, 14*(2), 219-238.

Shipman, D. W. (1981). The functional data model and the data language DAPLEX. *ACM Transactions on Database Systems, 6*(1), 140-173.

KEY TERMS

Access Path: A sequence of projection and deprojection operations applied to the source subset of items. Access paths are used for the logical navigation and retrieval of related items and can include additional constraints.

Bottom Concept: A direct or indirect subconcept for any other concept in the model. It is the most specific concept in the model that can be introduced formally if it does not exist.

Concept: A data modeling construct that physically includes a number of data items and logically has a number of parent concepts and child concepts. Parent concepts are referred to as superconcepts while child concepts are referred to as subconcepts.

Deprojection: An operation applied to a set of items that returns a subset of their subitems referencing the source items along the specified dimension. Multidimensional deprojection uses several bounding dimensions leading from the selected subconcept to the superconcept.

Dimension: A named link between this concept and some of its direct superconcepts. It is analogous to a column. The superconcept in this case is referred to as a domain of this dimension.

Dimension of Rank *k*: A sequence of dimensions leading from this concept to some of its superconcepts where each next dimension belongs to the domain of the previous dimension. Dimension of rank *k* can be viewed as an upward path in the concept graph. Dimensions are interpreted as single-valued attributes.

Inverse Dimension: A dimension with the opposite direction. Inverse dimension can be viewed as a downward path in the concept graph. Inverse dimensions are interpreted as multivalued attributes.

Primitive Concept: A direct subconcept of a top concept.

Projection: An operation applied to a set of items that returns a subset of their superitems referenced by the source items along the specified dimension. Multidimensional projection uses many bounding dimensions leading from the source concept to the selected superconcept.

Top Concept: A direct or indirect superconcept for any other concept in the model. It is the most general concept that is introduced formally.

Chapter XXI
Database Reverse Engineering

Jean-Luc Hainaut
University of Namur, Belgium

Jean Henrard
REVER s.a., Belgium

Didier Roland
REVER s.a., Belgium

Jean-Marc Hick
REVER s.a., Belgium

Vincent Englebert
University of Namur, Belgium

INTRODUCTION

Database reverse engineering consists of recovering the abstract descriptions of files and databases of legacy information systems. A legacy information system can be defined as a "data-intensive application, such as [a] business system based on hundreds or thousands of data files (or tables), that significantly resists modifications and changes" (Brodie & Stonebraker, 1995). The objective of database reverse engineering is to recover the logical and conceptual descriptions, or schemas, of the permanent data of a legacy information system, that is, its database, be it implemented as a set of files or through an actual database management system.

The logical schema is the technology-dependent (e.g., relational) description of the database structures while the conceptual schema is the abstract, technology-independent description of their semantics.

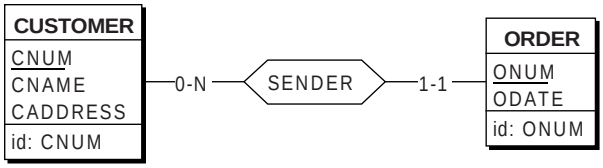
Database reverse engineering often is the first steps of broader engineering projects. Indeed, rebuilding the precise documentation of a legacy database is an absolute prerequisite before migrating, reengineering, maintaining or extending it, or merging it with other databases.

The current commercial offering in CASE tools poorly supports database reverse engineering. Generally, it reduces to the straightforward derivation of a conceptual schema such as that of Figure 1 from the following DDL code.

Figure 1. A naive view of data reverse engineering

```
create table CUSTOMER (
  CNUM decimal(10) not null,
  CNAME varchar(60) not null,
  CADDRESS varchar(100) not null,
primary key (CNUM))
```

```
create table ORDER (
  ONUM decimal(12) not null,
  SENDER decimal(10) not null,
  ODATE date not null,
primary key (ONUM),
foreign key (CNUM) references CUSTOMER))
```



Unfortunately, actual database reverse engineering often is closer to deriving the conceptual schema of Figure 2 from the following sections of COBOL code, using meaningless names that do not declare compound fields or foreign keys.

Getting such a result obviously requires additional sources of information, which may prove more difficult to analyze than mere DDL statements. Untranslated (implicit) data structures and constraints, empirical implementation approaches and techniques, optimization constructs, ill-designed schemas, and, above all, the lack of up-to-date documentation are some of the difficulties that the analysts will face when trying to understand existing databases.

The goal of this article is to describe the problems that arise when one tries to rebuild the documentation of a legacy database and the methods, techniques, and tools through which these problems can be solved. A more in-depth analysis can be found in Hainaut (2002).

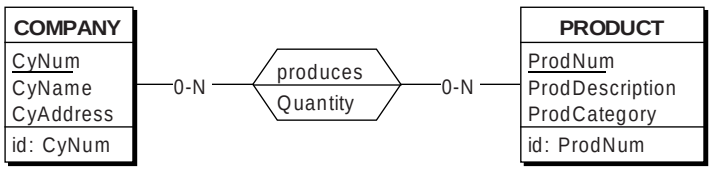
BACKGROUND: STATE OF THE ART AND KEY PROBLEMS

Database reverse engineering has been recognized to be a specific problem in the ’80s, notably in Casanova and Amaral De Sa (1984), Davis and Arora (1985), and Navathe (1988). These pioneer-

Figure 2. A more realistic view of data reverse engineering

```
select CF008 assign to DSK02:P12
organization is indexed
record key is K1 of REC-CF008-1.
select PF0S assign to DSK02:P27
organization is indexed
record key is K1 of REC-PFOS-1.
fd CF008.
record is REC-CF008-1.
01 REC-CF008-1.
   02 K1 pic 9(6).
   02 filler pic X(125).
```

```
fd PF0S.
records are REC-PF0S-1,REC-PF0S-2.
01 REC-PF0S-1.
   02 K1.
      03 K11 pic X(9).
      03 filler pic 9(6)
   02 filler pic X(180).
01 REC-PF0S-2.
   02 filler pic X(35).
```



Box 1.

```
01 CUSTOMER.
    02 C-KEY  pic X(14).
    02 filler pic X(57).
```

```
01 CUSTOMER.
  02 C-KEY.
     03 ZIP-CODE pic X(8).
     03 SER-NUM  pic 9(6).
  02 NAME     pic X(15).
  02 ADDRESS  pic X(30).
  02 ACCOUNT  pic 9(12).
```

ing approaches were based on simple rules such as those illustrated in Figure 1, which work nicely with databases designed in a clean and disciplined way. A second generation of methodologies coped with physical schemas resulting from empirical design. More complex design rules were identified and interpreted (Blaha & Premerlani, 1995), structured and comprehensive approaches were developed (Edwards & Munro, 1995; Hainaut, Chandelon, Tonneau, & Joris, 1993), and the first industrial tools appeared (e.g., Bachman's Reengineering Tool). Many contributions were published in the '90s, addressing practically all the legacy technologies and exploiting such sources of information as application source code, database contents, or application user interfaces. Among synthesis publications, let us mention Davis and Aiken (2000), the first tentative history of this discipline.

These second-generation approaches were faced with two families of problems that we will briefly discuss here, namely, the elicitation of implicit constructs and the semantic interpretation of logical schemas.

The Implicit Construct Problem

One of the hardest problems reverse engineers have to face is eliciting implicit constructs. The latter are data structures or data properties, such as integrity constraints, that are an integral part of the database, though they have not been explicitly declared in the DDL specifications, either because they have been discarded during the implementation phase, or because they have been implemented by other means, such as through validation code in the application programs. Let us describe two common patterns.

Implicit Field or Record Structure. Ideally, the record fields of the database should be identified and given meaningful names. However, compound fields or even whole record structures often are expressed as anonymous or unstructured fields so that their intended structure is left implicit. The following DDL code sample shows (left) the code that declares the database record type CUSTOMER, the actual intended structure of which is at the right. The application programs generally recover the actual structure by storing records in local variables that have been given the correct detailed decomposition.

Implicit Foreign Keys. Interrecord associations most often are translated into foreign keys in relational databases or into navigable parent-child associations in hierarchical and network databases. Databases made up of standard files also include foreign keys, that is, fields used to identify a record in another file, but these keys cannot be declared in the file system DDL and are left implicit. Surprisingly, hierarchical, network, and even relational databases often include implicit foreign keys. There are several reasons to that, such as awkward or lazy design, backward compatibility (e.g., compliance with Oracle 5, which ignores primary and foreign keys), and nonstandard ways to control referential integrity (e.g., simulation of an unavailable delete mode). Generally, the application programs include code fragments that ensure that data that violate referential integrity are identified and coped with. For instance, each statement that stores an ORDER record is preceded by a code section that verifies that the CUSTOMER file includes a record for the customer that issued the order.

Figure 3. Recovering ISA hierarchies from record types sharing common fields

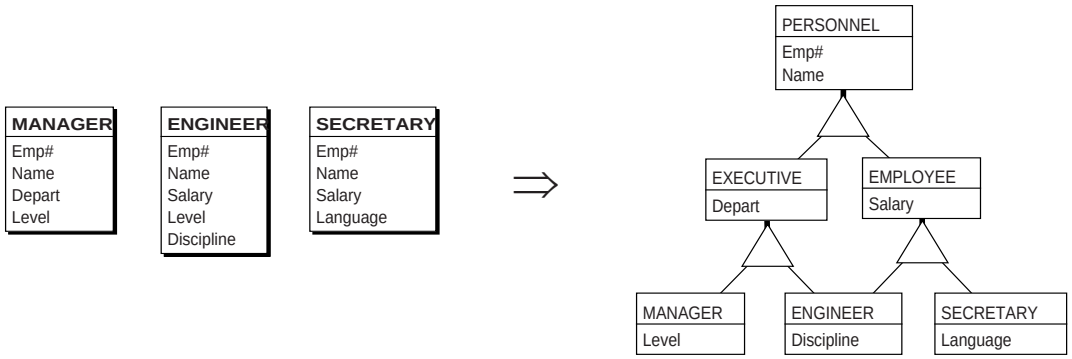
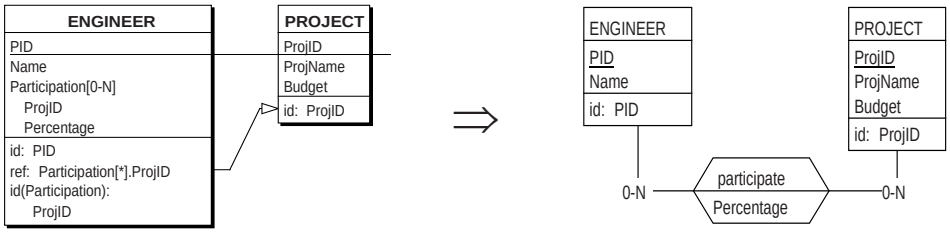


Figure 4. Expressing a complex multivalued compound attribute as a many-to-many relationship type



Semantic Interpretation of Logical Constructs

The process of recovering the intended semantics of the logical schema (conceptual), called interpretation or conceptualization, is fairly straightforward for small and simple database schemas, but it can prove complex for large and aging databases.

This process appears to be the inverse of the logical design of a database, the process through which a conceptual schema is translated into a technology-dependent model, such as a relational schema. Logical design has been described (Batini, Ceri, & Navathe, 1992) as a sequence of transformation rules, so that the conceptualization process could be defined by reversing these transformations (Hainaut, 2006). This is valid to a large extent, provided we know which rules have been applied when the database was developed. What makes this idea more complex than expected is that many legacy databases have been developed through empirical approaches, based on unusual and sometimes esoteric rules (Blaha & Premerlani, 1995). We illustrate this process through two common but nontrivial examples of reverse transformations.

Recovering ISA Relations

Since most legacy data management systems do not include explicit representations of supertype or subtype hierarchies, the latter must be converted into simpler constructs. One of the most popular techniques consists of implementing the bottom-most entity types through downward inheritance. Rebuilding the hierarchies from this implementation can be performed by such techniques as formal concept analysis (Figure 3).

Recovering Relationship Types

There are many different techniques to implement relationship types in logical models. Figure 4 shows a typical implementation of a many-to-many relationship type as a multivalued, com-

Figure 5. A typical data reverse engineering project

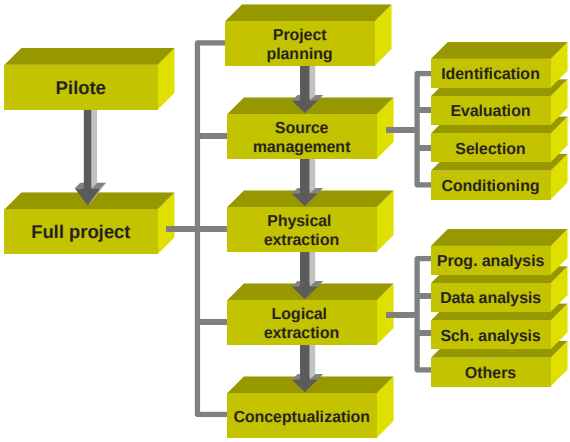
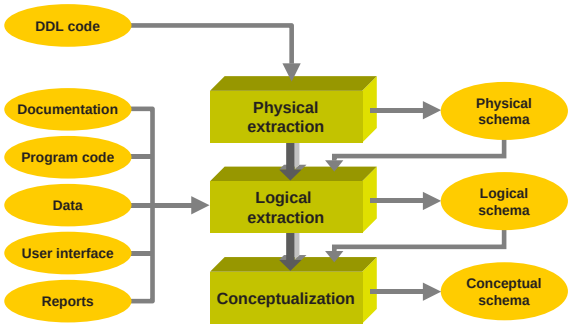


Figure 6. The main processes and products of database reverse engineering



pound field incorporated in one of its members. The source relationship type is recovered through a complex transformation.

THE PROCESSES OF DATABASE REVERSE ENGINEERING

Basically, reverse engineering a database consists of reversing its design process (Baxter & Mehlich, 2000). The latter can be decomposed into three main phases, namely, logical design, which transforms the conceptual schema into a DBMS- (database management system) dependent logical schema; physical design, through which technical constructs, such as indexes, are added and parameterized; and coding, which translates the physical schema into DDL code. Starting from the DDL code, as well as from other information sources, rebuilding the conceptual schema can also be performed in three phases: physical schema extraction (the inverse of coding), logical schema extraction (the inverse of physical design), and conceptualization (the inverse of logical design).

In empirical approaches, the standard design phases often are interleaved. This makes reverse engineering both more complex and less deterministic.

In this section, we first discuss the architecture of a reverse engineering project. Then, we describe the three main processes identified above. Finally, we discuss the role of reverse engineering tools.

Database Reverse Engineering Projects

Carrying out a pilot project first is common practice in risky processes. Indeed, it provides precise measures on the quality of the information sources, on the quality of the database, on the skills required, and finally on the resources needed in terms of money, time, and manpower (Aiken, 1996).

The project itself includes two preliminary phases, namely, project planning and management of the information sources (Figure 5). The three next phases are those mentioned here.

Physical Structure Extraction

Physical extraction is aimed at building the physical schema that is explicitly declared by the DDL code (or the data dictionary contents) of the database (Figure 6). This extraction generally uses a parser that stores the physical data structures in some repository. For instance, the contents of an SQL- (structured query language) DDL code is expressed in a graphical view that shows the tables; columns; primary, secondary, and foreign keys; indexes; table spaces; check predicates; triggers; and stored procedures.

This process can be more complex when useful information has also been coded in views instead of in the global schema. This will be the case for standard files (for which there is no global schema but a large collection of record descriptions included in the program source code), for CODASYL databases (subschemas) and for relational databases (views). These partial schemas are extracted then integrated.

Logical Structure Extraction

The goal of this process is to recover the complete logical schema of the database. The starting point is the physical schema extracted from the DDL code. It is then enriched with the implicit constructs that are recovered through various analysis techniques. Actual case studies show that the DDL code, and therefore the physical schema, conveys less than half the semantics of the logical schema, hence the importance of this phase, too often overlooked in the literature and in commercial CASE proposals.

The first constructs to be made explicit are the exact field and record structures, the unique keys of record types and of multivalued fields, and the foreign keys. Figure 4 provides an illustration of the latter two concepts. The subfield ProjID of compound field Participation in record type ENGINEER plays two roles. First, its values are unique among all the Participation instances of a given ENGINEER record. Second, each of its values references a PROJECT record. Therefore, this field is an identifier for the values of multivalued attribute Participation and a multivalued foreign key targeting PROJECT. No DDL can express these properties so that they basically are implicit. Additional constructs can be looked for as well, such as functional dependencies, value domains, optional fields, or more meaningful record and field names.

Among the information sources and techniques that can be used to recover implicit constructs, let us mention the following.

- **The physical schema.** Missing data structures and constraints can be inferred from existing structural patterns. For instance, names can suggest roles (unique keys, foreign key), data types, or relationships between data.
- **Technical/physical constructs.** There can be some correlation between logical constructs and their technical implementation. For instance, a foreign key is often supported by an index.
- **Application programs.** The way data are used, transformed, and managed in the application programs brings essential information on the structural properties of these data. Analyzing the source code of these programs requires techniques such as dataflow analysis, dependency analysis, programming cliché analysis, and program slicing, borrowed from the program understanding discipline (Henrard, 2003). The fine-grained analysis of DML statements can provide hints on data structures' properties (Andersson, 1994; Petit, Kouloumdjian, Bouliaut, & Toumani, 1994).
- **Screen/form/report layout.** A screen form or a structured report is a derived view of the data. Its layout as well as labels and comments can bring essential information on the data (Heeseok & Cheonsoo, 2000).
- **Data dictionaries and CASE repositories.** Third-party or in-house data dictionaries, as well as CASE tools, record and maintain essential descriptions of the information of an organization, including the file and database structures.
- **Data.** The data themselves can exhibit regular patterns, uniqueness, or inclusion properties that can be used to infer implicit structures and constraints.
- **Nondatabase sources.** Small volumes of data can be implemented with general-purpose software such as spreadsheet and word processors. In addition, semistructured documents are increasingly considered a source of complex data that also need to

be reverse engineered. Indeed, large text databases can be implemented according to a representation standard such as SGML, XML (extensible markup language), or HTML (hypertext markup language) that can be considered special-purpose DDL.

- And of course, the current documentation, if any.

Data Structure Conceptualization

This process extracts from the complete logical schema a tentative conceptual schema that represents the likeliest semantics of the former (Figure 6). It mainly relies on transformational techniques that undo the effect of the (often implicit) logical design process.

This complex process is decomposed in three subprocesses, namely, untranslation, de-optimization, and conceptual normalization.

The untranslation process consists of reversing the transformations that have been used to draw the logical schema from the conceptual schema. For instance, each foreign key is interpreted as the implementation of a many-to-one relationship type. This process relies on a solid knowledge about the rules and heuristics that have been used to design the database. These rules can be quite standard, which makes the process fairly straightforward, but they can also be specific to the company or even to the developer in charge of the database, in which case, reverse engineering can be quite tricky, especially if the developer has left the company. The main constructs that have to be recovered are relationship types (Figure 4), supertype and subtype hierarchies (Figure 3), multivalued attributes, compound attributes, and optional attributes.

The de-optimization process removes the trace of all the optimization techniques that have been used to improve the performance of the database. Among the most common optimization constructs, let us mention the redundancies that have to be identified and discarded, the unnormalized data structures that must be split, and horizontal and vertical partitioning that must be identified and undone.

The goal of the conceptual normalization process is to improve the expressiveness, the simplicity, the readability, and the extendability of the conceptual schema (Batini et al., 1992).

DATABASE REVERSE ENGINEERING TOOLS

Due to the complexity and to the size of the information sources, database reverse engineering cannot be performed without the support of specific tools, the requirements of which have been detailed in Hainaut, Roland, Hick, Henrard, and Englebert (1996). They generally are integrated into standard CASE tools. Unfortunately, they offer only limited reverse engineering functions. At best, CASE tools include elementary parsers for SQL databases, foreign-key elicitation under strong assumptions, and some primitive standard foreign-key transformations. None can cope with complex reverse engineering projects. The most advanced tool probably is DB-MAIN (http://www.info.fundp.ac.be/libd). It includes components that address most issues discussed above, such as code parsing, code analysis, schema analysis, and schema transformation.

TRENDS AND PERSPECTIVES

Database reverse engineering most often is the first step of larger scope projects such as system reengineering and migration. In system migration a legacy database as well as all its application programs is ported to a modern platform, a process that is being explored but not yet fully automated. The formal mapping between the (legacy) source schema and the (new) target schema includes enough information to automatically generate the data migration procedures, so that the new database could be fully converted to the new technology (Hick, 2001). In addition, this mapping also provides enough information to automatically restructure the legacy application programs so that they can run against the new database (Clève, 2006).

The second trend that is shaping important development in reverse engineering concerns techniques for semistructured data structures. In particular, Web reverse engineering techniques have been proposed to extract structured data from Web pages, generally through Web wrappers that are built by reverse engineering techniques (Chung, Gertz, & Sundaresan, 2002).

CONCLUSION

Many reverse engineering problems are now identified and formally described. For most of them, we are provided with techniques to solve them, at least partially. However, solving them in every possible situation is a goal that is far from being reached. In addition, some problems still remain to be worked out. We mention two of them.

- The economics of database reverse engineering is a critical point for which there is no clear proposal so far. In particular, we do not know at the present time how to evaluate the cost of a future reverse engineering project.
- The scalability of most techniques that have been proposed so far has yet to be demonstrated. Searching a 30,000 LOC program for a specific programming cliché is quite realistic. The same cannot be asserted for a 3,000,000 LOC set of programs. In the same way, finding primary key and foreign key patterns in a physical schema based on name and data type similarity is reasonably fast in a 200-table schema, but may be desperately slow in a tenfold larger schema.

REFERENCES

Aiken, P. (1996). *Data reverse engineering: Slaying the legacy dragon.* McGraw-Hill.

Andersson, M. (1994). Extracting an entity relationship schema from a relational database through reverse engineering. *Proceedings of the 13th International Conference on ER Approach*.

Batini, C., Ceri, S., & Navathe, S. (1992). *Conceptual database design: An entity-relationship approach.* Benjamin/Cummings.

Baxter, I., & Mehlich, M. (2000). Reverse engineering is reverse forward engineering. *Science of Computer Programming, 36,* 131-147.

Blaha, M. R., & Premerlani, W. J. (1995). Observed idiosyncrasies of relational database designs. *Proceedings of the 2nd IEEE Working Conference on Reverse Engineering.*

Brodie, M., & Stonebraker, M. (1995). *Migrating legacy systems.* Morgan Kaufmann.

Casanova, M. A., & Amaral De Sa, J. E. (1984). Mapping uninterpreted schemes into entity-relationship diagrams: Two applications to conceptual schema design. *IBM J. Res. & Develop., 28*(1).

Chung, C., Gertz, M., & Sundaresan, N. (2002). Reverse engineering for Web data: From visual to semantic structures. *Proceedings of the 18th International Conference on Data Engineering (ICDE'02).*

Clève, A. (2006). Co-transformations in database applications evolution. In R. Lämmel, J. Saraiva, & J. Visser (Eds.), *Lecture notes in computer science: Vol. 4143. Generative and transformational techniques in software engineering* (pp. 399-411). Springer-Verlag.

Davis, K. H., & Aiken, P. H. (2000). Data reverse engineering: A historical view. *Proceedings of the 7th Working Conference on Reverse Engineering (WCRE'00).*

Davis, K. H., & Arora, A. K. (1985). A methodology for translating a conventional file system into an entity-relationship model. *Proceedings of ERA.*

Edwards, H. M., & Munro, M. (1995). Deriving a logical model for a system using recast method. *Proceedings of the 2nd IEEE WC on Reverse Engineering.*

Hainaut, J.-L. (2002). *Introduction to database reverse engineering.* Retrieved February 2007

from http://www.info.fundp.ac.be/~dbm/publication/2002/DBRE-2002.pdf

Hainaut, J.-L. (2006). The transformational approach to database engineering. In R. Lämmel, J. Saraiva, & J. Visser (Eds.), *Lecture notes in computer science: Vol. 4143. Generative and transformational techniques in software engineering* (pp. 89-138). Springer-Verlag.

Hainaut, J.-L., Chandelon, M., Tonneau, C., & Joris, M. (1993, May). Contribution to a theory of database reverse engineering. Proceedings of the IEEE Working Conference on Reverse Engineering, Baltimore.

Hainaut, J.-L., Roland, D., Hick J.-M., Henrard, J., & Englebert, V. (1996). Database reverse engineering: From requirements to CARE tools. *Journal of Automated Software Engineering, 3*(1).

Heeseok, L., & Cheonsoo, Y. (2000). A form driven object-oriented reverse engineering methodology. *Information Systems, 25*(3), 235-259.

Henrard, J. (2003). *Program understanding in database reverse engineering.* Unpublished doctoral dissertation, University of Namur. Retrieved February 2007 from http://www.info.fundp.ac.be/~dbm/publication/2003/jhe_thesis.pdf

Hick, J.-M. (2001). *Evolution of relational database applications.* Unpublished doctoral dissertation, University of Namur. Retrieved February 2007 from http://www.info.fundp.ac.be/~dbm/publication/2001/these-jmh.pdf

Petit, J.-M., Kouloumdjian, J., Bouliaut, J.-F., & Toumani, F. (1994). Using queries to improve database reverse engineering. *Proceedings of the 13th International Conference on ER Approach.*

KEY TERMS

Database Reverse Engineering: The process through which the logical and conceptual schemas of a legacy database, or of a set of files, are recovered, or rebuilt, from various information sources such as DDL code, data dictionary contents, database contents, or the source code of application programs that use the database.

Data Structure Conceptualization: The process within reverse engineering that aims at deriving a plausible conceptual schema from the logical schema of a database.

Data Structure Extraction: The process within reverse engineering that attempts to recover the implicit constructs of a database.

Implicit Construct: A data structure or an integrity constraint that holds, or should hold, among the data, but that has not been explicitly declared in the DDL code of the database. Implicit compound and multivalued fields as well as implicit foreign keys are some of the most challenging constructs to chase when recovering the logical schema of a database.

Legacy Information System: "[A] data-intensive application, such as [a] business system based on hundreds or thousands of data files (or tables), that significantly resists modifications and change" (Brodie & Stonebraker, 1995).

Optimization Schema Construct: Any data structure or constraint that appears in a logical or physical schema and that conveys no information (does not appear in the conceptual schema). Its aim is to increase the performance of the database and of its application programs. Unnormalized tables, redundancies, and artificially split or merged tables are some popular examples.

System Empirical Design: A way to build a system that does not rely on a strict methodology but rather on the experience and the intuition of the developer. An empirically designed database often includes undocumented idiosyncrasies that may be difficult to understand when attempting to reverse engineer it.

Chapter XXII
Imprecise Functional Dependencies

Vincenzo Deufemia
Università di Salerno, Italy

Giuseppe Polese
Università di Salerno, Italy

Mario Vacca
Università di Salerno, Italy

INTRODUCTION

Functional dependencies represent a fundamental concept in the design of a database since they are capable of capturing some semantics of the data strongly connected with the occurrence of redundancy in a database. The development of applications working on fuzzy and multimedia databases has required the extension of the functional dependency notion to these new types of data. Unfortunately, the concept of imprecise functional dependence or fuzzy functional dependence (*IFD*, for short) has not had a cogent and largely accepted definition yet. In fact, in attempt to capture different aspects of this notion of many proposals of IFD definition exist in literature, all having semantics and objectives somewhat unclear, especially with respect to the concern of redundancy (Bosc, *et al.*, 1994, Cubero & Vila, 1994, Raju & Majumdar, 1988, Tyagi, *et al.*, 2005, Wang, *et al.*, 2000). Moreover, the debate on the definition of the concept of fuzzy functional dependency seems to be still in progress, as shown by the following question: "But the question remains: are these extended notions of functional dependency a natural generalization?" (Tyagi *et al.*, 2005).

The first problem that arises in the definition of a "well-defined" IFD is to avoid subjectivity. To this aim, in what follows, we report some pertinent dissertations from the literature. In particular, there is a conspicuous literature facing the problem of the definition of scientific concepts, and two behaviors are possible when the need for a new

definition arises. Firstly, according to the received view, the fact that scientists do not agree, shows that the definition has not been given yet up to now. In fact, for Carnap (1950), the procedure of explication is "the transformation of an inexact, prescientific concept, the explicandum, into a new exact concept, the explicatum." (p. 3). Secondly, according to the analytic method view (Cellucci, 1998; Cellucci, 2000), a definition is always an ongoing process, as it is based on the formulation of a hypothesis. Within this philosophy, a definition of the concept of IFD has only to rely on the requirements of the specific problem that a person is facing.

A better understanding of the concept of IFD is possible only by looking at the ongoing process of its definition, through a critical and unifying view. Therefore, the goal of this proposal is to present an overview of imprecise functional dependencies and to provide a critical discussion of the dependencies presented for fuzzy and multimedia data.

BACKGROUND

In the sequel, we use the following notation: t, t_1, t_2, etc are tuples; $t[X]$ are the values of the tuple t corresponding to the set of attributes X (these values are also called a *X-representation*); $A_k[t]$ is the k-th attribute value of the tuple t; $A_k[t_i, t_j]$ stand for the couple $(A_k[t_i], A_k[t_j])$.

In traditional relational databases a *functional dependency* (*FD*, for short) is defined as a constraint between two sets of attributes from the database. Given two sets of attributes X and Y, a functional dependency between them is denoted by $X \rightarrow Y$. The constraint says that, for any two tuples t_1 and t_2 having $t_1[X] = t_2[X]$, then $t_1[Y] = t_2[Y]$. More precisely, given a table R,

$$X \rightarrow Y \Leftrightarrow \forall t_1, t_2 \in R.(t_1[X] = t_2[X] \Rightarrow t_1[Y] = t_2[Y]).$$

This concept cannot be immediately applied to multimedia databases or to fuzzy databases, because of the impossibility or inopportunity to apply the strict equality. For example a dependency between electrocardiography (ECG) and the hearth sound (HS) exists if we relax the equality in such a way that we can reasonably compare two electrocardiography images and two hearth sounds. In other words, two electrocardiography images are usually considered "equals" even though they do not coincide perfectly.

Imprecise functional dependencies (IFDs) are extensions of classical FDs and they serve just to capture the relations between data when equality is relaxed.

In general, an imprecise functional dependency between two sets of attributes exists if "similar X-representations are associated to similar Y-representations". Naturally, this expression of the IFD concept is too vague, as it specifies neither the concept of similar representations, nor the nature of the association between representations.

In the sequel we specify the basic constituents of an IFD: the *fuzzy implication*, the *similarity* between attributes, and the *resemblance* between tuples. Their use in the definition of IFDs will be clear in the next section.

Fuzzy Logical Implication

The concept of fuzzy implication is an extension of the classical one. In fact, while classical implication can be seen as an operation taking couples of binary values to a binary value, fuzzy implication is an operation which take two real values from the interval [0,1] as input and associates a value always in [0,1]:

$\Rightarrow$: $\{0,1\}\times\{0,1\} \rightarrow \{0,1\}$ (classical implication)

$\Rightarrow_f$: $[0,1] \times [0,1] \rightarrow [0,1]$ (fuzzy implication)

Two kinds of fuzzy implication1[1] are used in the definition of IFDs: the Rescher-Gaines' implication and the Gödel implication.

The Rescher-Gaines' implication, denoted by $\Rightarrow_{RG}$, is equal to one if the value on the left side of the implication is less or equal than that on the right one, otherwise its result is equal to zero. Formally:

$$a \Rightarrow_{RG} b = \begin{cases} 1 & \text{if } a \leq b \\ 0 & \text{otherwise} \end{cases}$$

The Gödel implication, denoted by $\Rightarrow_G$, is one if the left side value is less or equal to the right side one, otherwise it is equal to the value on the right side. Formally:

$$a \Rightarrow_{G} b = \begin{cases} 1 & \text{if } a \leq b \\ b & \text{otherwise} \end{cases}$$

It is worthwhile to notice that both of implications reduce to the classical one when a and b ranges over {0,1}.

Similarity Relations

The concept of similarity relation between two elements *x* and *y* belonging to a given domain D is based on the concept of fuzzy subset of a given set.

Given a set D, a fuzzy subset S of D is defined by a function

$$\mu_S: D \rightarrow [0,1]$$

giving the degree of membership of the elements of D to S.

Analogously, a relation S on D is a fuzzy subset of the Cartesian product D x D:

$$\mu_S: D \times D \rightarrow [0,1],$$

where $\mu_S(x,y)$ is interpreted as the degree of membership of (x, y) to the relation *S*.

A similarity relation which is reflexive and symmetric is called *proximity* (Dubois & Prade, 2000). Formally:

Definition 1. A *proximity* μ_S on D is a similarity on S, $\mu_S: D \times D \rightarrow [0,1]$, such that

i. $\mu_S(x,x)= 1$ for all $x \in S$
ii. $\mu_S(x,y)= \mu_S(y,x)$ for all $x,y \in S$

Sometimes, also the *transitivity* should be satisfied:

iii. $\mu_S(x,z) \geq \max_{y \in D}\{\min(\mu_S(x,y), \mu_S(y,z))\}$ for all $x,y,z \in S$

When the transitivity holds, μ_S is called a *similarity* (Dubois & Prade, 2000). Moreover, $1 - \mu_S(x,y)$ is called *dissimilarity* (Dubois & Prade, 2000).

The proximity relations (also called *similarity* or *EQUAL fuzzy relation* in the database literature) are widely used in the definition of IFDs. An example of similarity is the *min* function.

Example 1. Let us consider attributes *Salary* and *Job* of a university employees' database. The following tables define, respectively, the proximity μ_{sal} on the *Salary* values and the proximity μ_{sal} on the *Job* values.

Resemblance Relations

Informally speaking, a resemblance relation is a relaxation of the equality on a subset of attributes. In this paper, according to Bosc, *et al.* (1994), we will use the notation

$$RES_X(t_1[X], t_2[X])$$

Table 1. Proximity μ Job on attribute Job

μ_{job}	teacher	researcher	associate	professor
teacher	1	0.6	0.3	0.1
researcher	0.6	1	0.6	0.3
associate	0.3	0.6	1	0.6
professor	0.1	0.3	0.6	1

Table 2. Proximity μSal on attribute Salary

μ_{sal}	1200€	1500€	2000€	>2000€
1200€	1	0.8	06	0.2
1500€	0.8	1	0.75	0.2
2000€	0.6	0.75	1	0.4
>2000€	0.2	0.2	0.4	1

to indicate the resemblance between $t_1[X]$, $t_2[X]$. Analogously to the similarity, $RES_X(t_1[X], t_2[X])$ is supposed to return a value in [0,1].

In general, tuples of a relation can be compared by means of a set of relevant features Φ. For instance, images can be compared using attributes like color, texture, shape, etc.; audio data can be compared using loudness, pitch, brightness, bandwidth, and harmonicity. The values of each feature $F \in \Phi$ belong to a domain D = dom(F).

The approaches to build resemblances are based on concepts from similarity theory (Eidenberger, 2006; Fagin, 1996; Santini & Jain, 1999; Tversky, 1977; Tversky & Gati, 1982).

A typical approach used in a lot of IFD definitions combines the results produced by the similarity functions applied to the elements of the tuples. More precisely:

$$RES_X(t_1[X], t_2[X]) = g(\mu_1(A_1[t_1, t_2]),\ldots, \mu_n(A_n[t_1, t_2]))$$

where $\mu_1,\ldots,\mu_n$ are similarities and $g:[0,1]^n \rightarrow [0,1]$ is called *aggregation function* (Fagin, 1996). An example of aggregation function is the minimum function. See Fagin (1996) for further examples and properties of aggregation functions.

Important aggregation functions are the so called *triangular co-norm*.

Definition 2. A *triangular co-norm g* is a 2-ary aggregation function with the following properties:

1. $g(1,1)= 1$; $g(a,0)=g(0,a)=a$ ($\vee$-conservation)
2. $g(x,y)= g(y,x)$ (commutativity)
3. $g(x,y) \leq g(x',y')$ if $x \leq x'$ and $y \leq y'$ (monotonicity)
4. $g(g(x,y),z) = g(x,g(y,z))$ (associativity)

IMPRECISE FUNCTIONAL DEPENDENCIES

Using the notation introduced in the above section, the concept of imprecise dependency can be expressed in the following general way:

IFD: $X \rightarrow Y \Leftrightarrow \forall t_1, t_2 \in R.(RES_X(t_1[X], t_2[X] \Rightarrow_f RES_Y(t_1[Y], t_2[Y])$

However, many definitions of IFD have been proposed in the recent years. In this section we describe the main features of several IFDs providing examples and critiques.

The Raju and Majumdar IFD

The first approach we deal with is that of Raju and Majumdar (Raju & Majumdar, 1988). This approach is currently considered a milestone in the development of the theory of fuzzy functional dependencies. In this view an imprecise dependency IFD is defined as follows:

IFD : $X \rightarrow Y \Leftrightarrow (\forall t_1, t_2 \in R)(RES_X(t_1[X], t_2[X]) \Rightarrow_{RG} RES_Y(t_1[Y], t_2[Y]))$

The resemblance function over tuples is defined using the method of the aggregation function:

$$RES_X(t_1[X], t_2[X]) = \min_{i \in I} \{\mu^i_{EQ}(x_{1i}, x_{2i})\} \quad i = \{1,..., n\}$$

where μ^i_{EQ} (i=1,…,n) are similarity relations over attributes.

From the definition of Rescher-Gaines' implication, it follows that

$(\forall t_1, t_2 \in R)(RES_X(t_1[X], t_2[X]) \leq RES_Y(t_1[Y], t_2[Y]))$

This notion of IFD is based on the assumption that the more resemblant the X-representations is, the more resemblant the Y-representations should be.

Table 3.

Name	Job	Salary
Marius	Teacher	1500€
Vincent	Researcher	2000€
Joseph	Professor	3000€

Table 4.

Name	Job	Salary
Marius	teacher	1500€
Vincent	researcher	2000€
Joseph	professor	3000€
Mary	researcher	1500€

Example 2. Let us consider the proximities μ_{sal} and μ_{sal} defined in example 1. Since we consider only one attribute, the resemblance and its underlying proximity will coincide: $RES_{Job} = \mu_{Job}$ and $RES_{Sal} = \mu_{Sal}$.

The relation in Table 3 contains information on the employee' salaries of a University.

It is possible to verify that an IFD: *Job* → *Salary* exists.

However, if we consider the relation in Table 4:

IFD: *Job* → *Salary* does not hold, because

RES_{Job}(researcher, researcher) = 1 > 0.75 = RES_{Sal}(2000€, 1500€)

RES_{Job}(professor, researcher) = 0.3 > 0.2 = RES_{Sal}(3000€, 1500€)

Therefore, this notion of IFD lacks in representing functional dependencies like those in which the value of the first resemblance function is greater than that of the second one. Moreover this type of dependency is syntactic, since it is showed by the case $RES_X(t_1[X], t_2[X]) = 0 = RES_Y(t_1[Y], t_2[Y])$ in which there may exist tuples whose X-

representation could share some possible values, while the corresponding Y-representation do not share a single value.

The Chen et al. IFD

Another approach to the definition of IFD (Chen *et. al.*, 1996) use a generic implication operator $\Rightarrow_f$ and a threshold $q \in (0,1]$:

$$IFD : X \rightarrow_q Y \Leftrightarrow \min_{\{t1,t2 \in R\}}(RES_X(t_1[X], t_2[X]) \Rightarrow_f RES_Y(t_1[Y], t_2[Y])) \geq q$$

The resemblance function over tuples is defined using the method of the aggregation function.

Example3. Let us consider proximity relations μ_{sal} and μ_{sal} of example 1, table 4 of example 2, and a special case of Chen *et al.* IFD with Gödel operator for $\Rightarrow_f$.

We obtain that:

RES_{Job}(teacher, researcher) = 0.6 < 0.75 = RES_{Sal}(1500€, 2000€) entails that $0.6 \Rightarrow_G 0.75 = 1$

RES_{Job}(teacher, professor) = 0.1 < 0.2 = RES_{Sal}(1500€, 3000€) entails that $0.1 \Rightarrow_G 0.2 = 1$

RES_{Job}(teacher, researcher) = 0.6 < 1 = RES_{Sal}(1500€, 1500€) entails that $0.6 \Rightarrow_G 1 = 1$

RES_{Job}(researcher, professor) = 0.3 < 0.4 = RES_{Sal}(2000€, 3000€) entails that $0.3 \Rightarrow_G 0.4 = 1$

RES_{Job}(researcher, researcher) = 1 > 0.75 = RES_{Sal}(2000€, 1500€) entails that $1 \Rightarrow_G 0.75 = 0.75$

RES_{Job}(professor, researcher) = 0.3 > 0.2 = RES_{Sal}(3000€, 1500€) entails that $0.3 \Rightarrow_G 0.2 = 0.2$

Thus, an IFD: Job $\rightarrow_q$ Salary exists for every $q < 0.2$

This example shows that this approach overcomes the limitation $RES_X(t_1[X], t_2[X]) \leq RES_Y(t_1[Y], t_2[Y])$ of the Raju and Majumdar IFD.

Moreover, it is worth to note that $X \rightarrow_1 Y$ reduces to the Raju and Majumdar IFD (A formal proof can be found in (Cubero & Vila,1994)).

However, Chen *et al.* IFD definition does not allow to define functional dependencies in which the resemblance value of X-representations is greater than the corresponding for Y-representations.

The Cubero and Vila IFD

In the Cubero and Vila approach (1994) the resemblance is defined as the vector of the values obtained by the similarities on attributes:

$$RES_X(t_1[X], t_2[X]) = (\mu^i_{EQ}(A_i[t_1], A_i[t_2]))_{i \in I}$$

The Cubero and Vila notion of imprecise functional dependency is defined as follows:

$$IFD : X \rightarrow_{\alpha,\beta} Y \Leftrightarrow (\forall t_1, t_2 \in R)\ (RES_X(t_1[X], t_2[X]) \geq \alpha \Rightarrow RES_Y(t_1[Y], t_2[Y]) \geq \beta)$$

where $\alpha = (\alpha_1, \ldots, \alpha_n)$ and $\beta = (\beta_1, \ldots, \beta_n)$ are vectors of thresholds. Moreover, $RES_X(t_1[X], t_2[X]) \geq \alpha$ if and only if $\mu^i_{EQ}(A_i[t_1], A_i[t_2]) \geq \alpha_i$ for every $i=1,\ldots,n$.

Example 4. Let us consider the proximities μ_{sal} and μ_{sal} of example 1, and table 2 of example 2. It is easy to verify that an IFD: $X \rightarrow_{\alpha,\beta} Y$ exists for $\beta \leq 0.75$ and for every α.

The Cubero and Vila definition of IFD generalizes the previous ones and is capable to capture dependencies that the previous approaches are not able to cover.

Type-M Functional Dependency

In the approach of Chang *et al.* (2005) the resemblance relation between tuples is defined as

aggregation function on dissimilarities (a kind of co-norm). The relations of similarity on tuples is defined as follows:

$$x \cong_{RES(\sigma)} y \Leftrightarrow RES_X(t_1[X], t_2[X]) \leq \sigma$$

where $\sigma \in [0,1]$ is a threshold. In this case, $t_1[X]$ is said to be similar to $t_2[X]$ within σ with respect to RES_X.

The notion of imprecise functional dependency that they introduced is suitable for multimedia database and was used to introduce the concept of normalization in this kind of database. This concept of imprecise dependency, called *type-M functional dependency* (Multimedia Functional Dependency), is defined as follows:

MFD: $X_{g1}(t') \rightarrow Y_{g2}(t'')$ is valid on a relation R if and only if

$$(\forall t_1, t_2 \in R)(t_1[X] \cong_{g1(t')} t_2[X] \Rightarrow t_1[Y] \cong_{g2(t'')} t_2[Y])$$

where g_1 and g_2 are two aggregation (t-conorm) functions, and t' and $t'' \in [0,1]$ are thresholds.

It is worthwhile to notice that the use of thresholds allows to avoid fuzzy implications. This means that the features used by g_2 on Y depend on the features used by g_1 on X; or, alternatively, the values of the features used by g_1 on X component imply the range of values for the features used by g_2 on Y component.

Example 5. If we define a functional dependency on a medical database between attributes ECG (electrocardiography) and PULSE (heartbeat), and use fractal dimensionality for comparing ECGs and a similarity measure, named *HS*, for comparing heart sounds, we would write as follows

$$ECG_{(FRACTAL,t')} \rightarrow PULSE_{(HS,t'')}$$

This constraint says that for any two tuples t_1 and t_2 such that $t_1[ECG]$ is considered similar within the threshold t' to $t_2[ECG]$ by the *FRACTAL*, then $t_1[PULSE]$ is considered similar within the threshold t'' to $t_2[PULSE]$ by the *HS*.

FUTURE TRENDS

Many definitions suffer from a serious problem: the presence of thresholds. On the one hand, thresholds contribute both to give flexibility and to extend the capability of IFD to capture dependency. On the other hand, thresholds introduce subjectivity in the definition and this is not always desirable. Therefore, a future research trend is the definition of threshold "free" IFDs, which could be realized by exploiting concepts from fuzzy set theory, like for example the notion of *cut* (Dubois & Prade, 2000).

Moreover, we think that future research trends include applications of IFDs to multimedia data, this require the "migration" of the basic concepts from the theory of classical database to that of multimedia.

CONCLUSION

In this paper we have presented the concept of imprecise functional dependency, both outlining its basic constitutive elements and examining representative definitions of IFD existing in the literature. We have provided a critical discussion of the concept of imprecise functional dependency also showing its winding development and discussing the open problems.

REFERENCES

Bosc, P., Dubois, D., & Prade, H. (1994). Fuzzy functional dependencies - An overview and a

critical discussion. *Proc. of 3rd IEEE Internat. Conf. on Fuzzy Systems*, 325-330. Orlando.

Carnap, R. (1950). *Logical foundations of probability*, University of Chicago Press.

Chang, S. K., Deufemia, V., & Polese, G. (2005). Normalizing multimedia databases. *Encyclopedia of Database Technologies and Applications*, 408-412.

Chen, G., Kerre, E. E., & Vandenbulcke, J. (1996). Normalization based on fuzzy functional dependency in a fuzzy relational data model. *Information Systems, 21*(3), 299-310.

Cubero, J. C., & Vila, M. A. (1994). A new definition of fuzzy functional dependency in fuzzy relational databases. *International Journal of Intelligent Systems, 9*(5), 441-448

Cellucci, C. (1998). The scope of logic: Deduction, abduction, analogy. *Theoria*, 217-242.

Cellucci, C. (2000). The growth of mathematical knowledge: An open world view. *The growth of mathematical knowledge*, 153-176.Kluwer.

Dubois, D., & Prade, H. M. (2000). *Fundamentals of fuzzy sets*, Kluwer Academic.

Eidenberger, H. (2006). Evaluation and analysis of similarity measures for content-based visual information retrieval. *Multimedia Systems, 12*(2), 71-87.

Fagin, R. (1996). Combining fuzzy information from multiple systems. *Proc. Fifteenth ACM Symposium on Principles of Database Systems*, 216-226.

Höhle, U. (1995). Commutative, residuated l-monoids. *Non-Classical Logics and Their Applications to Fuzzy Subsets*, 53-106. Kluwer Academic Publishers.

Raju, K. V. S. V. N., & Majumdar, A. K. (1988). Fuzzy functional dependencies and lossless join decomposition of fuzzy relational database systems. *ACM Transactions on Database Systems, 13*(2), 129-166

Santini, S., & Jain, R. (1999). Similarity measures. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 21*(9), 871-883.

Tversky, A. (1977). Features of similarity. *Psychological Review, 84*, 327-352.

Tversky, A., & Gati, I. (1982). Similarity, separability, and the triangle inequality. *Psychological Review, 89*, 123-154.

Tyagi, B. K., Sharfuddin, A., Dutta, R. N., & Tayal, D.K. (2005). A complete axiomatization of fuzzy functional dependencies using fuzzy function. *Fuzzy Sets and Systems, 151*(2), 363-379.

Wang, S. L., Tsai, J. S., & Hong, T. P. (2000). Mining functional dependencies from fuzzy relational databases. *Proceedings of the 2000 ACM symposium on Applied computing, 1*, 490-493.

KEY TERMS

Functional Dependency: Given two sets of attributes X and Y, a functional dependency between them is denoted by $X \rightarrow Y$. The constraint says that, for any two tuples t_1 and t_2 having $t_1[X] = t_2[X]$, then $t_1[Y] = t_2[Y]$. More precisely, given a table R, $X \rightarrow Y \Leftrightarrow \forall t_1, t_2 \in R.(t_1[X] = t_2[X] \Rightarrow t_1[Y] = t_2[Y])$.

Fuzzy Implication: It is an extension of the classical implication in which the two values involved are not necessarily true or false (1 or 0), but can be also two *degrees of truth* (belonging to [0,1]). The result is another degree of truth. More precisely, it is a function $\Rightarrow_f$: [0,1] $\rightarrow$ [0,1].

Imprecise Functional Dependency: Given two sets of attributes X and Y, an imprecise functional dependency between them is denoted by $X \rightarrow Y$. The constraint says that, for any two tuples t_1 and t_2 having $t_1[X]$ similar to $t_2[X]$, then $t_1[Y]$ is similar to $t_2[Y]$. More precisely, given a table

R, IFD: $X \rightarrow Y \Leftrightarrow \forall t_1, t_2 \in R.(RES_X(t_1[X], t_2[X] \Rightarrow_f RES_Y(t_1[Y], t_2[Y])$.

Multimedia Functional Dependency (Type-M Functional Dependency): Let R be a relation with attribute set U, and X, $Y \subseteq U$. $X_{g1}(t') \rightarrow Y_{g2}(t'')$ is a *type-M functional dependency* (MFD) relation if and only if for any two tuples t1 and t2 in R that have $t1[X] \cong g_1(t')$ t2[X], then $t1[Y] \cong g_2(t'')$ t2[Y], where $g_1 \in TD(X)$ and $g_2 \in TD(Y)$, whereas t' and t'' $\in$ [0,1] are thresholds.

Resemblance Relation: It is a relaxing of the equality on subset of attributes. Given a set of attributes X, and let *t*[*X*] be the values of the tuple *t* corresponding to the set of attributes *X*, the resemblance on X is denoted by $RES_X(t_1[X], t_2[X])$.

Similarity Relation: A similarity on a set D is a fuzzy subset of the Cartesian product D × D: μ_S: D × D → [0,1] with the properties of reflexivity ($\mu_S(x,x)$= 1 for all x∈S), symmetry ($\mu_S(x,y)= \mu_S(y,x)$ for all x,y∈S) and *max-min transitivity* ($\mu_S(x,z) \geq \max_{y \in D}\{\min(\mu_S(x,y), \mu_S(y,z))\}$ for all x,y,z∈S).

Triangular Co-Norm: A *triangular co-norm g* is a 2-ary aggregation function with the following properties:

1. g(1,1)= 1; g(a,0)=g(0,a)=a (∨-conservation)
2. g(x,y)= g(y,x) (commutativity)
3. $g(x,y) \leq g(x',y')$ if $x \leq x'$ and $y \leq y'$ (monotonicity)
4. g(g(x,y),z) = g(x,g(y,z)) (associativity)

ENDNOTE

[1] See Höhle (1995) for a detailed treatment of the topic.

Chapter XXIII
Horizontal Data Partitioning: Past, Present and Future

Ladjel Bellatreche
LISI/ENSMA - University of Poitiers, France

INTRODUCTION

Horizontal data partitioning is the process of splitting access objects into set of disjoint rows. It was first introduced in the end of 70's and beginning of the 80's (Ceri et al., 1982) for logically designing databases in order to improve the query performance by eliminating unnecessary accesses to non-relevant data. It knew a large success (in the beginning of the 80's) in designing homogeneous distributed databases (Ceri et al., 1982; Ceri et al., 1984; Özsu et al., 1999) and parallel databases (DeWitt et al., 1992; Valduriez, 1993). In distributed environment, horizontal partitioning decomposes global tables into horizontal fragments, where each partition may be spread over multiple nodes. End users at the node can perform local queries/transactions on the partition *transparently* (the fragmentation of data across multiple sites/processors is not visible to the users.). This increases performance for sites that have regular transactions involving certain views of data, whilst maintaining availability and security. In parallel database context (Rao et al., 2002), horizontal partitioning has been used in order to speed up query performance in a shared-nothing parallel database system (DeWitt et al., 1992). This will be done by both intra-query and intra-query parallelisms (Valduriez, 1993). It also facilitates the exploitation of the inputs/outputs bandwidth of the disks by reading and writing data in parallel. In this paper, we use fragmentation and partitioning words interchangeably.

There are two versions of horizontal partitioning (Özsu et al., 1999): *primary* and *derived*. Primary horizontal partitioning of a relation is performed using selection predicates that are defined on that relation. Note that a simple predicate (selection predicate) has the following form: A θ value, where A is an attribute of the table to be fragmented, θ is one of six comparison operators $\{=, <, >, \leq, \geq\}$ and value is the predicate constant

belonging to the domain of the attribute A. *Derived horizontal partitioning*, on the other hand, is the fragmentation of a relation that results from predicates being defined on another relation. In other word, the derived horizontal partitioning of a table is based on the fragmentation schema of another table (the fragmentation schema is the result of the partitioning process of a given table). The derived partitioning of a table R according a fragmentation schema of S is feasible if and only if there is a join link between R and S. The relation at the link is called the owner of the link (the case of the table S) and the relation at the head is called member (Ceri et al., 1982) (the case of the relation R).

In the context of data warehousing, the horizontal partitioning becomes an important aspect of physical design. Note that in relational data warehouses, two types of tables exist: dimension tables and fact table. A dimension table is a table containing the data for one dimension within a warehouse schema. The primary key is used to link to the fact table, and each level in the dimension has a corresponding field in the dimension table. The fact table is the central table in a star schema, containing the basic facts or measures of interest. Dimension fields are also included (as foreign keys) to link to each dimension table. In this context, horizontal partitioning allows to partition tables (fact and dimension tables) (Bellatreche et al., 2006) and materialized views and indexes (Sanjay et al., 2004). It has also been used designing parallel data warehouses. Due to its importance in the database fields, most of today's commercial database systems offer native DDL (data definition language) support for defining horizontal partitions of a table (Sanjay et al., 2004), where many partitioning methods are available: range, hash, list and composite (hash list, range hash). These methods map a given row in an object to a key partition. All rows of the object with the same partition number are stored in the same partition. These partitions may be assigned either in a single-node partitioning (single machine) or in multi-node partitioning (parallel machine). A range partitioning method is defined by a tuple (c, V), where c is a column type and V is an ordered sequence of values from the domain of c. This partitioning mode is often used when there is a logical range (examples: starting date, transaction date, close date, etc.).

Example 1

```
CREATE TABLE PARTITION_BY_RANGE
(. . . , BIRTH_MM INT NOT NULL, BIRTH_DD
INT NOT NULL, BIRTH_YYYY INT NOT
NULL)
PARTITION BY RANGE (BIRTH_YYYY,
BIRTH_MM, BIRTH_DD)
(PARTITION P_01 VALUES LESS THAN (1970,
01, 01) TABLESPACE TS01, . . .
PARTITION P_N VALUES LESS THAN
(MAXVALUE, MAXVALUE, MAXVALUE)
TABLESPACE TS05)
ENABLE ROW MOVEMENT;
```

The hash mode decomposes the data according to a hash function (provided by the system) applied to the values of the partitioning columns.

Example 2

```
CREATE TABLE PARTITION_BY_HASH
(FIRST_NAME VARCHAR2(10), MIDDLE_
INIT VARCHAR2(1), LAST_NAME VAR-
CHAR2(10), AGE INT NOT NULL)
PARTITION BY HASH (AGE)
(PARTITION P1TABLESPACE TS01,
 PARTITION P2 TABLESPACE TS02,
 PARTITION P3 TABLESPACE TS03,
 PARTITION P4 TABLESPACE TS04)
ENABLE ROW MOVEMENT;
```

The list partitioning splits a table according list values of a column. These methods can be combined to generate composite partitioning (range-hash, range-list). These methods are available in Oracle.

Another partitioning method, called, round robin used largely in parallel databases is used. It is defined as follows: if there are n processors, the jth tuple of a relation is assigned to processor j mod n. It is suitable for efficiently evaluating queries that access the entire relation. Recently, Oracle (Eadon & al., 2008) proposes an implementation of derived horizontal partitioning.

BACKGROUND

Many formalizations of horizontal partitioning problem have been studied in different contexts: centralized traditional databases (relational and object) (in the logical design), distributed homogeneous databases, parallel databases and relational data warehouses. In the distributed and parallel databases, horizontal partitioning is usually followed by the fragment allocation process that consists in allocating fragments to various sites or nodes. A general formulation of the horizontal partitioning problem in the context of centralized databases is the following: Given a set of relations $R = \{R_1, R_2, \ldots, R_n\}$ and a set of most frequently asked queries $Q = \{Q_1, Q_2, \ldots, Q_m\}$, the problem of horizontal partitioning consists in selecting a set of relations $R' \subseteq R$ to be partitioned and generating a set of fragments $F = \{F_1, F_2, \ldots, F_N\}$ per table such that: every fragment F_i $(1 \leq i \leq N)$ stores a subset of the tuples of a relation R_j of R' such as the sum of the query costs when executed on top of the partitioned schema is minimized. The number of generated fragments has a great impact on query processing (since a global query will be rewritten on the generated fragments) and data allocation (since one of the inputs of data allocation is the fragments).

Several horizontal partitioning algorithms have been proposed in the traditional databases, that we can classify into three categories: *predicate-based algorithm* (Ceri et al., 1982; Özsu et al., 1999), *affinity-based algorithm* (Karlapalem et al., 1996) and *cost-based algorithms* (Bellatreche et al., 2000). The predicate-based algorithm decides how to divide a relation horizontally into fragments using a set of predicates used by the most frequently asked queries. Given a relation R and a set of simple predicates $P = \{ p_1, p_2, \ldots, p_n \}$, the main steps of this algorithm are: (1) generation of minterm predicates $M = \{ m \mid m = \wedge (1 \leq k \leq n)\ p_k^* \}$, where p_k^* is either p_k or $\neg p_k$, (2) simplification minterms in M and eliminate useless ones and (3) For each m in M, generate a fragment σ_m R. This algorithm is very easy to use, but it has a high complexity. For n simple predicates, this algorithm generates 2^n minterms. It can be used for a reasonable number of simple predicates.

In order to reduce this complexity, another algorithm has been proposed which adapts the vertical partitioning algorithm proposed by (Navathe et al., 1984). It generates groups of predicates with a high affinity that represents the sum of access frequencies of queries. Each group gives a horizontal fragment by conjunction of its predicates. This algorithm has a low complexity (Karlapalem et al., 1996), but it takes into account *only* access frequencies to generate horizontal fragments. The last category of algorithm generates horizontal fragments based on a cost model calculating the goodness of each fragmentation schema (Bellatreche et al., 2000).

Any horizontal partitioning shall respect correctness rules: completeness, reconstruction and disjointness (Özsu et al., 1999). The union of the horizontal fragments must be equal to the original relation. Fragments are usually also required to be disjoint.

Once the member relations are fragmented, the owner relations can be easily partitioned using the semi join operation. For instance, suppose a member table R is decomposed into N horizontal fragments $\{R_1, R_2, \ldots, R_N\}$ such as: $R_i = \sigma_{cl_i}(R)$, where cl_i and σ represent a conjunction of simple predicates and the selection predicate, respectively. An owner relation S can be derived partition into N fragments $\{S_1, S_2, \ldots, S_N\}$, such as

S_i = S semi joint R_i ($1 \leq i \leq N$). Consequently, the derived fragmentation is straightforward.

A join between two horizontally partitioned relations R and S is called, distributed join. This join can be represented by a graph (called join graph), whose nodes represent horizontal fragments of each relation and an edge between two nodes exists if an implicit join between them. When the derived partitioning is used, the distributed graph becomes a simple one. Each node of R is joined with only one node of S. This means that the derived partitioning may reduce the cost of processing join operation.

MAIN THRUST: HORIZONTAL PARTITIONING IN DATA WAREHOUSES

In the data warehousing environment, the horizontal partitioning is very important in reducing query processing cost by eliminating no relevant access to data. Note that a relational data warehouse is modelled using a star schema or snow flake schema, where two types of tables are available: dimension and fact tables. On the top of this schema, complex queries, called, star join queries are executed. A **star query** is a join between a fact table and a number of dimension tables. Each dimension table is joined to the fact table using a primary key to foreign key join, but the dimension tables are not joined to each other. Join operations are preceded by selection operations on dimension tables.

The main advantages of the use of horizontal partitioning in data warehouses are:

1. Enhancing manageability, performance, parallelism and scalability: The manageability is ensured for example by (a) backing up data on a partition-by-partition basis, (b) loading a partition at a time, (c) storing data in different tablespaces on a partition-by-partition basis, (d) defining indexes on a partition-by-partition basis, (e) merging and splitting partitions to consolidate or further break apart partitions (using the following SQL statements: ALTER TABLE Table_Name MERGE PARTITIONS P1, P2 INTO PARTITION P3; ALTER TABLE Table_Name SPLIT P INTO (PARTITION P1, PARTITION P2), etc. Performance and Scalability are satisfied for instance by searching multiple table or index partitions in parallel and eliminating partitions from consideration based on a partition key. If a query includes a partition key as a predicate in the WHERE clause, a DBMS like Oracle will automatically route the query to the partition or partitions that are associated with the query and eliminate those partitions that will not have data included in the result set.
2. Since the OLAP queries use joins of multiple dimension tables and a fact table, the derived horizontal partitioning can be used to efficiently process joins between the fact table and various dimension tables (Bellatreche et al., 2006).
3. Designing a warehouse database for an integrated enterprise is a very complicated and iterative process since it involves collection of data from many departments/units and data cleaning, and requires extensive business modeling. Therefore, some organizations have preferred to develop a *data mart* to meet requirements specific to a departmental or restricted community of users. Of course, the development of data marts entails the lower cost and shorter implementation time. The role of the data mart is to present convenient *subsets of a data warehouse* to consumers having specific functional needs. There can be two approaches for developing the data mart, either it can be designed integrating data from source data (called bottom-up approach) or it can be designed deriving the data from warehouse (called,

top-down approach) (Bellatreche et al., 2000). By advocating the top down design for data mart and without partitioning, we need to assign to each data mart the whole data warehouse; even this data mart accesses only a subset of this data warehouse. By analyzing the needs of each data mart, we can partition the centralized data warehouse into several fragments that can be used as allocation units for data marts.

The main disadvantage of the horizontal partitioning in databases is that when update operations occur some tuples may migrate from one fragment to another (see example 1). The migration operation is sometimes costly. In the data warehouse environments, updates are not common compare to the append operations. Even if there are some update operations, the data warehouses are only periodically updated in a batch fashion during the time where the data warehouses are unavailable for querying. This situation makes the utilization of horizontal partitioning feasible, as the reorganization of its fragments can be done off-line.

In this context the horizontal partitioning is more challenging compare to that in relational and object databases. This challenge is due to the several choices of partitioning schema of a star or snowflake schemas:

1. Partition only the dimension tables using the primary partitioning. This scenario is not suitable for OLAP queries, because the sizes of dimension tables are generally small compare to the fact table. Most of OLAP queries access the fact table, which is very huge. Therefore, any partitioning that does not take into account the fact table is discarded.
2. Partition only the fact table using the primary partitioning. In a data warehouse modeled by a star schema, most of OLAP queries access dimension tables first and then the fact table.
3. Partition both fact and dimension tables, but independently. This means that we use the primary partitioning to partition the fact and dimension tables without taking into account the relationship between these tables. Based on the previous choices, this partitioning is not benefit for OLAP queries.
4. Partition *some/all* dimension tables using the primary partitioning, and use them to derived partitioned fact table. This approach is best in applying partitioning in data warehouses. Because it takes into consideration the queries requirements and the relationship between the fact table and dimension tables. We recommend the use of this scenario.

Example 3

To illustrate the procedure of fragmenting the fact table based on the fragmentation schemas of dimension tables according the last scenario, let's consider a star schema with three dimension tables: Customer, Time and Product and one fact table Sales. Suppose that the dimension table Customer is fragmented into two fragments CustFemale and CustMale defined by the following clauses:

CustFemale = $\sigma_{(Sex = `F')}$ (Customer) and CustMale = $\sigma_{(Sex = `M')}$ (Customer).

Therefore, the fact table Sales is fragmented based on the partitioning of Customer into two fragments Sales1 and Sales2 such as: $Sales_1$ = Sales *join* CustFemale and $Sales_2$ = Sales *join* CustMale. The initial star schema (Sales, Customer, Product, Time) is represented as the juxtaposition of two sub star schemas S1 and S2 such as: S1: (Sales1, CustFemale, Product, Time) (sales activities for only female customers) and S2: (Sales2, CustMale, Product, Time) (sales activities for only male customers). Note that the attributes used for partitioning are called fragmentation attributes (Sex attribute).

Complexity of our Horizontal Partitioning Methodology

The number of horizontal fragments of the fact table generated by the partitioning procedure is given by the following equation:

$$N = \prod_{i=1}^{g} m_i$$

where mi represents the number of fragments of the dimension table D_i.

Example 4

Suppose a data warehouse modelled with a star schema with three dimension tables Customer, Product and Time and one fact table Sales. The data warehouse administrator (DWA) partitions the three dimension tables as follows: Customer dimension into 50 fragments using the State attribute (the case of 50 states in the U.S.A.), Time into 48 fragments using the Month attribute (the sale analysis is done based on the four last years), and Product into 80 fragments using Type_of_ Package attribute. Therefore the fact table Sales is fragmented into 192 000 fragments (50 * 48 * 80). Consequently, instead to manage one star schema, the DWA will manage 192 000 sub star schemas. It will be very hard for him to maintain all these sub-star schemas.

A formulation of the horizontal partitioning problem is the following:

Given a set of dimension tables D = {D_1, D_2, ..., D_d} and a set of OLAP queries Q = {Q_1, Q_2, ..., Q_m}, where each query Q_i ($1 \leq i \leq m$) has an access frequency. The horizontal partitioning problem consists in determining a set of dimension tables $D' \subseteq D$ to be partitioned and generating a set of fragments that can be used to partition the fact table F into a set of horizontal fragments (called fact fragments) {F_1, F_2, ..., F_N} such that: (1) The sum of the query cost when executed on top of the partitioned star schema is minimized; and (2) $N \leq W$, where W is a threshold fixed by the DWA representing the maximal number of fragments that he can maintain. The respect of this constraint avoids an explosion of the number of the fact fragments.

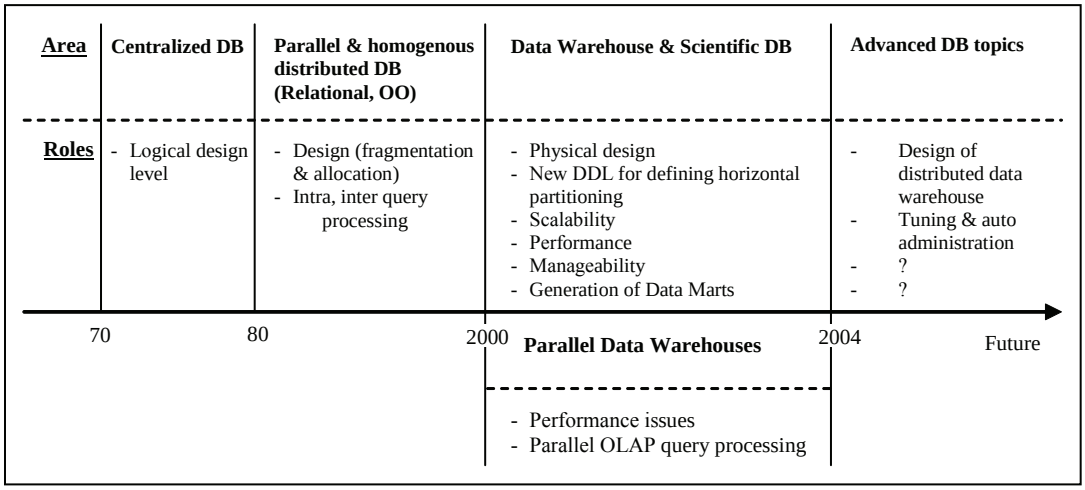
Many algorithms have proposed to deal with problem: genetic algorithm simulated annealing algorithm (Bellatreche et al., 2006; Bellatreche et al., 2005). These algorithms can fit into cost-based horizontal partitioning category.

Figure 1 summarizes the use of horizontal partitioning along the evolution of database technology.

FUTURE TRENDS

Many issues on horizontal partitioning are open. They concern implementations, physical design and tuning. (1) Regarding the implementations, we note that primary horizontal partitioning is largely supported by today's commercial systems, where native DDL support for defining horizontal partitions of a table (CREATE TABLE … PARTITION…). This DDL allows partitioning a table using only one (list, range and hash) or two attributes (composite range hash, list hash). There is a need to consider more than two attributes since to ensure a high performance, many attributes may be used to partition a database. (2) In physical design, there is a need to developed a powerful horizontal partitioning algorithms in the context of data warehouses that take into account the characteristics of the relational data warehouse (each join operation is between fact table and dimension tables and none join between dimension tables, etc.). Another point may concern the use of data partitioning in distributed environments, by considering data allocation and data localization problems. (3) For the tuning; it will interested to develop techniques combining the data partitioning and materialized views and advanced indexes (bitmap join indexes, and join indexes) in order to reduce the query processing

Figure 1. The use of horizontal partitioning along the database evolution



cost, storage cost and maintenance overhead and consider the data partitioning as a tool for tuning a database.

CONCLUSION

In this paper, we have presented an optimization structure, called, horizontal partitioning, used in traditional databases (relational and object), distributed and parallel databases, and data warehouses. We gave a history of the use of horizontal partitioning a long the evolution of the database. A special focus has been given to the data warehouses, where data partitioning represent an import aspect of physical design. Several horizontal partitioning scenarios of fragmenting a relational data warehouse modeled using a star schema are presented. The complexity of the number of generated fragments is given. A formulation of horizontal partitioning selection problem as an optimization problem with constraint is given. This constraint represents the number of fact fragments that the DWA should maintain.

REFERENCES

Bellatreche, L., & Boukhalfa, K. (2005). An Evolutionary Approach to Schema Partitioning Selection in a Data Warehouse. *7th International Conference on Data Warehousing and Knowledge Discovery*, (pp. 115-125).

Bellatreche, L., Boukhalfa, K., & Abdalla, H. I. (2006). SAGA: A Combination of Genetic and Simulated Annealing Algorithms for Physical Data Warehouse Design. In 23rd British National Conference on Databases (BNCOD'06), (pp. 212-219).

Bellatreche, L., Karlapalem, K., Mukesh, K. M., & Mohania, S. M. (2000). What Can Partitioning Do for Your Data Warehouses and Data Marts? *Proceedings on International Database Engineering and Applications Symposium* (IDEAS'2000), (pp. 437-446).

Bellatreche, L., Karlapalem, K., & Simonet, A. (2000). Algorithms and Support for Horizontal Class Partitioning in Object-Oriented Databases. *Distributed and Parallel Databases, 8*(2), 155-179.

DeWitt, D. J., & Gray, J. (1992). Parallel Database Systems: The Future of High Performance Database Systems. *Communications of the ACM, 35*(6), 85-98.

Ceri, S., Negri, M., & Pelagatti, G. (1982). Horizontal data partitioning in database design. *Proceedings of the ACM SIGMOD International Conference on Management of Data,* (pp. 128-136).

Ceri, S., & Pelagatti, G. (1984). *Distributed databases: principles & systems*. McGraw-Hill International Editions – Computer Science Series.

Karlapalem, K., Navathe, S. B., & Ammar, M. H. (1996). Optimal Redesign Policies to Support Dynamic Processing of Applications on a Distributed Relational Database System. *Information Systems, 21*(4), 353-367.

Eadon, G., Chong, E. I., Shankar, S., Raghavan, A., Srinivasan, J., & Das, S. (2008). Supporting Table Partitioning by Reference in Oracle. *To appear in ACM SIGMOD International Conference on Management of Data (SIGMOD'08).*

Özsu, M. T., & Valduriez, P. (1999). *Principles of Distributed Database Systems*: Second Edition. Prentice Hall.

Papadomanolakis, S., & Ailamaki, A. (2004). Autopart: Automating schema design for large scientific databases using data partitioning. *Proceedings of the 16th International Conference on Scientific and Statistical Database Management,* (pp. 383–392).

Garcia-Molina H., Labio W., Wiener L. J., & Zhuge, Y. (1998). Distributed and Parallel Computing Issues in Data Warehousing. *Proceedings of the Seventeenth Annual ACM Symposium on Principles of Distributed Computing,* (p. 7).

Rao, J., Zhang, C., Megiddo, N., & Lohman, G. (2002). Automating Physical Database Design in a Parallel Database. *Proceedings of the ACM SIGMOD International Conference on Management of Data,* (pp. 558-569).

Sanjay, A., Narasayya, V. R., & Yang, B. (2004). Integrating vertical and horizontal partitioning into automated physical database design. *Proceedings of the ACM International Conference on Management of Data (SIGMOD),* (pp. 359–370).

Valduriez P. (1993). Parallel Database Systems: Open Problems and New Issues. *Distributed and Parallel Databases, 1*(2), 137–165.

KEY TERMS

Data Warehouse: An integrated decision support database whose content is derived from the various operational databases. (1) A subject-oriented, integrated, time-variant, non-updatable collection of data used in support of management decision-making processes.

Dimension: A business perspective useful for analyzing data. A dimension usually contains one or more hierarchies that can be used to drill up or down to different levels of detail

Dimension Table: A table containing the data for one dimension within a star schema. The primary key is used to link to the fact table, and each level in the dimension has a corresponding field in the dimension table.

Distributed Database: Data in distributed database system is stored across several sites, and each is site is typically managed by a DBMS that can run independently of the other sites (autonomy of sites).

Fact Table: The central table in a star schema, containing the basic facts or measures of interest. Dimension fields are also included (as foreign keys) to link to each dimension table.

Horizontal Fragmentation: Each fragment consists of a subset of rows of the original relation.

Inter-Query Parallelism: Tthe parallel execution of multiple queries generated by concurrent transactions.

Intra-Query Parallelism: The parallel execution of multiple, independent operations possible within the same query.

Parallel Database: A database management system that is implemented on a tightly coupled multiprocessor.

Partitioning (or Fragmentation): Consists of breaking a relation into smaller relations or fragments.

Chapter XXIV
Database Support for Workflow Management Systems

Francisco A. C. Pinheiro
Universidade de Brasília, Brasil

INTRODUCTION: WORKFLOW SYSTEMS

A workflow is a series of work processes performed under rules that reflect the formal structure of the organization in which they are carried out and the relationships between their various parts. Workflow applications are software applications used to automate part of workflow processes. They run under the control of a workflow management system (WfMS). The WfMS usually comprises an organizational model, describing the process structure, and a process model, describing the process logic. The Workflow Management Coalition (WfMC, 2008) publishes a set of workflow definitions and related material, including a reference model.

Databases are commonly used as a WfMS supporting technology. Not only workflow data are maintained in databases but also the rules governing processes can be stored in database schemas. Database functionality can be used both for defining and managing process models as well as for environment notification and process enactment. This article shows how particular database-related technologies can be used to support WfMS.

Table 1 relates workflow issues and the database technologies that can be used to deal with them. It summarizes the content of this article presenting the relationships discussed in the text. The next two sections discussing workflow management issues and database support are related to the columns and lines of the table and provide an explanation for these relationships. Only the general relationships stressed in the text are shown. For example, data replication and partitioning have an influence on scalability and privacy but the table does not show the same influence with respect to distributed database technology, although data replication and partitioning are basic strategies of distributed databases.

BACKGROUND: WORKFLOW MANAGEMENT ISSUES

Workflow applications are usually complex, distributed applications consisting of activities performed by people playing different roles, sometimes involving multiple departments and organizations. It is not unusual for different workflows to be governed by different WfMS running under a variety of platforms, application servers, and communications middleware. This situation requires ways of dealing with diversity and complexity and imposes a need for interoperability and scalability. Other issues like heterogeneity,

Table 1. General relationships between workflow issues and database technologies

		ENVIRONMENT			PEOPLE			PROCESS			DATA		
		Diversity	Interoperability	Scalability	Collaborative work	Flexibility	Evolution	Assignment of tasks	Consistency	Changes	Privacy	Location	Semantic heterogeneity
DATABASE TYPES	Distributed	✓	✓										✓
	Parallel			✓									
	Multi	✓	✓										
	Active							✓				✓	
DATABASE TECHNOLOGIES	Data replication			✓							✓		
	Data partitioning			✓							✓		
	Synchronization techniques								✓	✓			
	Transaction models				✓	✓			✓				
	Schema					✓	✓		✓	✓			✓
	Metadata				✓	✓	✓		✓			✓	✓

consistency, privacy, and flexibility naturally arise when we think about the configuration of such an environment and the development of workflow applications. Stohr and Zhao (2001) present a detailed discussion of workflow issues.

Environment Issues

Workflow diversity ranges from organization to infrastructure. Organizational diversity involves different processes, rules, and ways of organizing work. Infrastructure diversity involves different platforms, communication protocols, programming languages and data formats. Interoperability is needed for applications running in such a diverse environment. The changing environment also makes scalability crucial to cope with addition of new organizational units and redistribution of work to other (sometimes geographically distant) players. Support to these issues may be found in technologies incorporated into distributed, parallel and multidatabases.

People Issues

Collaborative work should not be hindered by overly rigid procedures. There should be room to modify processes as necessary, making possible to follow different paths by changing the ordering of activities or suppressing some of them. Flexibility and evolution are necessary to cope with different views about the organization of work and use of resources. The need for flexible and evolving environments is supported by metadata descriptions and schema evolution. Advanced transaction models should be in place to reconcile flexibility with consistency.

Process Issues

A number of workflow patterns for relevant process issues is described by van der Aalst et al. (2003). Particularly, the assignment of tasks is a highly dynamic activity that should be performed promptly and correctly. Dynamic team formation (Georgakopoulos, 2004), with members being added and released as a process goes on, requires the system to be aware of its environment. Active databases provide some sort of environment awareness and synchronization techniques may be used to keep processes and their data in a consistent state.

Data Issues

Data change in different ways, including location, as a result of people performing their activities. To locate the relevant data in a changing environment, with autonomous collaborating agents, is a difficult task. A complicating factor in distributed environments is that of semantic heterogeneity, in which people assign different meanings to the same or closely related data. We also have the issue of transparency in which some data should be shared while others should be kept private.

The use of metadata description and schema evolution help to maintain data availability and consistency, and to deal with semantic heterogeneity. Techniques of data replication and partitioning have an impact on privacy, and active database technology may be employed to notify and deliver the needed data to agents.

MAIN THRUST: DATABASE SUPPORT

This section relates particular database technologies to the workflow issues discussed above.

Distributed, Active, and Multi-Databases

Distributed databases are used to manage data stored in several places, while maintaining a uniform control over the operations that use and

change them. Parallel databases allow the parallel execution of actions and multi-databases, also referred to as federated or heterogeneous databases, make possible the coordinated use of several databases. These types of databases are already being used by workflow management systems that apply their internal mechanisms to deal with diversity, scalability, interoperability and heterogeneity.

Active database management systems make possible for the system to be aware of what is happening around it and to react properly and spontaneously, for example, firing triggers and alerters (Abiteboul et al., 2005). They are useful for the assignment of tasks and location of data and provide an appropriate support to build adaptation mechanisms into workflow systems.

Data Replication and Partitioning

Data may be partitioned or replicated in different locations to improve performance and scalability and to assure privacy. Data replication and partitioning are basic strategies used in distributed and parallel databases. An optimal strategy to set the right combination of replication and partitioning depends on the application, but it is possible to determine general data access patterns in case of WfMS (Schuster & Heinl, 1997).

Metadata Description and Database Schemas

Metadata descriptions are employed by database systems to make them aware of other database needs and also to locate relevant data in distributed systems, providing enough background information to judge the suitability and trustworthiness of the data sources (Shankaranarayanan & Even, 2006). It is an important technology concerning collaboration, flexibility and evolution in workflow systems.

Database schemas reflect the rules governing processes and their data. Schema evolution (Lerner, 2000) gives the user the ability to dynamically change the schema of a data source. This is important to preserve flexibility and guarantee consistency in changing environments.

Transaction Models

New transaction models, allowing flexible ways of dealing with consistency and recovery is paramount to support collaborative work. Nested transactions and open nested transactions are well-known advanced transaction models, allowing composite units of work to be reflected as transaction structures. A survey of advanced transactions models can be found in (Barghouti & Kaiser, 1991). Some workflow applications may require the definition of application-specific transaction models (Georgakopoulos, Hornick & Manola, 1996). Ray and Xin (2006) discuss the different kinds of dependencies that may arise in the context of advanced transaction models.

Database Synchronization Techniques

Concurrency requires synchronization to guarantee the integrity of concurrent data and processes. A number of synchronization techniques, such as fetch-and-add, set-and-test, and wait-free synchronization are already being used by database systems and may be applied by WfMS.

FUTURE TRENDS

There should be advances in all areas related to the flexible, cooperative and diversified ways of doing business. This points to an increasing use of multiple WfMS engines to support distributed workflow process execution. New architectures such as those related to grid computing impose further restrictions on concurrency and resource sharing. This has an impact on both database and workflow systems (Taniar and Goel, 2007).

We should expect more research in the areas of flexible coordination, awareness provision, team scalability, and integration of workflow systems with computer support for cooperative work (CSCW) and content management technologies (Georgakopoulos, 2004).

The overall trend is to increase flexibility, to incorporate controlled inconsistency, and to manage huge amounts of distributed data (Abiteboul et al., 2005). Distributed object databases (Kirchberg et al., 2007), customized transaction models (Georgakopoulos, 2004), advances on metadata description (Shankaranarayanan & Even, 2006) and semantic annotation (Uren et al., 2006) should all have a positive impact on collaborative work.

CONCLUSION

A WfMS implemented on top of a database system is a special database application. Therefore, it helps to understand how advances on databases as a supporting technology may be applied to build more useful workflow applications and, on the other hand, how workflow application needs may drive the improvement of database technologies.

Nevertheless, workflow applications are highly human-centered and should not be limited by technology. Organizational requirements have to be taken into consideration when developing workflow applications and in choosing an appropriate WfMS (Silva & Pinheiro, 2003). These requirements may impose restrictions that will never be completely solved by any supporting technology. For example, the most challenging kind of diversity is cultural diversity. It may be ameliorated using techniques to deal with semantic heterogeneity, but there is more to culture than differences in the interpretation of terms.

REFERENCES

Abiteboul, S., Agrawal, R., Bernstein, P., Carey, M., Ceri, S., Croft, B., et al. (2005). The Lowell database research self-assessment. *Communications of the ACM*, *48*(5), 111-118.

Barghouti, N., & Kaiser, G. (1991). Concurrency control in advanced database applications. *ACM Computing Surveys*, *23*(3), 269-317.

Georgakopoulos, D. (2004). Teamware: An evaluation of key technologies and open problems. *Distributed and Parallel Databases, 15*(1), 9-44.

Georgakopoulos, D., Hornick, M., & Manola, F. (1996). Customizing transaction models and mechanisms in a programmable environment supporting reliable workflow automation. *IEEE Transactions on Data and Knowledge Engineering, 8*(4), 630-649.

Kirchberg, M., Schewe, K-D., Tretiakov, A., & Wang, B. (2007). A multi-level architecture for distributed object bases. *Data and Knowledge Engineering*, *60*(1), 150-184.

Lerner, B. S. (2000). A model for compound type changes encountered in schema evolution. *ACM Transactions on Database Systems, 25*(1), 83-127.

Ray, I., & Xin, T. (2006). Analysis of dependencies in advanced transaction models, *Distributed and Parallel Databases, 20*(1), 5-27.

Schuster, H., & Heinl, P. (1997). A workflow data distribution strategy for scalable workflow management systems. *ACM Symposium on Applied Computing*, (pp. 174-176).

Shankaranarayanan, G., & Even, A. (2006). The metadata enigma. *Communications of the ACM, 49*(2), 88-94.

Silva, L. P., & Pinheiro, F. A. C. (2003). Eliciting requirements for identifying workflow categories. *Proceedings of the VI Workshop on Requirements Engineering (WER'03)* (pp 16-31).

Stohr, E. A., & Zhao, J. L. (2001). Workflow automation: Overview and research issues. *Information Systems Frontiers, 3*(3), 281-296.

Taniar, D., & Goel, S. (2007). Concurrency control issues in grid databases. *Future Generation Computer Systems, 23*(1), 154-162.

Uren, V., Cimiano, P., Iria, J., Handschuh, S., Vargas-Vera, M., Motta, E., & Ciravegna, F. (2006). Semantic annotation for knowledge management: Requirements and a survey of the state of the art. *Web Semantics: Science, Services and Agents on the World Wide Web, 4*(1), 14-28.

van der Aalst, W., ter Hofstede, A. H. M., Kiepuszewski, B., & Barros, A. P. (2003). Workflow patterns. *Distributed and Parallel Databases, 14*, 5-51.

WfMC. (2008). *Workflow management coalition*. Retrieved July 16, 2008, from www.wfmc.org.

KEY TERMS

The definitions below are based on (Georgakopoulos, 2004; Schuster & Heinl, 1997; Stohr & Zhao, 2001; WfMC, 2008).

Content Management Systems: Provide tools for organizing, delivering and sharing documents and images. Usually used in conjunction with CSCW systems or workflow systems.

Computer-Supported Cooperative Work (CSCW): Provides tools for supporting people working together, for example, video and audio conferences, group calendar, e-mail, text chat. Differs from Workflow applications in having more flexibility and less coordination.

Data Partitioning: A storage technique through which each data item is assigned to exactly one node. Data operations accessing different data partitions can be executed in parallel. However, if one operation needs to access more than one data partition, the execution is more complicated.

Data replication: A storage technique through which some nodes have copies of the same data. Replication of data is a common method to improve read performance but is rather problematic if data is often updated.

Metadata Description: Abstract data description used to describe concrete data items. Usually contains descriptions of data structures, relationships to other data items, location, scope, and other features of the data item.

Process Enactment: A WfMS is said to enact the real world process for each process instance.

Task Assignment: The assignment of a task to an agent responsible for it, made by the WfMS. The agent may be a person or a process. When it is a person, the assignment usually implies a notification to the agent; when it is a process, the assignment usually results in its execution.

Workflow Instance: Also, process instance. Each execution instance of a process managed by the WfMS.

Chapter XXV
Politically Oriented Database Applications

Francisco A. C. Pinheiro
Universidade de Brasília, Brazil

INTRODUCTION: THE ROLE OF POLITICS

Technology pervades every aspect of modern life. It has an impact on the democratic life of a nation (Chen, Gibson, & Geiselhart, 2006) and is frequently an object of dispute and negotiation, affecting the way politics is done, by shaping new forms of planning and performing political actions.

The changing of professional politics is fuelled up by the increasing use of information and communication technologies by both citizens and organizations. Attempts to increase the use of technology to run elections (Hawthorn & Simons, 2006) are just one aspect of this movement. Many researchers also identify a steady pace of change on the nature of political parties and the ways politics is performed by politicians: Margetts (2001) notices the emergence of cyber parties, characterized by technologically-aided relationships between party and voters rather then formal membership. Pedersen and Saglie (2005) present a survey discussing how this and other kinds of party organizations (e.g. media and networked parties) affect traditional Scandinavian parties.

Information systems are themselves instruments of power in several ways. As an example, Hayes and Walsham (1999) discuss the political use of information stored in shared databases. Their study shows how databases storing contact and activity information were used by employees and managers to foster they careers, impose their views and control people subordinated to them.

Applications used in or related to politics are information intensive, making databases a prime element in building politically oriented applications. This article discusses some aspects of database related technology necessary for this kind of application.

BACKGROUND: POLITICALLY ORIENTED DATABASES

The politically oriented use of databases can be viewed under three main areas:

- **Research on politics.** Databases used for political science research contain data on a diverse range of social, economical and political aspects of countries and regions. They are called political databases.
- **Professional politics.** This area comprises the party databases, containing data necessary for running political parties, and the government databases, including the electoral databases as well as regulatory and other kind of databases used by parliament, congress, etc. From the government area, this article discusses the electoral databases containing data necessary to run and control elections.
- **Corporate politics.** This area comprises almost every information system, although very few of them explicitly incorporate political aspects in their design.

Political Databases

Political databases are used for political science research. Their structure and content vary depending on their sponsors and objectives but usually they contain a large volume of data about social, economical and political life of countries. They store data on diverse aspects such as elections, electoral rules, type of political system, party composition, and also on more subjective aspects such as military influence on government, stability of government and policies, and the extent of regulation, competition and openness of government actions.

There are a number of political databases regularly referenced by political scientists. The list below is partially drawn from Scartascini and Oliveira (2003) (all indicated websites were visited in May 2008):

Cross National Time Series Data Archive is a comprehensive listing of facts for 256 countries, since 1815 (http://www.databanks.sitehosting.net).

Database of Political Institutions, sponsored by the World Bank, contains information about 177 countries, covering the period from 1975 to 2000 (described in Beck et al., 2001).

Democratic Audit of Australia since 2002 conducts audits to assess Australia's strengths and weaknesses as a democracy (http://democratic.audit.anu.edu.au).

International Institute for Democracy and Electoral Assistance has a website with information on voter turnout for national presidential and parliamentary elections since 1945 for several countries (http://www.idea.int).

Parline Database contains parliamentary information for over 140 countries (http://www.ipu.org).

Political Database of the Americas offers centralized and systematized information about institutions and political processes for 35 countries of America (http://pdba.georgetown.edu).

Political Risk Database provides data on economic indicators, risk ratings, geographic and political data, and social indicators (http://www.prsgroup.com).

Polity series of databases contains a large volume of information on political structures of countries. The Polity IV database contains information about 156 nations covering the period from 1800 to 2004 (http://www.cidcm.umd.edu/polity).

Transparency International Index contains data used to build a corruption perception index for 90 countries since 1995 (http://www.transparency.org).

United Nations Common Database provides access to over 300 data series on economic, social, health, environmental, and other variables for all available countries (http://unstats.un.org/unsd/cdb).

World Bank Datasets provide information on a range of political and economic issues (http://econ.worldbank.org).

Party Databases

Professional politics is performed by politicians and party members, and involves their relationships with other society actors: electorate, media organizations, civilian and military institutions, and so on.

Major parties use databases to assist with their political activities, including the clerical tasks of registering members, their contributions, preferences, and interests. Party databases are used both as a national resource and as a local aid for politicians and party members. Nationally, they help to plan party-wide campaigns and to tailor party communications, allowing for messages targeted to the interests of party supporters. Locally, they are used as a tool for politicians to deal with their electorate, keeping track of theirs interests, behavior and contacts. Party databases may also be used to create advantages based on their proprietary data, such as prospecting potential donors in fundraising campaigns (Wang et al., 2005). One reported difficulty is the reluctance of local users to disclose their data to the party's national bodies (van Onselen & Errington, 2004).

Electoral Databases

Electoral databases are a particular kind of database used in professional politics. They contain information on elections and electoral rules, people and their status as electors, whether they are able to vote, whether they have already voted in a particular election, absenteeism, penalties, and so on. Due to its demographic nature, they are a rich source for several kinds of government actions and socio-political studies. Political parties also use them as a resource to build and maintain their party databases (van Onselen & Errington, 2004).

Reliability and accuracy are paramount characteristics of voter registration systems, and therefore of electoral databases. They should assure public confidence on the election system, contributing to the prevention of fraud.

Corporate Politics

Politics is an essential part of organizational life, affecting and being affected by technology. Decisions to employ technology and, for that matter, to develop information systems are usually driven by business reasons. The success of system implementation is dependent on management support and employees' resistance to change is a frequent cause of burden to the successful adoption of new information systems. The implementation of a new system usually affects some aspects of the organizational culture and may create different forms of resistance to system adoption. There are evidences that cultural changes are often related to distribution of power. Deveaux (2003) argues that "cultural disputes often have more to do with the concrete interests of members and distribution of power in communities than they have with differences in moral value".

The design of systems is influenced by people wanting to gain or expand control and influence, by devising new ways of empowerment through

the use of systems and their data. An important aspect of organizational politics is the emergence of conflicts among parties and the conflict resolution techniques employed to solve them (Johnstone, Huff & Hope, 2006).

MAIN THRUST: DATABASE ISSUES

Data Granularity

It is important for political databases to have disaggregated data. Broad indicators such as whether country elections are free or not may render the database not amenable for comparison with other data (collected from other perspectives). Aggregated data are useful, but they are better modeled keeping the data from which they are aggregate together with a rule used to derive the desired information. For example, the Database of Political Institutions (Beck et al., 2001) utilizes data on the number of parties on government coalition, the number of different parties controlling the legislature and the executive branch in presidential systems, the number of veto players, etc., to derive a measure for the checks and balances of a political system, understood as a measure of how easy is to change policy in a country.

Database Queries

Political databases may store large pieces of text such as laws, procedures, and legislative decisions. This is also true for bibliographical databases on political subjects, and for party databases allowing party members to register their opinions and comments on a variety of issues. This kind of data is highly unstructured and imposes a need for powerful information retrieval facilities.

Schaffer (2001) shows the importance to have a number of query mechanisms; the most common being boolean, keyword, wild cards, adjacency and proximity searches. He also points to the importance of having mechanisms to find related records and cited references.

Security and Protection

Politically oriented applications should control their sensitive data in order to prevent corruption and unauthorized access. Sometimes one should allow only statistical analysis on individual information stored in a database, or only parts of the information should be allowed, or it may be that information should be allowed to some people but not to others.

Dobkin, Jones, and Lipton (1979) state that queries should be treated as a security problem, and the access to information should take into account the history of the query (and not only the relationships between the elements being interrogated). One should be able to remember the sources from which the information was encoded and the history of the information already given out. Frequently a query is composed by several parts, comprising in fact a number of subqueries, each one subject to its own access rules. Only after deciding if a query is allowed one should verify its constituent parts. With respect to electoral databases a simpler mechanism of allowing the minimum access necessary for users to perform their operations may be employed. This is called "principle of least privilege" (Hawthorn & Simons, 2006).

Distributed Data

Politically oriented applications usually deals with the balance between centralization and decentralization as it affects organizational governance. Distribution is also related to empowerment as it may improve the sense of ownership and autonomy. Van Alstyne, Brynjolfsson, & Madnick (1994) say that "Many real world centralization and standardization efforts have failed, typically because departments lacked incentives or needed greater local autonomy".

The use of political databases is inherently distributed, since they are sponsored by different organizations and stored in different places. Party databases may use distribution to cope with local needs of their users. It may be an answer to the reluctance of politicians to disclose to national bodies the data they gathered in their local activities.

Confidentiality and Accountability

Privacy is paramount for electoral systems. Assuring a voter will remain anonymous is at the core of voting systems. Also, the data stored in party databases may not be disclosed, except in aggregated form, for statistical purpose.

In dealing with political information it is important to assure accountability. It should be clear who is responsible for entering and modifying the data, when and by whom the data should be used, the purpose of using it, and the mechanisms in place to enforce these issues.

For politically oriented applications, regular audits, backup and recovery procedures, and other control mechanisms, more than a technical need to rebuild data and restart processes, are a tool to foster confidence in the correct use of databases.

Interoperability, Data Model and Usability

Political databases are built from diverse sources, using different technologies and having different data models. To use them together, as is usual in political science research, requires the understanding of all data models and how they may relate to each other.

Party databases and databases used in politically oriented applications may also have interoperability issues, when they are built from diverse local autonomous databases. Many types of heterogeneity are due to technological differences, for example, differences in hardware, system software (such as operating systems), and communication systems. There are also heterogeneity due to differences in database management systems, including different data models and constraints, query languages, etc., and semantic heterogeneity due to different interpretation of the same term.

FUTURE TRENDS

There is a trend towards a more intensive use of technology in politically oriented applications. We should expect, for example, a more intensive use of party databases and electoral databases, generating debate on how parties and government bodies build their databases and about the kind of data they kept and how they are used. This kind of debate is usually driven by security and fairness concerns. It also raises concerns of misuse and abuse. All of this indicates a need to incorporate politics into the design of politically oriented applications.

Issues related to granularity of data, queries, interoperability, usability, security, protection, confidentiality and accountability are important when designing databases for politically oriented applications. Two databases related technologies are particularly helpful in taking these issues into account: metadata and federated databases.

Metadata

Metadata cover the whole "data life cycle", i.e., they describe the collecting of data from a data source, data storage, data processing and retrieval, and the data dissemination within an electronic data interchange.

The use of metadata may help tackling some issues related to data granularity, database queries data models, and usability. Metadata can also be used to tackle the problem of life cycle misalignment, characterized by the fact that several data records have a lifespan larger than the system on

which they are stored. This is a very common situation regarding electoral records. The knowledge of the source of data and of their semantic and structural characteristics makes data models more comprehensible, greatly enhancing the effectiveness of politically oriented application (Meng, Yu, & Liu, 2002).

Federated Databases

Federated databases are a natural response to cope with distribution and heterogeneity. They may be used to preserve locality and to foster the sense of ownership and autonomy necessary for many politically oriented applications.

Data may be distributed among multiple databases in different ways. These databases may be stored on a single computer system or on multiple computer systems, co-located or geographically distributed but interconnected by a communication system (Haas, Lin, & Roth, 2002; Sheth & Larson, 1990).

CONCLUSION

The shrinking of political party's membership and the role of mass media in shaping political communication and, to a large extent, the political agenda (Callaghan & Schnell, 2001), placing us in a era of perpetual election campaigning, may arguably be viewed as a drawback. Nevertheless, the role of technology is unavoidable and it may also facilitate life, being a motif of change and evolution (Dunleavy et. al., 2002). A first step is to acknowledge the importance of politically oriented applications and to design them taking politics as a prime aspect.

By its nature, based on negotiation of positions, ideas and principles, politics is carried out through the exchange of information between the involved parties (Peled, 2001). In this sense databases are a key element for incorporating political aspects into system design.

Recognizing the role of politics would help in designing databases that more appropriately capture the political needs of stakeholders. Issues like privacy, sensitive information, rule-making processes, and ownership could be shifted from a mere technical discussion to a proper political consideration.

REFERENCES

Avan Alstyne, M., Brynjolfsson, E., & Madnick, S. E. (1994). *Why not One Big Database? Principles for Data Ownership.* Sloan School of Management, Technical Report WP 3695-94. Cambridge, MA: Massachusetts Institute of Technology.

Beck, T., Clarke, G., Groff, A., Keefer, P., & Walsh, P. (2001). New Tools in Comparative Political Economy: the Database of Political Institutions. *The World Bank Economic Review, 15*(1), 165-176.

Callaghan, K., & Schnell, F. (2001). Assessing the Democratic Debate: How the News Media Frame Elite Policy Discourse. *Political Communication, 18*(2), 183-213.

Chen, P., Gibson, R., & Geiselhart, K. (2006). *Electronic Democracy? The Impact of New Communications Technologies on Australian Democracy.* Democratic Audit of Australia, Australia National University.

Deveaux, M. (2003). A Deliberative Approach to Conflicts of Culture. *Political Theory, 31*(6), 780-807.

Dobkin, D., Jones, A. K., & Lipton, R. J. (1979). Secure Databases: Protection Against User Influence. *ACM Transactions on Database Systems, 4*(1), 97-106.

Dunleavy, P., Margetts, H., Bastow, S., Callaghan, R., & Yared, H. (2002). *Government on the Web II.* London School of Economics, LSE Research Online, Technical Report, London, UK.

Haas, L. M., Lin, E. T., & Roth, M. A. (2002). Data Integration through Database Federation. *IBM Systems Journal, 41*(4), 578-595.

Hawthorn, P., Simons, B., et al. (2006). *Statewide Databases of Registered Voters: Study Of Accuracy, Privacy, Usability, Security, and Reliability Issues*. U.S. Public Policy Committee of the Association for Computing Machinery, Washington, D.C., U.S.

Hayes, N., & Walsham, G. (1999). Safe Enclaves, Political Enclaves and Knowledge Working. *First International Critical Management Studies Conference*, Manchester, England.

Johnstone, D., Huff, S., & Hope, B. (2006). IT Projects: Conflict, Governance, and Systems Thinking. *The 39th Hawaii International Conference on System Sciences.*

Margetts, H. (2001). The Cyber Party. *The Causes and Consequences of Organizational Innovation in European Political Parties Workshop, ECPR Joint Sessions of Workshops,* Grenoble.

Meng, W., Yu, C., & Liu, K-L. (2002). Building Efficient and Effective Metasearch Engines. *ACM Computing Surveys, 34*(1), 48-89.

van Onselen, P., & Errington, W. (2004). Electoral Databases: Big Brother or Democracy Unbound? *Australian Journal of Political Science, 39*(2), 349-366.

Pedersen, K., & Saglie, J. (2005). New Technology in Ageing Parties: Internet Use among Danish and Norwegian Party Members. *Party Politics, 11*(3), 359-377.

Peled, A. (2001). Networks, Coalition, and Institution: the Politics of Technological Innovation in the Public Sector. *Information Technology & People, 14*(2), 184-205.

Scartascini, C. G., & Oliveira, M. (2003). *Political Institutions, Policymaking Processes and Policy Outcomes: A Guide to Theoretical Modules and Possible Empirics.* Design Paper, Inter-American Development Bank, American Research Network.

Schaffer, T. (2001). Databases and Political Science Research. *Online Information Review, 25*(1), 47-53.

Sheth, A. P., & Larson, J. A. (1990). Federated Database Systems for Managing Distributed, Heterogeneous, and Autonomous Databases. *ACM Computing Surveys, 22*(3).

Wang, K., Zhou, S., Yang, Q., & Yeung, J.M.S. (2005). Mining Customer Value: From Association Rules to Direct Marketing. *Data Mining and Knowledge Discovery, 11,* 5779.

KEY TERMS

Cyber Party: A political party whose relations to their supporters and voters are technologically mediated, instead of being based on membership.

Electoral Database: A database used to support election systems, storing information on electoral sections, candidates, voters, etc.

Federated Database System: A collection of cooperating but autonomous component database systems.

Metadata: Data about data, containing semantic information describing its source, structure and interpretation.

Party Database: Database containing data used to run party's business

Political Database: A database containing data useful for political research.

Voter Registration Database: A kind of electoral database used primarily to control and conduct elections.

Chapter XXVI
Semantically Modeled Databases in Integrated Enterprise Information Systems

Cheryl L. Dunn
Grand Valley State University, USA

Gregory J. Gerard
Florida State University, USA

Severin V. Grabski
Michigan State University, USA

INTRODUCTION

Semantically modeled databases require their component objects to correspond closely to real world phenomena and preclude the use of artifacts as system primitives (Dunn and McCarthy, 1997). Enterprise information systems (also known as enterprise resource planning systems) based on semantically modeled databases allow for full integration of all system components and facilitate the flexible use of information by decision-makers. Researchers have advocated semantically designed information systems because they provide benefits to individual decision-makers (Dunn and Grabski, 1998, 2000), they facilitate organizational productivity and inter-organizational communication (Cherrington et al., 1996; David, 1995; Geerts and McCarthy, 2002), and they allow the database to evolve as the enterprise does through time (Abrial, 1974).

Organizations have implemented enterprise resource planning (ERP) systems in an attempt to improve information integration. Much of the value of these ERP systems is in the integrated database and associated data warehouse that is implemented. Unfortunately, a significant portion of the value is lost if the database is not a semantic representation of the organization. This

value is lost because the semantic expressiveness is insufficient -- relevant information needed to reflect the underlying reality of the organization's activities is either not stored in the system at all, or it is stored in such a way that the underlying reality is hidden or disguised and therefore cannot be interpreted.

Partly as a result of systems lacking expressive semantics, researchers have been developing ontologies. Gruber (2008) provides a useful definition of ontology:

"In the context of database systems, ontology can be viewed as a level of abstraction of data models, analogous to hierarchical and relational models, but intended for modeling knowledge about individuals, their attributes, and their relationships to other individuals. Ontologies are typically specified in languages that allow abstraction away from data structures and implementation strategies; in practice, the languages of ontologies are closer in expressive power to first-order logic than languages used to model databases. For this reason, ontologies are said to be at the "semantic" level, whereas database schema are models of data at the "logical" or "physical" level. Due to their independence from lower level data models, ontologies are used for integrating heterogeneous databases, enabling interoperability among disparate systems, and specifying interfaces to independent, knowledge-based services."

We base our discussion in this paper on the Resources-Events-Agents (REA) ontology (McCarthy, 1982; Geerts and McCarthy 1999; 2000; 2004; 2001; 2002; Haugen and McCarthy, 2000) which is considered an enterprise ontology or a business domain ontology. Ontologically-based information systems with common semantics are regarded as a necessity to facilitate inter-organizational information systems (Geerts and McCarthy, 2002). Presently, most inter-organizational data is sent via EDI (which requires very strict specifications as to how the data are sequenced and requires some investment by adopting organizations). The same requirement holds true for web-based systems. There is no or very limited knowledge inherent in those systems. Alternatively, if trading partners implement systems based on the same underlying semantic model, many of the current problems can be eliminated.

This chapter first presents a normative semantic model for enterprise information systems that has its roots in transaction processing information systems. We use this model because the majority of information processed and tracked by information systems is transactional in nature. We review empirical research on semantically modeled information systems and then provide an example company's semantic model as a proof of concept. We next discuss how this model can be applied to ERP systems and to inter-organizational systems and present future trends and research directions, and provide concluding comments.

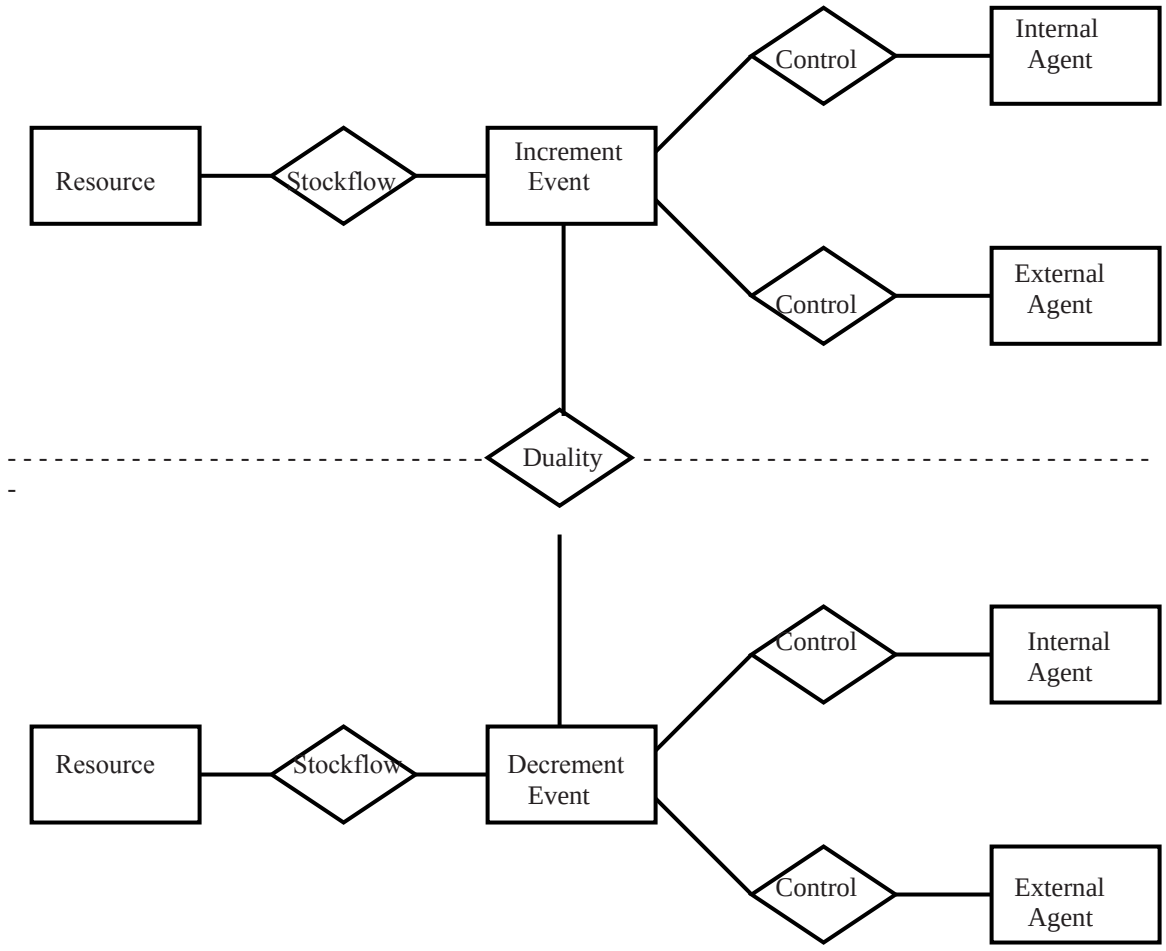
Semantic Model Development

In this chapter, we adopt a definition of an enterprise information system that is based on David et al.'s (1999) definition of an accounting information system: an enterprise information system that captures, stores, manipulates, and presents data about an organization's value-adding activities to aid decision-makers in planning, monitoring, and controlling the organization. This definition is also consistent with much of the research on ERP systems. We recommend that the REA ontology (REA semantic model) (McCarthy, 1982) be used as the core foundation of enterprise information systems due to the model's robust and general nature. The semantics of the REA model are designed to capture the essential features of value added activities – activities that correspond to exchanges of resources (e.g., giving inventory and receiving cash) and transformations of resources (converting raw materials into finished goods). The basic REA model is presented in figure 1 using

entity-relationship notation (Batini et al., 1992), however, it has also been implemented in NIAM (Geerts and McCarthy, 1991) and in object notation (Nakamura and Johnson, 1998). The general REA model for any particular transaction cycle consists of the following components (these components are presented in list form for the sake of brevity). Readers are encouraged to read McCarthy (1982) and Dunn et al. (2005) for more detail.

- Two economic events that represent alternative sides of an economic exchange (one increment event and one decrement event).
- Two resources that represent what is received and given up in the economic exchange.
- Two internal agents that represent the company's personnel responsible for the economic events (usually one agent type for each event).
- One external agent that represents the person or company with whom the company is engaging "at arms' length" in the exchange.
- Duality relationship between the increment and decrement economic events.
- Stock-flow relationships between the events and the associated resources, representing

Figure 1. The original REA model (McCarthy, 1982)



the inflows or outflows of the resources resulting from the events.
- Responsibility relationships between the events and the internal agents.
- Participation relationships between the events and the external agents.

The general model outlined above can be modified to accommodate specific needs. The original model was described at only the operational level, which is sufficient for historical accounting purposes. The REA model has now been extended to an inter-organizational database systems ontology that incorporates both the operational (what has happened) and knowledge levels (what should happen as a policy or expectation) and includes the following additional features and components, again presented in list form (Geerts and McCarthy, 2002).

- Integration of process (cycle) level models into a value-chain level model at a higher level of abstraction.
- Expansion of process (cycle) level models into workflow or task level models at lower levels of abstraction.
- Separation of components into continuants (enduring objects with stable attributes that allow them to be recognized on different occasions throughout a period of time) and occurrents (processes or events that are in a state of flux).
- Type images that represent category level abstractions of similar components.
- Commitment images that represent agreements to engage in future economic events.
- Assignment relationships between agents that represent the designation of an agent category to work with another agent category (e.g. salesperson assigned to customer).
- Custody relationships between agents and resources that represent the agents that are accountable for various resources.
- CommitsTo relationships between commitment images and the resulting economic events.
- Partner relationships between commitment images and the participating agents.
- Reserved relationships between commitment images and the resources that are the proposed subject of the future exchange.
- Typification description relationships between continuant components and the categories to which they belong, e.g. resource-resource type relationships.
- Characterization description relationships between continuant type images, e.g. agent type-agent type relationships and agent type-resource type relationships.
- Typification history relationships between physical occurrents and their types, indicating that the occurrents share the same script, e.g. event-event type.
- Scenario history relationships between abstract occurrents and other abstractions, e.g. event type-resource type.
- Business Process as a description of the interaction between resources, agents, and dual events.
- Partnering as the purpose of the interaction between resources, agents, and commitments.
- Segmentation as a description of the grouping of physical categories into abstract continuant categories.
- Policy or Standard as a description of the expression of knowledge level rules between abstract types (e.g. scripts and scenarios).
- Plan as a description of the application of a script to physical occurrents.
- Strategy as a description of rules for the execution of a Business Process or Partnering.

The semantics contained within the expanded REA model facilitate the information exchange between trading partners and likely provide a

needed core foundation for ERP systems. For example, research reported by Andros et al. (1992) and Cherrington et al. (1996) found that IBM was able to obtain significant benefits from a semantically modeled system based on the REA model. Reported benefits of the new system included a significant reduction in the time to process employee reimbursements, significant cost reductions, and a generally high level of employee satisfaction. These studies demonstrate how the semantic models were used as the basis for systems design, and then how the resultant systems were perceived by the end-users, thereby completing the research loop from design to end-user.

Weber (1986) empirically evaluated the REA semantic model. His objective was to determine whether software practitioners had both identified and solved the same problems as identified by academicians. He found that the REA model fulfilled its objective as a generalized model and that it was a good predictor of the high-level semantics found in all twelve packages. He believed that the REA model excluded certain types of events such as contracts and suggested extensions (many of which have been subsequently incorporated into the model). Weber reported that important differences in the low-level semantics existed across all packages, and that these seem to reflect the relative complexity of the different packages. Nonetheless, the clustering of the low-level semantics of the packages according to the REA model allowed for the identification of similarities and differences among the packages, and the likely strengths and limitations of the package become apparent. On a datalogical level, Weber found few violations of normal form given the number of fields in the packages. He concluded that theory was a good predictor of design practice and that the empirical evidence supports normalization theory.

David (1995) developed a metric to classify organizations' accounting systems characteristics (ASC) along a continuum between traditional general ledger based accounting systems and REA systems. The ASC metric was developed based upon characteristics that were identified in theoretical research as critical characteristics for REA systems. David visited companies in the pulp and paper industry and conducted structured interviews and used the ASC metric to determine each system's position on the continuum. Data was also gathered as to the companies' productivity, efficiency, and the company executives' perceptions of competitive advantage. REA-like systems were found to be associated with productivity and administrative efficiencies.

O'Leary (2004) compared the REA semantic model with an enterprise resource software package. He compared information about SAP from various sources and determined that SAP is consistent with the REA model in its database, semantic and structure orientations. However, SAP was also found to contain implementation compromises in the structuring and semantic orientation, based in part on accounting artifacts. Part of the compromises are likely due to the fact that software vendors' products are constantly evolving in an incremental fashion, and vendors never do a software redesign starting with a clean slate.

Research has demonstrated that organizations are able to obtain significant benefits from a semantically modeled system based on the REA framework (Andros et al., 1992; Cherrington et al., 1996). However, even semantically modeled systems are not sufficient to ensure success. Rather, success is dependent upon a variety of factors including top management support and corporate philosophy in addition to the benefits inherent in the semantically modeled system (Satoshi, 1999). Research has also demonstrated that the REA model is sufficient as a generalized model (Weber, 1986; O'Leary, 2004), and the more REA-like a system is, the more it is associated with productivity and administrative efficiencies (David, 1995).

We next present an REA model of a prototypical retail firm. This example will serve as further

proof of concept and allow the reader to verify that the semantic data model accurately represents the reality of the described organization, and that there is an application of the REA model with consistent patterns. First, we will provide a narrative for the firm and describe in detail the sales cycle. Then we will describe the components of a REA model (in entity-relationship notation using guidelines set forth by Batini et al., 1992) that captures the reality. Finally, we will discuss the benefits of the REA model that occur independent of the entity-relationship notation. Our example is for a company called Robert Scott Woodwinds (RSW), a retail organization that sells, leases and repairs woodwind musical instruments, and sells music supplies. This organizational setting encompasses many of the activities performed by most business organizations. While the semantics of the model are specific to RSW's environment, the model provides a generic template for a retail organization's semantic model, it also provides a template for leasing organizations, and it has significant commonalities with manufacturing firms in the repair aspects of the business.

REA Semantic Model Example

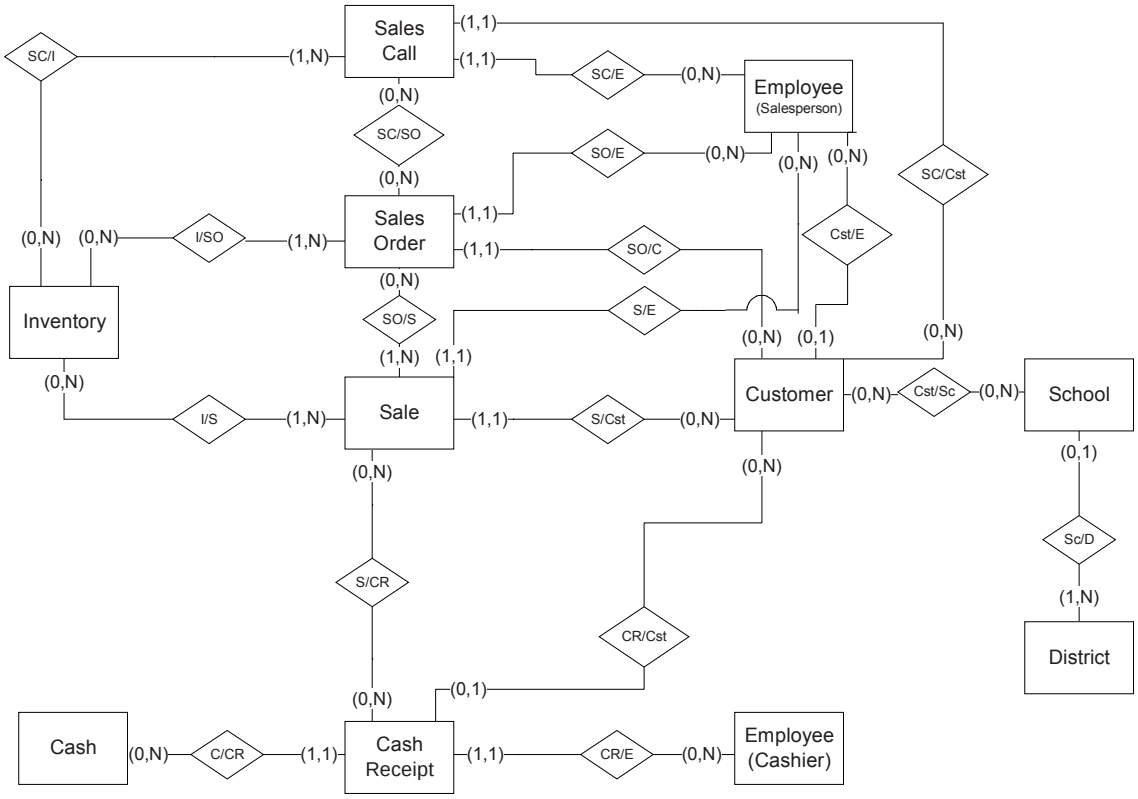
For the scenario presented below, the resources, events, and agents are captured in REA diagrams (in entity-relationship form) in figures 2 through 5. RSW sells, repairs, and leases musical woodwind instruments from a single store. The business caters to professional musicians, casual musicians, and children who are involved in school music programs. While RSW clearly needs to have a web presence, it anticipates only a minor portion of its revenues from web sales (primarily for consumables such as reeds and sheet music). Professional musicians want to play and get a feel for the particular instrument that they are interested in purchasing and will always come into the store. Further, these musicians are very particular about who will repair their instrument. For repairs, the instrument must be brought or sent to the store. School sales are a result of sales staff calling on the band directors at the various schools and gaining permission to be the supplier for that school (usually this means that the children will rent from RSW, not that the school will purchase the instruments from RSW). Following is information about RSW's revenue (including sales, sales returns, repairs, and leases) cycle.

Revenue Cycle

RSW generates revenue in three ways. The first is through sales of instruments; the second is through rental of instruments; and the third is through the repair of instruments. In the retail instrument business, renting instruments so that customers can "try before they buy" is a great way of generating sales. This is particularly true in the school marketplace. Thus, RSW sends salespeople (who also have music teacher training) into the schools to host "music talent exploration" sessions for the children. When a child has been identified as having talent for (and some interest in) a specific instrument, the parents will be offered the opportunity to rent that instrument for 3 months. At the end of the 3-month trial period, the parents may purchase that instrument (or a different instrument) and the sale price they pay will reflect a discount equal to the amount of rent paid. When an instrument is rented out, it is transferred from "Inventory for Sale" to "Inventory for Rent". If it is later sold, the "Inventory for Rent" category will be decreased.

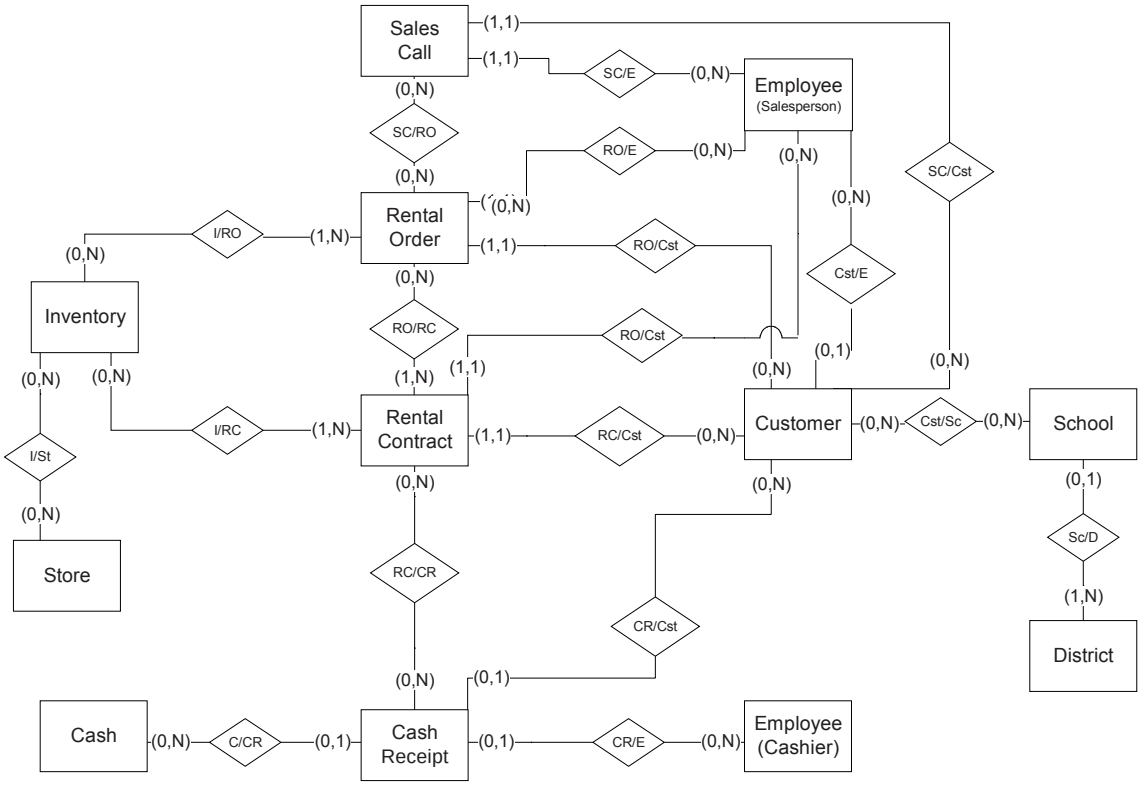
RSW's salespeople complete sales invoices and rental invoices for customers. The customer may purchase (rent) multiple instruments on a single sales (rental) invoice. Instruments are either delivered to the customer at the school during the salesperson's next sales call, or the customer may pick up the instruments at the store. Cash receipts are processed by cashiers and are deposited into the company's main bank account each day. The semantic data models for sales and for leases are presented in figures 2 and 3 respectively.

Figure 2. RSW sales revenue expanded REA data model



As can be seen in the sales revenue data model (figure 2), the original REA template has been applied. The resource (inventory) is associated with an economic decrement event (sale) via a stockflow relationship. The sale event is associated with the internal agent (salesperson) and the external agent (customer) via control relationships. There is a duality relationship between the economic decrement event (sales) and the economic increment event (cash receipt). Cash receipt is associated with the resource (cash) via a stockflow relationship and to the internal agent (cashier) and the external agent (customer) via control relationships. Note that the cardinalities between sale and cash receipt disclose that a sale does not require the immediate receipt of cash. Rather, a sale may be paid for at one later point in time, in installments over a period of time, or not paid for at all (because of a sale return or bad debt). Newer aspects of the REA ontology have also been included in this model. The sales order event is a commitment image and is related to the resulting economic event (sales) via a CommitsTo relationship. It is also related to salesperson via a Control relationship, to customer via a Partnering relationship, and to inventory via a Reserved relationship. In addition, there is an Assignment relationship established between salesperson and customer. Sales personnel can be assigned to many different customers, and a customer has, at most, one sales person assigned to them. The REA ontology allows for the inclusion of additional entities that managers need information about for planning, controlling and evaluating individual and organizational performance. These additions do not change the underlying REA template; they

Figure 3. RSW rental revenue cycle expanded REA data model

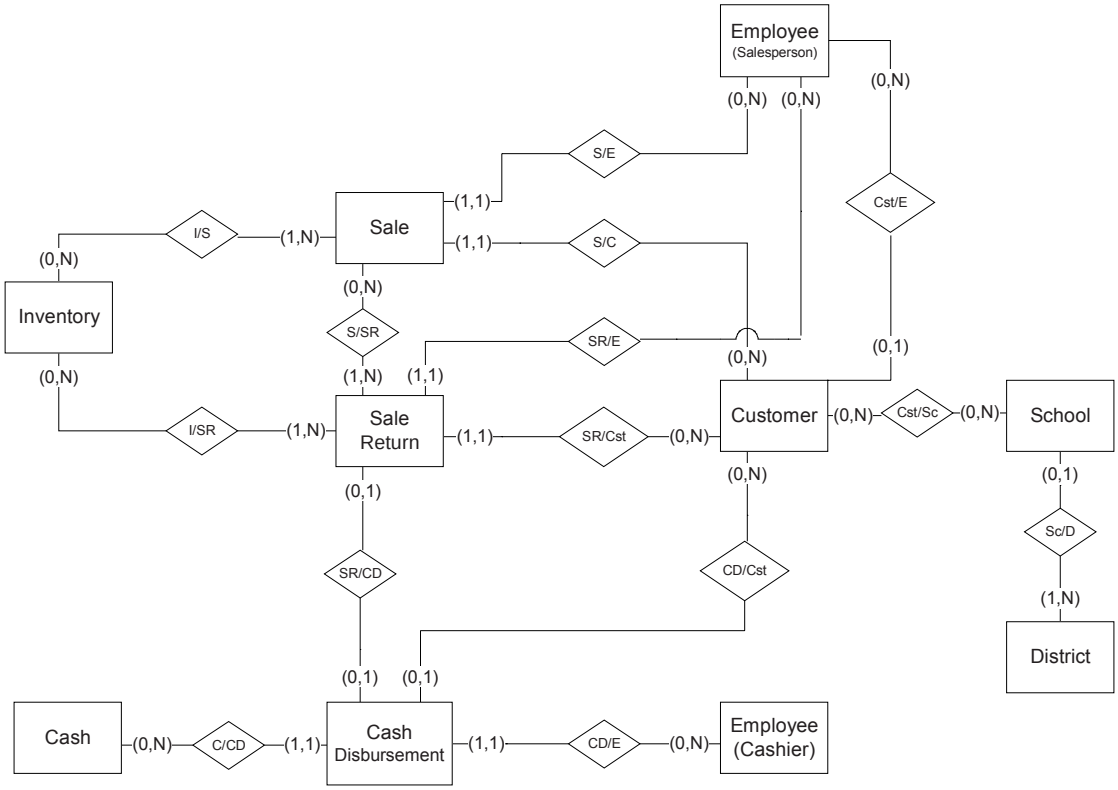


simply add to it. For example, the sales call event that leads to the sales order event is modeled, and is linked to the associated resource and agents. Further, the external agent set has been expanded to include school and district. A customer does not need to be associated with a school, and a school may be listed even if no customers from that school have begun to trade with RSW. A school does not need to belong to a school district (i.e., it is a private school), however, if it belongs to a school district, it can only belong to one school district. A school district has at least one school.

The information presented in the semantic model is a representation of the reality of the world of RSW. The cardinalities contain the "business rules" that are followed by the entities in their relationships. A similar analysis can be performed for the lease revenue data model (figure 3), and in fact, the only differences between the two are the events of rental order and rental contract in place of sales order and sale. The observation of such commonalities allows for the development of common business practices for all revenue activities and facilitates reengineering at the task level.

In a similar manner to the sales and leases, a semantic model of sales returns and lease returns can also be created. The sale return data model (figure 4) has the economic decrement sale event associated with the economic increment event sale return, and the sale return is associated with the subsequent economic decrement event cash disbursement (the sale return might not result in a cash disbursement if no cash was received at the time of the sale; this is represented by the 0 min cardinality on the sale return). The rental returns data model is analogous (the difference is in the economic increment event of rental return which is

Figure 4. RSW sales returns expanded REA data model



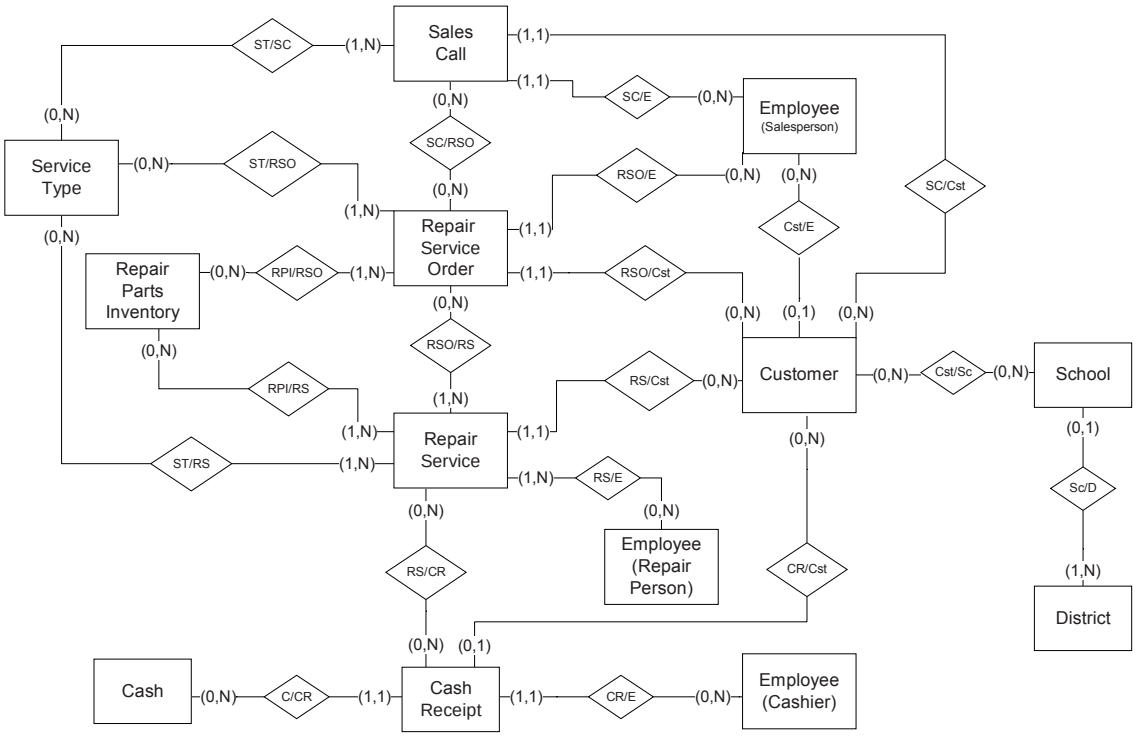
associated with the rental contract, and this model is not presented in the interest of brevity)

The third form of revenue generated by RSW is through instrument repair services. RSW offers a variety of repair service types. "Complete overhauls" are available for many instruments. Prices of these complete overhauls vary by instrument type. "Repad overhauls" are also available for each instrument type. These are less expensive than complete overhauls and price varies by instrument type. Customers may have individual keys replated at a fixed cost per key. Crack repairs will be judged as minor or major and will be priced accordingly. Other repairs are also available. The semantic model for repair revenue is presented in figure 5.

In the repair service REA model, more of the features of the newer REA ontology are introduced. Typification is used, with an event to event-type relationship between repair service and service type and also a commitment-commitment type relationship between repair service order and service type. The service type entity represents the details of the various repair service types RSW can provide (e.g. complete overhauls, repad, etc.). This serves as the bill of materials for the service and allows storage of information about the "categories" to which the repair services belong. Another internal agent is also included, that of repair person. This allows RSW to track the person was responsible for a given repair. This is similar to any type of job shop or organization that must track employee time (e.g., consulting firms).

For simplicity and in the interest of brevity, we do not present the acquisition, payroll or financ-

Figure 5. RSW repair service revenue expanded REA data model



ing cycles. The acquisition model is similar to the revenue cycle, with cash disbursement being the decrement event, purchase the increment event and vendor is the external agent. The payroll cycle would be similar to the acquisition cycle except that services are acquired from employees (hence employees act as external agents) instead of from vendors or suppliers. The financing model is also similar to the acquisition process except that the resource acquired and consumed is cash. Sometimes the flow of resources may not be measurable, for example, in the payroll cycle the resource acquired is employee labor. This is an intangible resource that may be difficult, if not impossible to measure in terms of stockflow. The REA ontology requires that such things be considered and only left out of the model if they are impossible to implement.

The REA ontology can be implemented using alternative notations, including object notation, UML, and others. We have chosen to represent it with entity-relationship notation because of its simplicity and widespread use. The benefit of the REA ontology is not found in the notation itself, but in the repeated application of the pattern to all of the various business cycles. Consistent use of the template gives users an ability to understand the reality being represented, and it also provides system reusability and extendibility. Dunn and McCarthy (2000) describe four benefits of the REA model. First, the standardized use and definition of information structures across organizational boundaries facilitates electronic commerce, by enabling the retrieval, sending, and integration of data among business partners or research sources. Second, intellectual complexity is made manageable by the use of natural primitives that abstract to generalized descriptions of structures which in turn cover many thousands of cases with as few exceptions as possible. Third, consistent use of the

REA pattern can enable system-based reasoning and learning. Fourth, use of system primitives that closely mirror the underlying phenomena enables system adaptability.

It is important to note that the REA ontology and other semantic modeling approaches have been studied at the individual decision-making level, both for system designers and users. Dunn and Grabski (2002) provided an extensive literature review of individual-level semantic models research. Overall conclusions of their review were as follows. Accounting systems based on the REA model are perceived as more semantically expressive by end users than are accounting systems based on the traditional debit-credit-account model. Also, accounting systems perceived as semantically expressive result in greater accuracy and satisfaction by end-users than do non-semantically expressive accounting systems (Dunn and Grabski, 2000). Conceptual modeling formalisms are superior to logical modeling formalisms for design accuracy (Sinha and Vessey, 1999; Kim and March, 1995). The ability to disembed the essential objects and relationships between the objects in complex surroundings depends on a cognitive personality trait, field independence and leads to more accurate conceptual model design (at least for undergraduate students) (Dunn and Grabski, 1998). The focus on increment and decrement resources and events along with the associated agents to those events is consistent with database designers' thought processes. Additionally, knowledge structures consistent with the REA template's structuring orientation are associated with more accurate conceptual accounting database design (controlling for knowledge content, ability, and experience level) (Gerard, 2005). The lack of mandatory properties with entities is not critical (Bodart et al., 2001), perhaps because of the semantics inherent in the modeled system. System designers distinguish between entities and relationships, with entities being primary (Weber, 1996). Data and process methodologies are easier for novices than the object methodology, and resulted in less unresolved difficulties during problem-solving processes (Vessey and Conger, 1994), and there is a more pronounced effect for process-oriented tasks (Agarwal, Sinha, and Tanniru, 1996a). Experience in process modeling matters, regardless of whether the modeling tool (process versus object-oriented) is consistent with the experience (Agarwal, Sinha, and Tanniru, 1996b).

Semantic Models, ERP Systems, and Inter-Organizational Systems

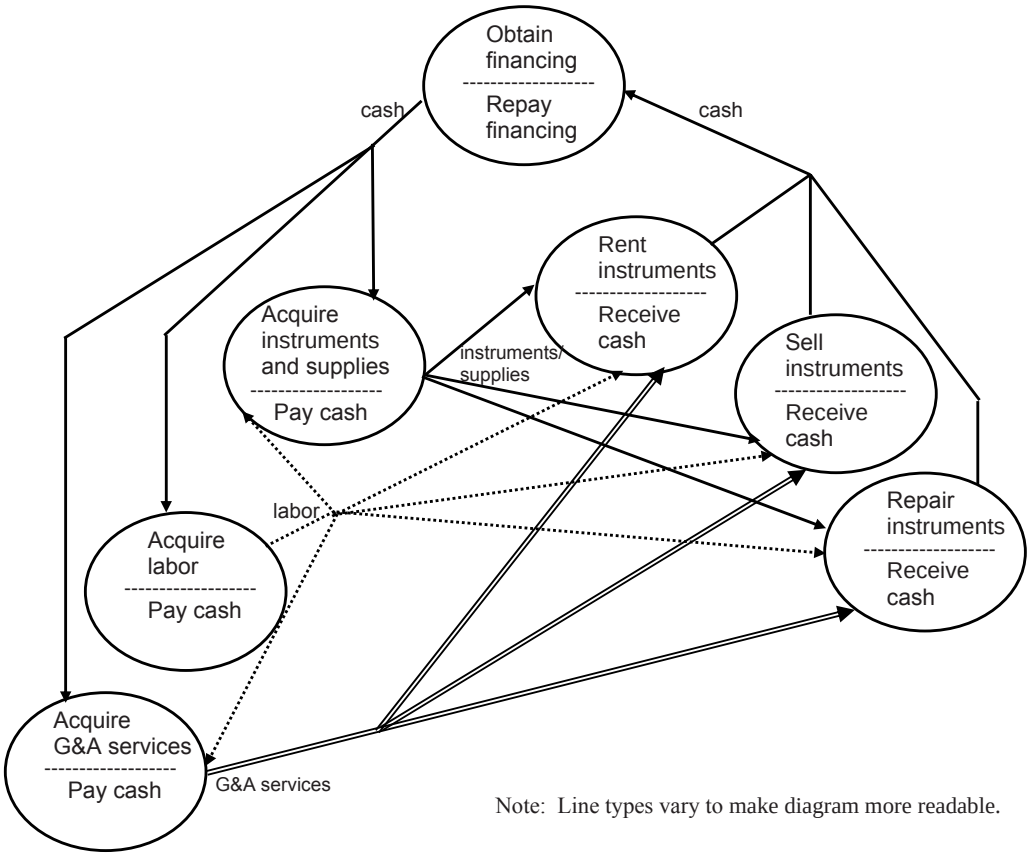
The American Production and Inventory Control Society (APICS, 1998) defined an ERP system as "an accounting-oriented information system for identifying and planning the enterprise-wide resources needed to take, make, ship, and account for orders." David et al. (1999) proposed using REA as a basis for comparison among systems and ERP packages. This was based, in part, on Weber (1986) who reported that a comparison at the level of symbol sets made semantic similarities and differences apparent. Additionally, O'Leary (2004) compared SAP to the REA model and determined that SAP is REA-compliant; however, SAP has significant implementation compromises based on accounting artifacts. Consistent with David et al. and based upon the APICS definition of ERP systems, we believe that REA is a robust candidate to which ERP systems may be compared because of its strong semantic, microeconomic and accounting heritage. More importantly, we believe that semantic models must be used as a basis for the information system because of the information contained within the semantics.

Watson and Schneider (1999) also emphasized the importance of ERP systems in providing the process-centered modeling perspective necessary for successful organizations. They emphasized understanding the underlying process model inherent in ERP systems. That is, the underlying semantics of the system. Further, the SAP

R/3 business blueprint provides four enterprise views; process, information, function, and organization. It is in the organization view that the semantic relationships among organizational units are represented. As most ERP implementations will require some modifications to meet the specific business needs, the analysts, designers, and users need to understand the changes and their subsequent planned and unplanned impact. A semantically based model will facilitate this understanding.

The REA ontology provides a high-level definition and categorization of business concepts and rules, enterprise logic, and accounting conventions of independent and related organizations (Geerts and McCarthy, 2002). The REA ontology includes three levels: the value chain level, the process level, and the task level. The value chain level models an enterprise's "script" for doing business. That is, it identifies the high-level business processes or cycles (e.g. revenue, acquisition, conversion, financing, etc.) in the enterprise's value chain and the resource flows between those processes. The process level model represents the semantic components of each business process. The example in section 2 depicted the process level of the REA ontology for RSW. The task (or workflow) level of the REA ontology is the most detailed level, and includes a breakdown of all steps necessary for the enterprise to accomplish the business events that were included at the process level. The task level can vary from company to company

Figure 6. RSW value chain level model



Note: Line types vary to make diagram more readable.

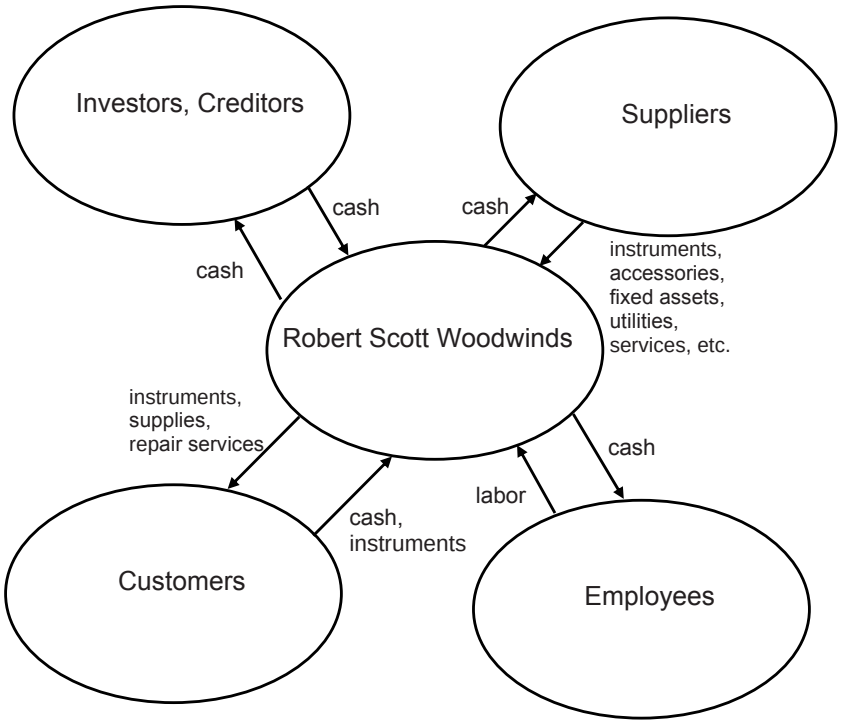
without affecting the integration of processes or inter-organizational systems; therefore we do not present an elaboration of this level (it is also the level at which most reengineering occurs). An illustration of the value chain level for RSW is presented in figure 6.

The value chain level describes how the business processes within the company fit together and what resources flow between them. In RSW, financing is obtained and as a result, cash is distributed, as needed to the various acquisition processes. Some of that cash is used to acquire labor, which is then in turn distributed to each of the other business processes. Some of the cash is used to acquire instruments and supplies (both supplies for sale and also supplies used in the repair services). Some of the cash is used to acquire G&A services. The instruments, supplies, G&A services, and labor are received by the three revenue processes and are combined in various ways to generate revenue. The resulting cash is then distributed to the financing process where it is used to repay financing (including distribution of earnings to stockholders if this were a corporation) or to re-distribute cash to the other business processes. The value chain level demonstrates integration opportunities for the enterprise-wide information system. Resources that flow from one business process to another can be modeled in the database one time and accessed by the appropriate business processes. Then as the separate business processes have their events linked to those resources, the entire system is integrated and can be examined at a level of abstraction higher than the individual business process level.

For inter-organizational system modeling, a fourth level could be added to the REA ontology – the value system level. This enables an integration that spans the conceptual models of each organization to recognize that many business activities and phenomena such as electronic commerce and supply chain management that involve multiple organizations. A company's value system consists of its relationships with all other entities with which the company makes exchanges. For example, a company engages in exchanges with its employees, its suppliers, its customers, and its creditors. An enterprise system model at this level would represent the resource flows between the various types of organizations in the enterprise's value system. For example, for RSW, the value system model is depicted in Figure 7. Partnering organizations that build their enterprise databases on the REA ontology will be better able to integrate their databases. This will probably be best-accomplished using object technology and artificial intelligence concepts such as automated intensional reasoning (Geerts and McCarthy, 1999) and automated intelligent agents. Automated intensional reasoning systems make inferences based on database table intensions, and require completely consistent use of any underlying pattern such as REA.

Although REA is being used by some consultants and software developers, as indicated in the literature review in section 2 (also see e.g., REA Technology at http://reatechnology.com/ and Workday at http://www.workday.com), there are various other approaches to integrating systems within and across organizations. Approaches to integrating systems within organizations have primarily been ERP software-driven and have also focused on document exchange. Wakayama et al. (1998) state that "rethinking the role of documents is central to (re)engineering enterprises in the context of information and process integration" because documents "are a common thread linking integration issues." Electronic data interchange is certainly document-oriented, with standards developed for issues such as which field on an electronic purchase order transmitted between companies represents the quantity ordered. More recent advances for web-based transmission of documents, such as various markup languages, have also focused on identifying fields that exist on these documents.

Figure 7. RSW value system level model



We believe it is not the documents themselves that provide the common thread among company's business processes. Rather it is the similar nature of the underlying transactions and information needs associated with managing these transactions that are depicted within the organization's semantic model. Although organizations can have very different "scripts" for doing business, the core of the scripts is often the same. Therefore, rather than taking a document-oriented approach to inter-organizational system integration, we believe it is more important to apply an enterprise ontology approach such as REA.

Vendors of enterprise resource planning systems recognize the need for inter-organization integration and supply chain management. This integration issue has not been easily resolved. "Bolt-on" applications are typically used. It is possible that some of the difficulties in integrating systems across organizations with ERP systems are due to the requirement of many ERP packages that organizations conform to the software (in the form of "best practices") at the task level. REA encourages conformance to a core pattern at the business process level (or at least to a core set of pattern components), but allows considerable variation at the task level. We suggest that the integration of organizations' information systems does not need to occur at the task level; it can occur at the value system and process levels where the core pattern components can easily be standardized (however, as noted above, the systems need to include task level specifications of homonyms and synonyms to facilitate integration). As Swagerman, Dogger, and Maatman (2000) note, standardization of patterns of behavior ensure the semantic context is clear to every user. We believe the standardization of patterns

and pattern components along with the development of appropriate artificial intelligence tools will allow system integration without imposing formal structures on the social domain.

Limited research has been conducted as to the similarities of the semantic models underlying the current ERP packages. Nonetheless, these models do exist and many organizations reengineer themselves to become consistent with the best practices embodied within these models. Unfortunately, this is often at the task level and the benefits of the underlying semantics are lost. This is very apparent when organizations seek to extend their value chains up and down their supply chain. If the underlying semantics were preserved, along with the standardization of semantic patterns, then automated intensional reasoning and other knowledge-based tools would be able to facilitate the inter-organizational trading. Semantically modeled enterprise information systems will provide many benefits, from the individual decision maker level to the inter-organizational level. The critical issue is to ensure that the semantics are not lost upon the implementation of the system and obscured by the task level mechanics. When this occurs, all subsequent benefits are lost and we are faced with the task of integrating disparate systems that are conceptually identical.

FUTURE TRENDS IN SEMANTICALLY MODELED SYSTEMS

Electronic commerce is commonplace; however the majority of transactions in business to business electronic commerce rely on the transmission of data based on rigidly structured electronic data interchange formats. We believe that the use of an ontological modeling approach such as the REA ontology discussed in the previous sections has the potential to enhance business to business electronic commerce.

The issues related to supply chain management are similar to e-commerce. In fact, when examining inter-organizational systems from a supply chain perspective, the same set of issues apply as found in business-to-business e-commerce. Consequently, our recommendations and expectations are the same, and these have been presented in the prior sections. The use of an ontological system like the REA ontology is a necessary but not sufficient condition to facilitate effective and efficient inter-organizational systems. Again, intelligent agents and automated intensional reasoning is also required for this to occur. Further, the semantics of the systems must not be obscured by subsequent implementation artifacts.

There are many issues that still need to be resolved, and as such these present many research opportunities. One issue focuses on the scalability of the systems based on the REA ontology. The system presented in this chapter is for a relatively small organization. How this translates into a system for a large multinational firm needs to be explored. Also, while research has been conducted on automated intensional reasoning (Rockwell and McCarthy, 1999), much more is needed. Further, this needs to be extended to the use of intelligent agents and to object-based environments.

Another issue is that of preserving the semantics at an operational level, beyond the level of the database itself. This would allow decision-makers additional insight into the problems and the information available to address the issues that they face. Again, object-based systems seem to provide the most benefit, but additional research is needed.

Regardless of what the research demonstrates, organizations will need to be convinced that they should change the ERP systems that they have acquired. These systems required an immense investment, and in some cases, these systems are still not functioning in an acceptable manner. It is most likely that change will need to be driven by the ERP vendors themselves. They would have a

vested interest in selling upgrades to their systems as long as they can demonstrate some type of advantage for the consumer. This has occurred with a great deal of regularity in the PC market. A significant number of organizations would need to make the change in order for the benefits discussed in this chapter to occur. The first fax machine sold did not provide as much value as the one millionth fax machine sold; the more fax machines are sold, the greater are the opportunities for sending and receiving information. Similarly, the value of the first REA based system for inter-organizational use will be limited (although it will facilitate intra-organization needs), but the more companies that realize the value of these systems and build them, the more value will accrue to the first such system.

CONCLUSION

In this chapter we presented a normative semantic model for designing integrated databases for enterprise information systems. This model was developed by McCarthy (1982) and its expanded form has been proposed as an enterprise ontology by Geerts and McCarthy (2002). This ontology is intended to serve as a foundation for integrated enterprise-wide and inter-organizational systems. To take full advantage of the semantically-rich ontological patterns and templates, the REA ontology must implemented with current advances in artificial intelligence technology and object-oriented database technology. Many of the current problems faced by companies who attempt to install ERP systems and integration tools such as EDI can be minimized by use of common semantic patterns that can be reasoned about by intelligent systems. Companies may integrate their systems without using identical business practices.

Non-accounting researchers have conducted most of the existing research on semantic models, both at the individual level and at the organization level. Because REA originated in an accounting domain, non-accounting researchers have not embraced it (perhaps because of their lack of awareness of the REA ontology). We hope that by making information about this enterprise ontology more available to non-accounting researchers who are interested in semantically modeled information systems we will encourage more interest and participation in REA research.

REFERENCES

Abrial, J. R. (1974). Data semantics. In J. W. Klimbie, & K. L. Koffeman (Eds.), Data Base Management. Amsterdam: North Holland Publishing Company, (pp. 1-60).

Agarwal, R., Sinha, A. P., & Tanniru, M. (1996a). Cognitive fit in requirements modeling: A study of object and process methodologies. Journal of Management Information Systems, 13(2) (Fall), 137-162.

Agarwal, R., Sinha, A. P., & Tanniru, M. (1996b). The role of prior experience and task characteristics in object-oriented modeling: An empirical study. International Journal of Human-Computer Studies, 45, 639-667.

Andros, D., Cherrington, J. O., & Denna, E. L. (1992). Reengineer your accounting the IBM way. The Financial Executive, July/August, (pp. 28-31).

APICS. (1998). Defining enterprise resource planning. http://www.apics.org/OtherServices/articles/defining.htm.

Batini, C., Ceri, S., & Navathe, S. B. (1992). Conceptual Database Design: An Entity Approach. Redwood City, CA: Benjamin Cummings.

Bodart, F., Patel, A., Sim, M., & Weber, R. (2001). Should optional properties be used in conceptual modeling? A theory and three empirical tests. Information Systems Research, 12(4), 384-405.

Cherrington, J. O., Denna, E. L., & Andros, D. P. (1996). Developing an event-based system: The case of IBM's national employee disbursement system. Journal of Information Systems, 10(1), 51-69.

David, J. S. (1995). An empirical analysis of REA accounting systems, productivity, and perceptions of competitive advantage. Unpublished doctoral dissertation, Michigan State University.

David, J. S., Dunn, C. L., & McCarthy, W. E. (1999). Enterprise resource planning systems research: The necessity of explicating and examining patters in symbolic form. Working paper, Arizona State University.

Dunn, C. L., Cherrington, J. O, & Hollander, A. S. (2005). Enterprise Information Systems: A Pattern-based Approach, 3rd edition. New York: McGraw-Hill Irwin.

Dunn, C. L., & Grabski, S. V. (1998). The effect of field independence on conceptual modeling performance. Advances in Accounting Information Systems, 6, 65-77.

Dunn, C. L., & Grabski, S. V. (2000). Perceived semantic expressiveness of accounting systems and task accuracy effects. International Journal of Accounting Information Systems, 1(2), 79-87.

Dunn, C. L., & Grabski, S. V. (2002). Empirical research in semantically modeled accounting systems. In V. Arnold & S. G. Sutton (Eds.), Researching Accounting as an Information Systems Discipline. Sarasota, FL: American Accounting Association (pp. 157-180).

Dunn, C. L., & McCarthy, W. E. (1997). The REA accounting model: Intellectual heritage and prospects for progress. Journal of Information Systems, 11(1), 31- 51.

Dunn, C. L., & McCarthy, W. E. (2000). Symbols used for economic storytelling: A progression from artifactual to more natural accounting systems. Working paper, Michigan State University.

Geerts, G. L., & McCarthy, W. E. (1991). Database accounting systems. IT and Accounting: The Impact of Information Technology. In B. C. Williams & B. J. Spaul (Eds.), London: Chapman & Hall, (pp. 159-183).

Geerts, G. L., & McCarthy, W. E. (1999). An accounting object infrastructure for knowledge-based enterprise models. IEEE Intelligent Systems & Their Applications, (July-August), (pp. 89-94).

Geerts, G. L., & McCarthy, W. E. (2000). Augmented intensional reasoning in knowledge-based accounting systems. Journal of Information Systems, 14(2), 127-150.

Geerts, G. L., & McCarthy, W. E. (2001). Using object templates from the REA accounting model to engineer business processes and tasks. The Review of Business Information Systems, 5(4), 89-108.

Geerts, G. L., & McCarthy, W. E. (2002). An ontological analysis of the primitives of the extended-REA enterprise information architecture. International Journal of Accounting Information Systems, 3, 1-16.

Geerts, G. L., & McCarthy, W. E. (2004). The ontological foundation of REA enterprise information systems. Working paper, Michigan State University.

Gerard, G. J. (2005). The REA pattern, knowledge structures, and conceptual modeling performance. Journal of Information Systems, 19, 57-77.

Gruber, T. (2008). Ontology. In Ling Liu and M. Tamer Özsu (Eds.) Encyclopedia of Database Systems, Springer-Verlag. Found online at http://tomgruber.org/writing/ontology-definition-2007.htm (accessed 10/5/07).

Haugen, R., & McCarthy, W. E. (2000). REA: A semantic model for internet supply chain collaboration. Presented at Business Object Component Design and Implementation Workshop VI: Enter-

prise Application Integration which is part of The ACM Conference on Object-Oriented Programming, Systems, Languages, and Applications, October 15-19, Minneapolis, Minnesota.

Kim, Y. K., & March, S. T. (1995). Comparing data modeling formalisms. Communications of the ACM, 38(6), 103-115.

McCarthy, W. E. (1982). The REA accounting model: A generalized framework for accounting systems in a shared data environment. The Accounting Review, 57(3), 554-578.

Nakamura, H., & Johnson R. E. (1998). Adaptive Framework for the REA Accounting Model, Proceedings of the OOPSLA'98 Business Object Workshop IV, http://jeffsutherland.com/oopsla98/nakamura.html.

O'Leary, D. E. (2004). On the relationship between REA and SAP. International Journal of Accounting Information Systems, 5(1), 65-81.

Rockwell, S. R., & McCarthy, W. E. (1999). REACH: automated database design integrating first-order theories, reconstructive expertise, and implementation heuristics for accounting information systems. International Journal of Intelligent Systems in Accounting, Finance & Management, 8(3), 181-197.

Satoshi, H. (1999). The contents of interviews with the project manager of FDWH in IBM Japan. Proceedings of the 1999 SMAP Workshop, San Diego, CA.

Scheer, A. W. (1998). Business Process Engineering: Reference Models for Industrial Enterprises. Berlin: Springer-Verlag.

Sinha, A. P., & Vessey, I. (1999). An empirical investigation of entity-based and object-oriented data modeling. Proceedings of the Twentieth International Conference on Information Systems. P. De & J. DeGross (Eds.), Charlotte, North Carolina, (pp. 229-244).

Swagerman, D. M., Dogger, N., & Maatman, S. (2000). Electronic markets from a semiotic perspective. Electronic Journal of Organizational Virtualness, 2(2), 22-42.

Vessey, I., & Conger, S. A. (1994). Requirements specification: Learning object, process, and data methodologies. Communications of the ACM, 37(5), 102-113.

Wakayama, T., Kannapan, S., Khoong, C. M., Navathe, S., & Yates, J. (1998). Information and Process Integration in Enterprises: Rethinking Documents. Norwell, MA: Kluwer Academic Publishers.

Watson, E. E., & Schneider, H. (1999). Using ERP systems in education. Communications of the Association for Information Systems, 1(9).

Weber, R. (1986). Data models research in accounting: An evaluation of wholesale distribution software. The Accounting Review, 61(3), 498-518.

Weber, R. (1996). Are attributes entities? A study of database designers' memory structures. Information Systems Research, 7(2), 137-162.

KEY TERMS

Business Process: A term widely used in business to indicate anything from a single activity, such as such as printing a report, to a set of activities, such as an entire transaction cycle; in this paper business process is used as a synonym of transaction cycle. (p. 20)

Enterprise Resource Planning System: An enterprise wide group of software applications centered on an integrated database designed to support a business process view of the organization and to balance the supply and demand for its resources; this software has multiple modules that may include manufacturing, distribution, personnel, payroll, and financials and is considered to

provide the necessary infrastructure for electronic commerce. (p. 2)

Ontologically-Based Information System: An information system that is based upon a particular domain ontology, and the ontology provides the semantics inherent within the system. These systems facilitate organizational productivity and inter-organizational communication. (p. 3)

Process Level Model: A second level model in the REA ontology that documents the semantic components of all the business process events. (p.20)

Resources-Events-Agents (REA) Ontology: A domain ontology that defines constructs common to all enterprises and demonstrates how those constructs may be used to design a semantically modeled enterprise database. (p. 3)

Semantically Modeled Database: A database that is a reflection of the reality of the activities in which an enterprise engages and the resources and people involved in those activities. The semantics are present in the conceptual model, but might not be readily apparent in the implemented database. (p. 3)

Task Level Model: A third level model in the REA ontology is the most detailed level, which specifies all steps necessary for the enterprise to accomplish the business events that were included at the process level. (p. 6)

Value Chain: The interconnection of business processes via resources that flow between them, with value being added to the resources as they flow from one process to the next. (p. 20)

ENDNOTES

1 An alternative semantic model has been presented by Scheer (1998). Additional research is needed comparing these two semantic models with subsequent evaluations of commercially available ERP packages.

2 The REA framework uses the term business process to mean a set of related business events and other activities that are intended to accomplish a strategic objective of an organization. In this view, business processes represent a high level of abstraction. Some non-REA views define business processes as singular activities that are performed within a business. For example, some views consider "process sale order" to be a business process. The REA view considers "sale order" to be a business event that is made up of specific tasks, and it interacts with other business events within the "Sales-Collection" business process.

Chapter XXVII
The Linkcell Construct and Location-Aware Query Processing for Location-Referent Transactions in Mobile Business

James E. Wyse
Memorial University of Newfoundland, Canada

INTRODUCTION

The technologies that enable the transactions and interactions of mobile business are now as ubiquitous as any business-applicable technology that has emerged in recent decades. There is also an exploding base of literature with mobile business as its subject. The variety and volume of literature present a challenge to defining mobile business (*m*-business) in a way that differentiates it from other forms of technology-enabled business activity. For purposes here, *m-business* is held to be an extension of electronic business wherein transactions occur through communication channels that permit a high degree of mobility by at least one of the transactional parties. Within *m*-business, the distinct sub-area of *location-based mobile business* (*l*-business) has recently emerged and is rapidly expanding (Frost & Sullivan, 2006). In *l*-business, the technologies that support *m*-business transactions are extended to incorporate location-aware capabilities. A system is 'location aware' when it senses a transactional party's geographical position and then uses that positional information to perform one or more of the CRUD (create, retrieve, update, delete) functions of data management in support of a mobile user's transactional activities (Butz, Baus, and Kruger, 2000). In their discussion of "location awareness", Yuan and Zhang (2003) suggest that it "is a new dimension for value creation" applicable to an extensive variety of areas in which mobility is a salient characteristic: travel and tourism, commercial transportation, insurance

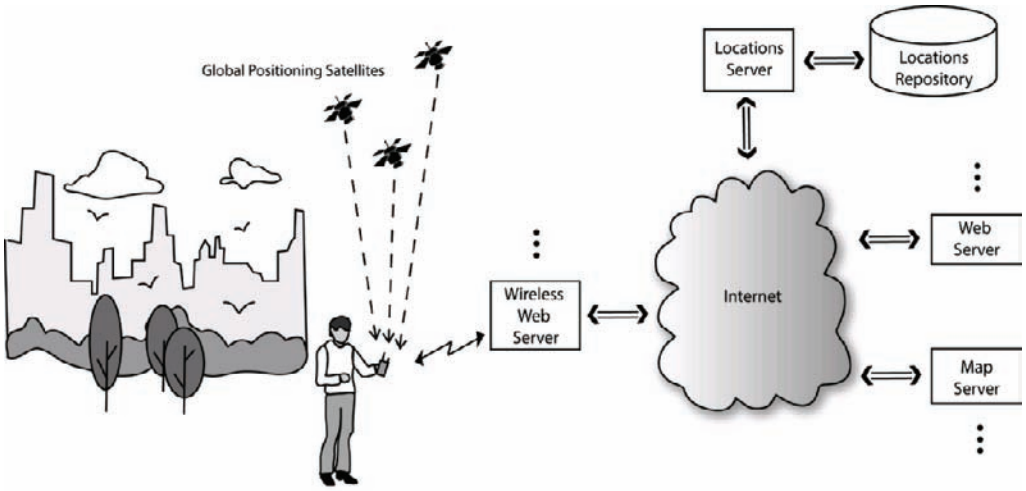
risk/recovery management, emergency response, and many others.

Figure1 depicts a mobile user in an *l*-business context in which a GPS-enabled, Internet-connected, mobile, client platform employs a wireless communication channel to interact with web-based resources to support the user's transactional activities. Here, the provision of *l*-business services relies on an accessible collection of location-qualified business information. This information (contained in the Locations Repository of Figure 1) and its management (performed by the Locations Server of Figure 1) are the focal concerns of this article. In what follows, Wyse's (2003, 2004, 2006, 2008) location-aware linkcell (LAL) repository management method is described and discussed. Selected research results are reviewed that assess the performance and applicability of the method. The article concludes with an identification of trends in *l*-business addressed by the location-aware method. We begin with a discussion of the *problem* of managing data for which the location-aware method is intended as a potential *solution.*

BACKGROUND: THE LOCATION REPOSITORY MANAGEMENT PROBLEM

l-business applications must process transactions in which the proximity of the transactional parties is a material consideration. (Transactions of this type are referred to here as location-referent transactions.) *l*-business applications must also accommodate the location-variant behaviour of the transactional parties. Proximity-related questions such as *Where is the nearest emergency medical facility?* or *Is there a cash machine close by?* or, more generally, *What is my proximity to a goods-providing/service-offering location in a targeted business category?* typify those posed by mobile users when contemplating or concluding transactions in respect of user-sought goods and services. Queries (reflecting such questions) that are formulated by, and issued from, *l*-business applications based on mobile computing platforms require a resolution approach that accommodates not only the static locational attributes associated with the *targeted business category* but also the

Figure 1. Location-based m-business with a Web-based locations repository and location-aware repository management (Adapted from Wyse, 2007. Reproduced with the permission of IEEE).



dynamic locational attributes associated with *proximity.*

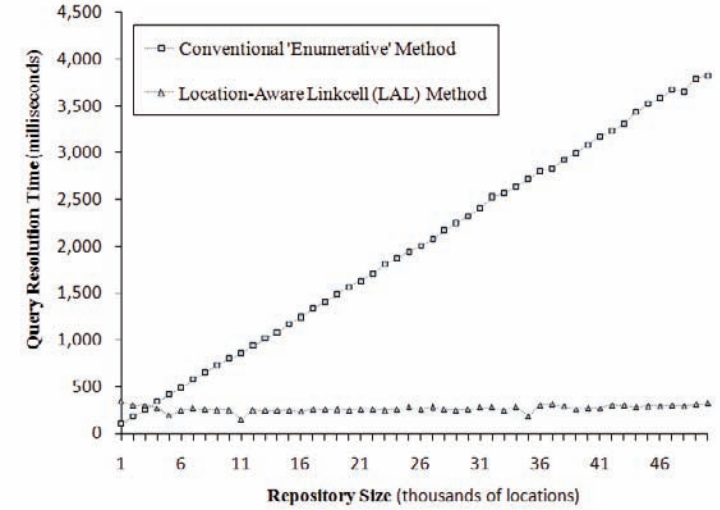
A static locational attribute such as *business category* may be patently resident in a locations repository and thus readily available for retrieval by conventional methods. A dynamic locational attribute such as *proximity* must be derived from the positions of both the mobile user and businesses in the targeted category. Queries bearing criteria relating to both static and dynamic locational attributes may be resolved by conventional query resolution approaches; however, Nievergelt and Widmayer (1999) point out that queries involving "proximity" present a challenge to "conventional database systems" arising from the need to regenerate proximity attributes for all businesses in the targeted category at a rate consonant with the frequency and extent of the mobile user's change in geographical position. Gaede and Gunther's (1998) review of "point access methods" and Chavez et al.'s (2001) examination of "proximity searches" in "metric spaces" represent comprehensive surveys of the methods and techniques whose development has been motivated by the inadequacy of conventional methods of resolving proximity queries. When a repository contains a small number of locations, a proximity query issued in mobile circumstances may be resolved by a conventional 'enumerative' method in a timely manner; however, the query resolution time of such methods increase (i.e., degrade) as the number of locations contained in a repository increases. An examination of 'non-enumerative' techniques such as those surveyed in Gaede and Gunther (1998) and Chavez et al. (2001) suggest that such degradation may be mitigated to varying extents but never eliminated. In other words, no proximity search method has been identified for which query resolution time is independent of repository size.

The upper curves in Figures 2(a) and 2(b) illustrate the degradation in query resolution time for larger repositories that results when a conventional enumerative method is used. Larger repositories are potentially more valuable to mobile users than smaller repositories.[1] However, the degradation in mobile user service levels attributable to increased query resolution times represents an important problem in sustaining the value of large repositories. The lower curves of Figures 2(a) and 2(b) represent the query resolution performance realized by the location-aware linkcell (LAL) retrieval method. That this resolution performance outcome appears to become independent of repository size as locations are added to the repository suggests that the LAL method is a potential solution to the degradation problem associated with repository growth. Note from Figure 2(a) that the LAL method's retrieval performance for a query about the nearest three locations does not dominate the enumerative method for very small repositories. In Figure 2(b) the region of enumerative method dominance is amplified for a query about the nearest thirty locations. However, in both cases retrieval performance eventually becomes substantively independent of repository size as the range of repository sizes is extended. We shall see in what follows that LAL retrieval performance is observed to drift upwards when evaluated using repository sizes that exceed those used to generate the performance profiles in Figure 2. We shall also see how the linkcell 'size' parameter may be manipulated to mitigate the this drift and maintain the substantive independence of the method's retrieval performance profile.

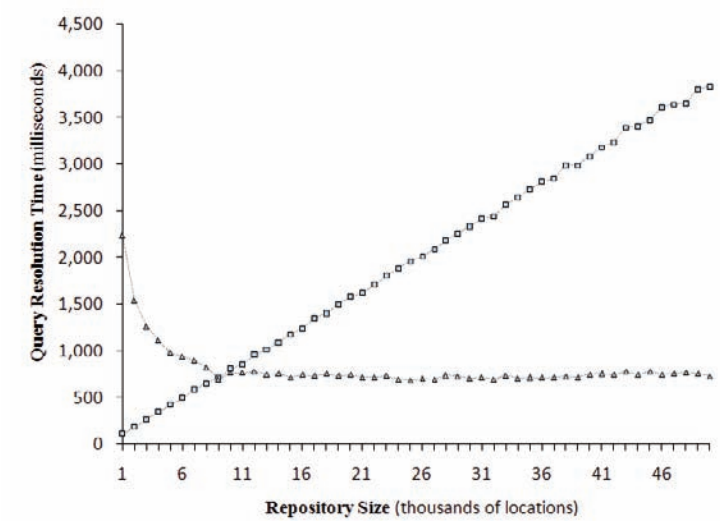
THE LOCATION-AWARE LINKCELL (LAL) METHOD OF REPOSITORY MANAGEMENT: A POTENTIAL SOLUTION

The proximity problem faced by query resolution in *l*-business exhibits characteristics similar to the nearest neighbour (NN) problem in computational geometry. Several NN solution approaches have been developed (Cary, 2001; Lee, 1999; Arnon et al., 1999; along those surveyed in Gaede and

Figure 2. Enumerative and LAL query resolution (Adapted from Wyse, 2003. Reproduced with the permission of Interscience Publishers.)



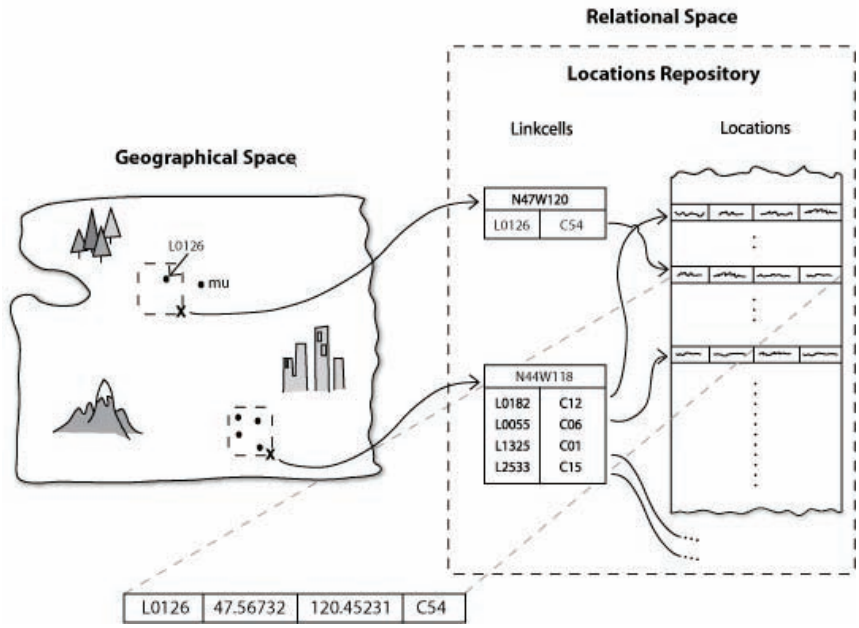
(a) Nearest 3 locations



(b) Nearest 30 locations

Gunther, 1998 and Chavez et al., 2001) but all exhibit behaviour wherein solution times degrade as dataset (i.e., repository) size grows larger. LAL's computational objective is essentially the same as that of NN solution approaches; however, the LAL method does not incorporate the constructs and procedures of computational geometry. Instead, its location-aware "linkcell" approach is based on (1) the relational model of data management (Codd, 1970) and (2) the widely used latitude/longitude convention for designating geographical position. Figure 3 illustrates the role of linkcells in transforming locations in geographical space to tuples and tables in relational space. Linkcells are manifested as relational tables in which the structure of a table's name and the table's content

Figure 3. Linkcell-based transformation of geographical space to relational space (Source: Wyse, 2007. Reproduced with the permission of IEEE)

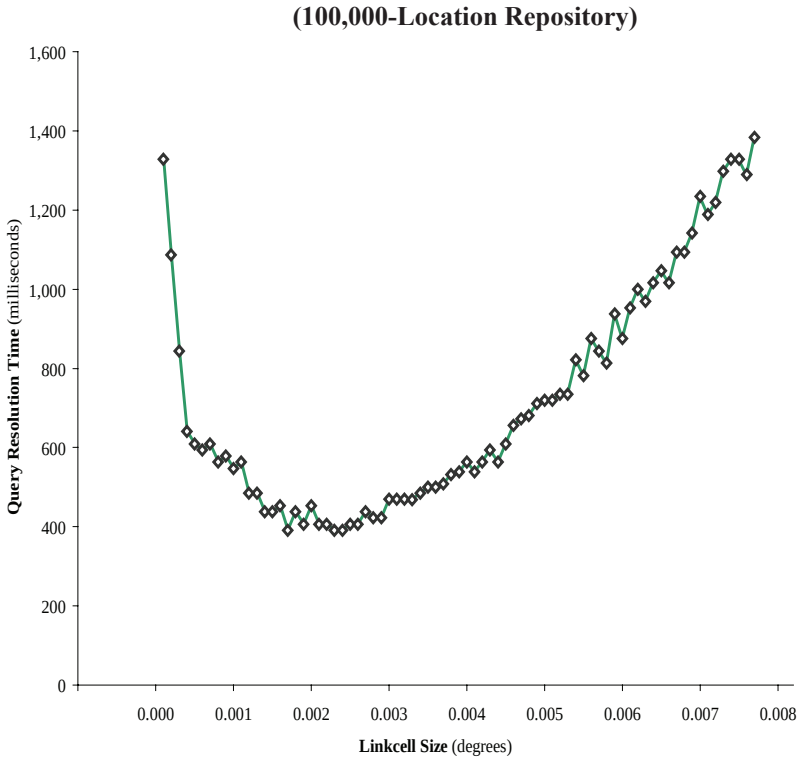


link the locations in a specific geographical area (a *cell*) to a list of business locations in a locations repository. A linkcell's table name (for example N47W120 in Figure 3) is constructed using the location's positional coordinates and a specified linkcell "size", S, corresponding to the geographical area encompassed by a linkcell. As discussed below, variations in S result in substantial variations in query resolution time.

Figure 4 illustrates the effect of variations in linkcell size on query resolution time. For a fixed geographical area, smaller linkcell sizes result in greater numbers of linkcells (S = 0.0005 results in 95,271 linkcells for the 100,000-location repository of Figure 4); however, these linkcells can be examined relatively quickly for locations in a targeted category because they contain, on average, relatively small numbers of locations. This aspect of the LAL method is referred to as the table *name* effect on query resolution performance, or the *n*-effect.[2] Larger linkcell sizes, on the other hand, result in fewer linkcells (S = 0.007 results in 5,250 linkcells) but these linkcells take relatively longer to examine because they contain, on average, greater numbers of locations. This is referred to as the table *content* effect, or the *c*-effect. Figure 4 reflects how the interplay between the *n*-effect and the *c*-effect results in a discernible region where retrieval performance is effectively maximized (i.e., query resolution time is minimized).

Results such as those shown in Figure 2 suggest that the LAL method holds considerable potential for improved transactional support to mobile users operating location-based applications whilst results such as those in Figure 4 suggest that a poor choice for linkcell size, S, could considerably reduce this potential. Recent research (Wyse, 2006, 2008) on LAL-based repository management reports the formulation of a heuristic that may be used to make good S choices. Before the heuristic is presented and discussed, the impact on data management of implementing the LAL method is reviewed. It will be seen that LALrelated benefits come at the cost of extra computational burdens for all four data management functions.

Figure 4. Query resolution performance by Linkcell size



LINKCELL-BASED DATA MANAGEMENT

Any minimally complete scheme of data management must provide functions that CRUD (create, retrieve, update, and delete) the items contained in any collection of application-supporting data. In what follows, the four CRUD functions are discussed in terms of how each function incorporates the LAL method's linkcell construct.

Create Function

When a non-LAL data management approach is taken, a create function inserts a location's tuple {location identifier, vertical geo-positional coordinate, horizontal geo-positional coordinate, location category code} in the table of locations (seen in the right half of the Locations Repository in Figure 3). When the LAL method is used, tuple insertion in the table of locations also takes place; however, the LAL method further requires that another tuple {location identifier, location category code} be placed in a linkcell table. As noted above, the name of a linkcell table is derived from a location's geo-positional coordinates and the value, S, that has been selected for linkcell size. If the linkcell table already exists then the tuple is added to it and the 'LAL-ized' create function has completed its work. When the table does not exist, then it is created and the tuple becomes the table's initial entry.

S plays a central role in linkcell table creation. As seen in Figure 3, when S is 1.0, the table's name is derived from the 'whole degree' portion of the location's coordinates. With respect to linkcell N47W120 in Figure 3, S = 1.0 implies that the table contains tuples for any location with a vertical coordinate in the interval [47.00000, 47.99999) and a horizontal coordinate in [120.00000,

120.99999). S may, and generally will, correspond to 'fractional degrees'. For example, S may be set to 0.1. In this case the tuple {L0126, C54} would now be an entry in N475W1204 along with any other location having a vertical coordinate in [47.50000, 47.59999) and a horizontal coordinate in [120.40000, 120.49999)[3]. For any given set of locations, smaller S values will generally result in greater numbers of linkcell tables than larger S values. Note that variations in S have no effect on a business's location in geographical space; however, a variation in S will, in general, result in a redistribution of the locations throughout linkcell tables in relational space.

Delete Function

Whenever a location is removed from the repository's table of locations, it must also be removed from the linkcell in which it currently resides. The only attribute required for deletion is the location identifier. Upon receipt of a location identifier, the location's coordinates may be retrieved and the entire tuple removed from the repository's table of locations. Using the linkcell size, S, a linkcell name is derived from the location's coordinates. The location's tuple is then removed from the linkcell that bears the derived name. If the location is the only remaining entry in the linkcell, then the linkcell itself is removed from the repository.

Update Function

Changes to any of the attributes identified above for a location in the repository will always involve the linkcell with which the location is associated. Three different update procedures are required: one for changing the location identifier (denoted below as UF1), another for changes to positional coordinates (UF2), and a third for changing a location's category code (UF3). UF1: When the location's identifier is changed in the repository's table of locations, it must also be changed in the corresponding linkcell. The linkcell is identified based on S and the location's coordinates. UF2: When a location's coordinate is to be changed, a *candidate* linkcell name is derived using S and the *new* coordinate value; also, a *current* linkcell name is derived using S and the *previous* coordinate value. When the candidate name is the same as the current name then the location remains in the same linkcell and, consequently, no further action is needed (beyond changing the coordinate values in the table of locations). However, if the candidate name differs from the current name, then the location must 'change' linkcells. This is accomplished by a 'delete' action involving the location's identifier and the current linkcell name followed by a 'create' action based on the candidate linkcell name. UF3: When a location's category code is changed, a linkcell name is generated and the location code is changed in both the linkcell and the repository's table of locations.

Retrieval Function

The LAL retrieval function is also burdened with extra computational work. However, unlike the create, delete, and update functions whose linkcell-related burdens will always slow them down, the retrieval function may yield substantial improvements in operational performance.

An examination of the retrieval process reveals how linkcells are the key to the method's superior performance. Referring to Figure 3, imagine that a mobile user issues a query for the nearest emergency medical facility ('medfac' – category code C54) from location **mu.** Upon receiving a query bearing positional data (47.523°N, 119.137°W) for **mu**, a LAL-managed repository server transforms the user's location into a linkcell name, N47W119 in this case. This name references the repository's relational space at a point corresponding to the mobile user's proximity in geographical space. The LAL method then proceeds to generate linkcell names in a clockwise, outward-spiraling, pattern centered on N47W119: {N48W119, N48W118,

N47W118, N46W118, N46W119, N46W120, N47W120, … }[4]. As each table name is generated, the repository is checked for its existence. Whenever a linkcell is present in the repository, the table is queried for the targeted category (C54 in this case). Areas corresponding to N48W119 through to N46W120 are readily dismissed because linkcells with these names are not present in the repository. Query resolution is achieved when linkcell N47W120 is encountered, since it is not only present in the repository but it also contains a location in the sought-after category (C54). Note that had N47W120 (containing the user-sought location) *not* been encountered, the method would have continued to pivot around the linkcell N47W119 (containing the user's location) until either (1) a location of the type sought by the user is found or (2) the geographical limits of the repository are encountered.

The generation and examination of linkcell names in patterns that radiate from a mobile user's position permit the method to exclude substantial numbers of locations from explicit consideration as locations in the mobile user's proximity. Exclusionary actions take place in two ways: (1) by initiating an examination of the repository's locations in close proximity to a user's position, and (2) by eliminating areas between the user's position and the targeted location(s) on the basis of the non-existence of linkcells. The frequency of exclusionary actions, and consequently the speed of retrieval, is significantly affected by the linkcell size, S; thus, determining a performance maximizing value of S is an important repository management task.

LINKCELL SIZE SELECTION

Determining an appropriate linkcell size may be done using brute force approaches wherein query resolution performance is examined across a range of S values with respect to a given set of repository characteristics. Figure 4 shows the brute force results for one specific combination of repository characteristics. A discernible linkcell size interval is revealed wherein query resolution performance is maximized. Since 'optimal' linkcell sizes vary across repository characteristics, brute force determination of S will be burdensome to the point of being impractical for operational circumstances in which business locations of differing characteristics are continually being added to, and removed from, a locations repository. This impracticality motivated the search for a method of linkcell size determination "that is more computationally efficient and managerially useable than brute force identification methods" (Wyse, 2006).

This lead to the formulation of a linkcell size determination method whereby, S, is determined with respect to the probability that a linkcell contains a location in the category targeted by a mobile user. Specifically, the probability, $P_{TC}(S)$, that a linkcell contains a location in the targeted category, TC, is given by:

$$P_{TC}(S) = 1 - \left(1 - {}^{n_{TC}}/_{N}\right)^{N/C_S}$$

where n_{TC} is the number of locations in the repository with category code, TC; N is the total number of locations contained in the repository; N/C_S is the mean number of entries per linkcell; and C_S, the maximum number of linkcells of size, S, created from the repository's N locations, is given by:

$$C_S = \left(\left\lceil {}^{NL}/_{S}\right\rceil - \left[{}^{SL}/_{S}\right] + 1\right)\left(\left\lceil {}^{WL}/_{S}\right\rceil - \left[{}^{EL}/_{S}\right] + 1\right)$$

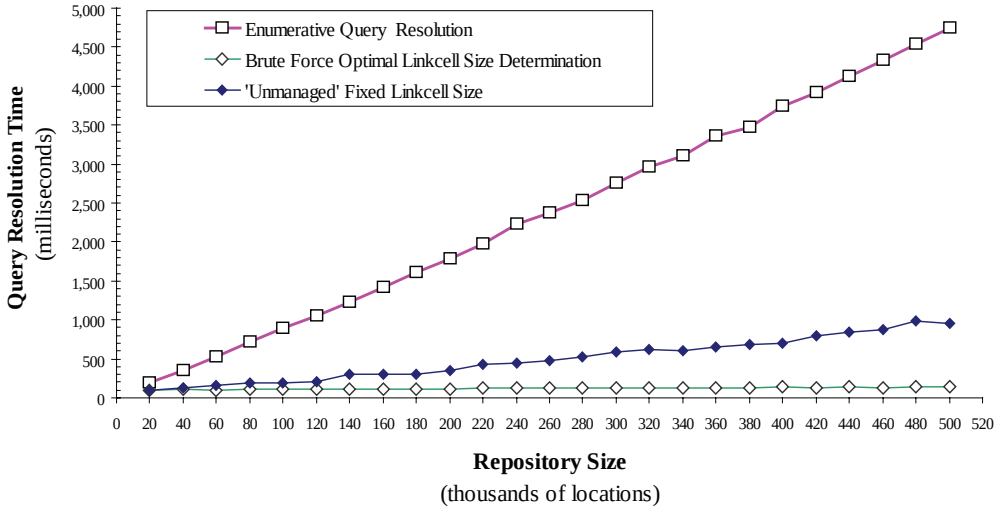
where NL and SL represent the northern and southern limits, respectively, of the latitudinal extent of the geographical area covered by the repository's locations, and WL and EL represent the western and eastern limits, respectively, of the area's longitudinal extent. The first factor for C_S represents the number of S-sized intervals along the vertical extent of the geographical area covered by a repository's locations while the second fac-

tor represents the number of such intervals with respect to the repository's horizontal extent.

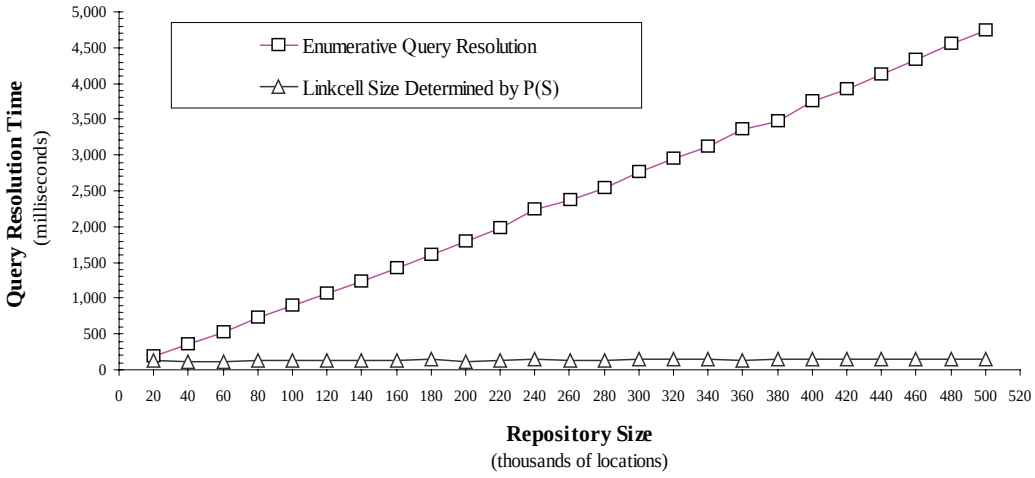
Research has assessed the capability of $P_{TC}(S)$ to identify an optimal linkcell size across various repository characteristics. One study (Wyse, 2008) examined repositories ranging in size from 1,000 to 50,000 locations for a fixed set of category codes within a fixed geographical area. A second study (Wyse, 2006) extended the range of examined repository sizes to 500,000 locations, varied geographical coverage over 24 areas of increasing size, and included 25 differing sets of category codes. In both studies maximum query resolution performance was observed in a region of the performance profiles where $P_{TC}(S) \approx 0.6$.

Figure 5 shows selected results from the second study. Figure 5(a) provides three curves: (1) the uppermost curve is the result of using an enumerative method of query resolution; (2) the bottom curve plots query resolution performance for optimal linkcell size determined by brute force; and (3) the middle curve represents the results

Figure 5. Query resolution performance by repository size (Source: Adapted from Wyse, 2006)



(a)



(b)

for a situation in which linkcell size remains fixed ("unmanaged") as the repository is increased in size. Figure 5(b) provides two curves: the uppermost curve is (again) the result of an enumerative resolution method whilst the bottom curve is the "managed" result of selecting a linkcell size at successive repository sizes using $P_{TC}(S) = 0.6$. A comparison of the bottom curves in Figures 5(a) and 5(b) indicates that setting $P_{TC}(S) = 0.6$ yields a query resolution profile that tracks the brute force profile, a result favouring the usefulness of $P_{TC}(S)$ in determining appropriate linkcell sizes for repositories of differing characteristics.

FUTURE TRENDS

Two significant trends in mobile business provide a basis of relevancy for the location-aware approach to data management: (1) a growing interest in 'context awareness' in mobile applications (Panagiotakis et al., 2003; Rupnik et al., 2004; Turel, 2006; Syukur and Loke, 2006) and (2) a trend toward 'thin client' mobile computing platforms in which the processing work required by *l*-business applications, particularly access to large-scale databases, should be performed to the greatest extent possible by server-based functionality (Siau, Lim, and Shen, 2001; Siau and Shen, 2003). The 'context-aware' trend reflects the desirability of automatically detecting and accommodating the preferences and circumstances of mobile users. Although location-awareness would be but a special case of context awareness, it is one of critical importance to mobile consumers engaged in location-referent transactions. The 'thin-client' trend suggests that server-based functionality such as LAL data management will become increasingly more important in the provision of transactional support to mobile consumers as *l*-business operations seek to increase the value of their repositories to mobile consumers by including more business locations, by providing greater business category variety, and by extending geographical coverage.

CONCLUDING REMARKS

Enabling the location-referent transactions of mobile consumers presents an important data management challenge to the operations managers of location-based mobile systems. The provision of transactional support must incorporate, in some form, a locations repository as well as the means by which its content is managed. When repository sizes are small, repositories may be effectively managed using conventional methods; however, as repository size is increased conventional methods yield an increasing degradation in the service level realized by mobile consumers. An alternative to conventional methods, the location-aware linkcell method displays the potential to significantly ameliorate service level degradation. Although the method is more burdensome than conventional methods in certain data management respects, assessments of its query resolution performance indicate its potential as a useful *l*-business data management tool.

REFERENCES

Arnon, A., Efrat, A., Indyk, P., and Samet, H. (1999). Efficient regular data structures and algorithms for location and proximity problems. *Proceedings of the 40th Annual Symposium on Foundations of Computer Science*. New York (Oct 17-19), 160-170.

Butz, A., Baus, J., & Kruger, A. (2000). *Different views on location awareness.* Accessed November 10, 2006 at http://w5.cs.uni-sb.de/~butz/publications/papers/awareness.pdf

Cary, M. (2001). Towards optimal ε-approximate nearest neighbor algorithms. *Journal of Algorithms, 41*(2), 417-428.

Chavez, E., Navarro, G., Baeza-Yates, R., & Marroquin, J. (2001). Searching in metric spaces. *ACM Computing Surveys, 33*(3), 273-321.

Codd, E. (1970). A relational model for large shared data banks. *Communications of the ACM, 13*(6), 377-387.

Frost & Sullivan. (2006). Location-based services offer high revenue potential in competitive mobile communications market. *Directions Magazine*. (January 17): Accessed November 10, 2006 at http://www.directionsmag.com/press.releases/index.php?duty=Show&id=13403&trv=1

Gaede, V., & Gunther, O. (1998). Multidimensional Access Methods . *ACM Computing Surveys, 30*(2), 170-231.

Lee, D. (1999). Computational geometry II. In M. Atallah (Ed.), *Algorithms and Theory of Computation Handbook* (pp. 20-1 to 20-31). CRC Press.

Nievergelt, J., & Widmayer, P. (1997). Spatial data structures: concepts and design choices. M. van Kreveld, J. T. NievergeltRoos, & P. Widmayer (Eds.), *Algorithmic Foundations of Geographic Information Systems* (pp. 153-197). Berlin: Springer Verlag.

Panagiotakis, S., Koutsopoulou, M., Alonistioti, A., Houssos, N., Gazis, V., & Merakos, L. (2003). An advanced service provision framework for reconfigurable mobile networks. *International Journal of Mobile Communications, 1*(4), 425-438.

Porter, M. (2001). Strategy and the internet. *Harvard Business Review* (March 2001): (pp. 63-78).

Rupnik, R., Krisper, M., & Bajec, M. (2004). A new application model for mobile technologies. *International Journal of Information Technology and Management, 3*(2/3/4), 282-291.

Siau, K., Lim, E., & Shen, Z. (2001). Mobile commerce: Promises, challenges, and research agenda. *Journal of Database Management, 12*(3), 4-13.

Siau, K., & Shen, Z. (2003). Mobile communications and mobile services. *International Journal of Mobile Communications, 1*(1/2), 3-14.

Syukur, E., & Loke, S. (2006). Implementing context-aware regulation of context-aware mobile services in pervasive computing environments. *International Journal of Web and Grid Services, 2*(3), 260-305.

Turel, O. (2006). Contextual effects on the usability dimensions of mobile value-added services: A conceptual framework. *International Journal of Mobile Communications, 4*(3), 309-332.

Yuan, Y., & Zhang, J. J. (2003). Towards an appropriate business model for m-commerce. *International Journal of Mobile Communications, 1*(1/2), 35-56.

Wyse, J. E. (2003). Supporting m-commerce transactions incorporating locational attributes: An evaluation of the comparative performance of a location-aware method of locations repository management. *International Journal of Mobile Communications, 1*(1/2), 119-147.

Wyse, J. E. (2004). *System and Method for Retrieving Location-Qualified Site Data.* Patent Number 6,792,421, United States Patent and Trademark Office. Patent Issued: September 14.

Wyse, J. E. (2006). Location-aware query resolution for location-based mobile commerce: performance evaluation and optimization. *Journal of Database Management, 17*(3), 41-65.

Wyse, J. E. (2007). Applying location-aware linkcell-based data management to context-aware mobile business services. *Proceedings of the Sixth International Conference on Mobile Business*, Toronto, Ontario, Canada, (July 9-11): 8 pages.

Wyse, J. E. (2008). Optimizing server performance for location-aware applications in mobile commerce: The repository manager's formula. *International Journal of Wireless and Mobile Computing.* (forthcoming).

KEY TERMS

c-Effect (Content Effect): An increase in query resolution time attributable to an increase in the average number of locations contained in a linkcell for a repository of fixed characteristics.

Dynamic Locational Attribute: A proximity attribute (e.g., direction, distance) associated with a business location that must be regenerated at a rate consonant with the frequency and extent of a mobile user's change in geographical position.

Linkcell: A relational table in which the structure of the table's name and the table's content *link* the geographical positions of business locations in a specified geographical area (a *cell*) to their respective entries in the table of locations in a locations repository.

Linkcell Size: A value for the size of the geographical area covered by a linkcell. For example, a linkcell size of 1.0 refers to a geographical area 1º of latitude by 1º of longitude.

Location-Awareness: The capability to (1) sense a mobile user's geographical position and (2) use such positional information to perform one or more of the CRUD functions of data management in support of the mobile user's transactional activities.

Location-Based Mobile Business (l-Business): An area of electronic business in which (1) transactions occur through communication channels that permit a high degree of mobility by at least one of the transactional parties and (2) the systems processing such transactions have location-aware capabilities.

Location-Referent Transaction: A transaction in which the proximity of the transactional parties is a material transactional consideration.

n-Effect (Name Effect): A decrease in query resolution time attributable to an increase in the linkcell size implicitly reflected in linkcell names.

Static Locational Attribute: A locational attribute patently resident in a locations repository and invariant with respect to changes in a mobile user's geographical position.

ENDNOTES

1 Porter's (2001) "network effect" implies that repositories containing greater numbers of business locations and attracting greater numbers of mobile consumers are potentially more valuable than those containing fewer businesses and attracting fewer mobile consumers.

2 For very small values of S relative to a given geographical area, many of the linkcell names generated to retrieve a location in a mobile user's proximity will correspond to non-existent linkcell table names. Although their non-existence allows such linkcells to be quickly dismissed, their numbers may be high enough at small S values to become a significant component of the *n*-effect.

3 In this particular embodiment of the LAL method, the linkcell table name is set to southeast corner of the geographical area covered by the linkcell. This is an arbitrary choice; other embodiments of the method could use some other point on, or within, the linkcell's boundaries.

4 Other linkcell name generation patterns could be used.

Chapter XXVIII
Caching, Hoarding, and Replication in Client/Server Information Systems with Mobile Clients

Hagen Höpfner
International University in Germany, Germany

INTRODUCTION

Redundant data management is a must in client server information systems with mobile clients. Based on the level of autonomy of mobile devices/users techniques for handling such data can be divided into caching, hoarding, and replication. These three terms are often used incorrectly in the literature. To our knowledge the exact definition of the terms has never been published in an international book or journal. We fill this gap with this article. We furthermore explain the terms cache replacement, cache invalidation, cache maintenance, automated hoarding, and synchronization of replicated data.

BACKGROUND

Many mobile information systems are client/server information systems with mobile, lightweight, portable clients that access data and information over wireless links. This scenario implies various restrictions that must be considered:

- Mobile clients must be mobile and therefore need to use rechargeable batteries for power supply.
- The availability of wireless links is uncertain and unstable, very often slow, and/or expensive.
- Wireless data transmission is energy intensive and reduces the uptime of mobile clients.

The only possibility to react to these limitations is to store and reuse data on the mobile device. Unfortunately, redundant data management also implies a high inconsistency risk. Updates to local stored data items and updates to server side data must be harmonized.

MAIN THRUST: MANAGING REDUNDANT DATA

The literature describes three major classes of approaches for managing redundant data in a client server manner if clients are mobile: caching, hoarding, and replication. The following three general questions help to understand the differences: Is the data on the mobile device read only or can the user update it? Who decides about what data is stored on the mobile device - the user or the system? Is the decision for storing data on the device dynamic or static? Figure 1 illustrates various important aspects in which the three classes differ from each other:

- The grade of possible data manipulations: *Are updates allowed on the mobile device?*
- The possibility to work offline: *Is all required data guaranteed to be on the mobile device?*
- The potential dynamic of data: *How often does the data, which are on the mobile device, change?*
- The required resources: *Which kind of software is required for a certain technique?*
- The influence on local data: *Is the user able to specify the data that he or she needs on the mobile device?*

In the following we discuss these aspects per technique in more detail.

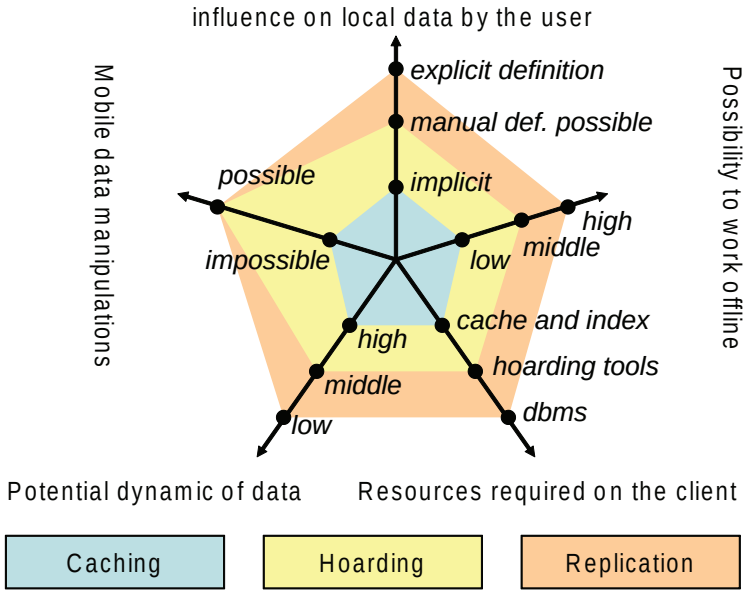
Caching

The easiest way to provide a mobile user with data locally is to cache data once it has been received. So, data is requested implicitly. Web and WAP browsers, for example, automatically cache WML or HTML pages and pictures. We know caching approaches from network based desktop applications as well as from database based information systems. The research on caching in the context of mobile information system focuses on semantic caching instead of caching pages/blocks or objects. Thus, query results are stored as they are but indexed using the corresponding query (Keller and Basu, 1996, Lee et al., 1999, Ren and Dunham, 1999). Because of this, it is possible to analyze whether new queries can be answered without communicating with the server. To a certain degree, it is also possible to recombine cached data with additional data to answer a query (Godfrey and Gryz, 1999, Höpfner and Sattler, 2003, Ren and Dunham, 2003).

Unfortunately, the algorithms are complex and suffer from strict limitations of the underlying query containment problem (Chandra and Merlin, 1977, Chekuri and Rajaraman, 1997, Klug, 1988, Guo et al., 1996, v. d. Meyden, 1997). The availability of data in the cache cannot be guaranteed as it depends on the used replacement strategy. If the cache is full, these algorithms decide on removing (hopefully) data no longer required. However, that decision is typically based on the time a data item is already in the cache and/or on the number of accesses of this data item in the cache might be incorrect. Semantic caches tend to use semantic correlations between different cached items as replacement criteria (Dar et al., 1996).

Caches are primarily used for reducing costs and delays of data transmissions. However, they do not provide full offline work with the information system. On the other hand caches are easy realizable and do not require additional functionalities such as synchronization or data mining that are

Figure 1. Classification of caching, hoarding, and replication (based on (Höpfner et al., 2005))



necessary for replication or hoarding. Nevertheless, semantic caches need an index structures and algorithms for retrieving and recombining cached data. Another aspect of caching is that local data modifications are normally not supported and therefore used in read only scenarios.

Replication

Replication approaches enable the possibility of working offline with the data stored on the mobile device. The user explicitly copies the required data on the mobile device; she defines the replicas. In fact, nearly each database management system vendor offers mobile DBMS that can be used in combination with a back end DBMS for replication purposes (e.g. Oracle Lite (Oracle, 2004), IBM DB2 Everyplace (IBM, 2004), Microsoft SQL Server CE (Microsoft, 2001), or Sybase Adaptive Server Anywhere (Sybase, 2004)). The replication process depends on the underlying system architecture that could be either a simple client/server approach or a middle-ware solution. Client/server architectures offer direct access from the client to the server database. Middleware approaches use additional components (the middle-ware) that manage the communication between the server database(s) and the mobile client. Replicas are defined similarly to database (materialized) views. Hence, in order to permit updates to the replicated data, one has to consider the same query limitations that hold for updateable views. Therefore, selection and projection operations are allowed but join operations are mostly forbidden. Along with the content definition, the user can also specify as to when the system shall synchronize updates. If the mobile device runs out of memory, the user has to free it manually by deleting replicated data.

Hoarding

The class of hoarding approaches covers aspects of caching and of replication. Similar to caches, it uses a replacement strategy and implicitly stores data on the mobile device. In addition algorithms analyze the usage of data. So, necessary data that was not explicitly requested but

is needed for working offline can be found and cached. Furthermore, some hoarding systems also support manual specification of data that needs to be cached. Such techniques overlap with the replica specification. However, hoarding can be found mainly in the form of hoarding file systems such as CODA (Satyanarayanan et al., 1990) and its Seer extension (Kuenning and Popek, 1997) that support *automatic hoarding.* This file system analyzes data accesses. If, e.g., the user compiles a C-program, the source code as well as the involved header files is cached. From an information systems point of view, hoarding can be realized via query rewriting such as harming select conditions or extending the projection list of database queries. Less restrictive select conditions increase the cardinality of the query result (vertical extension). Hence, additional tuples that might be usable for upcoming queries are copied to the client. Adding additional attribute names to the projection list or even removing the projection operation completely extends the query result in a horizontal way. The additionally hoarded attribute values might be usable for answering upcoming queries. However, both approaches require a local filtering of the query result as they change the semantics of the current query. Furthermore, additional information might be used as hoarding criteria. For example, (Peissig, 2004) or (Kubach and Rothermel, 2001) propose systems that cache data relative to the current or a predicted future location.

Data Coherency

Caching, hoarding, and replication create redundancies. Data items appear more then once in the information system. Therefore, changes can potentially lead to inconsistencies if copies of the same data item are updated independently. Inconsistencies are possible even if no client updates are allowed. If a client caches data and the server data becomes updated, the cache becomes outdated.

The easiest possibility of handling this problem is to invalidate the data on the mobile client. Unfortunately, this means that local data cannot be used any more even if only one single datum has been changed. More enhanced incremental *cache invalidation* strategies (Cai et al., 1997) invalidate not whole caches but only parts. Another problem results from the weak connectivity of mobile clients. It cannot be guaranteed that all clients that must be informed about an update are available when the invalidation report is sent out. There exist certain approaches that combine a repeated reporting and time stamp. A taxonomy of the alternatives is included in (Cai et al., 1997). However, they can only exacerbate the problem and not solve it. Furthermore, in general the server must decide which client needs to be informed about an update (Höpfner, 2006).

In contrast to cache invalidation, *cache maintenance* techniques try to patch cached data in order to make them coherent. Since cached data can be considered to be materialized views, i.e., the cache maintenance problem is the same as the view maintenance problem (Gupta et al., 1993). Unfortunately, the incremental view update problem in general is not decidable. Reducing it to conjunctive select-predicate-join views only makes it easier but practically irrelevant. However, commercial database management systems tend to reduce the capabilities of the query sublanguage that can be used for the definition of update-able views.

Since replication strategies allow for updating data on the mobile client, one must be able to integrate these updates into the server database. This process is called *synchronization*. The literature describes *data centric* and *transaction centric* synchronization approaches. Data centric synchronization, like Palms HotSync, uses status bits to monitor changes, whereas transaction centric synchronization (Gray et al., 1996) supports tentative transactions. In other words, all client side modifications result in a tentatively committed new version of data items. The synchronization

then tries to perform the tentative transaction on the server. Not conflicting executions result in a permanent commit, conflicts in a rollback. Conflicts might, e.g., appear if the same data item was modified on the client and on the server in parallel. Data centric approaches use simple rules to solve conflicts. Either the server data or the client data are overwritten.

FUTURE TRENDS

There are many issues besides data coherency or costs for data transmissions that will arise as devices become smaller and more ubiquitous. The most important limitation is the battery supply. As shown in this article, caching, hoarding, and replication help to reduce energy intensive data transmissions. Other resources, besides the resources energy and network communication, are, e.g., the CPU or memory. Data might be transmitted in a compressed manner but this means that the CPU resource is used more extensively. A future trend is to find a proper strategy for balancing all resources of mobile devices (Höpfner and Bunse, 2007). Another future trend was mentioned in this article already: context awareness. The idea is to use knowledge about the users' behavior, his or her current location or about her equipment in order to ease the data processing. Context aware hoarding, caching, and replication are currently discussed and initial results exist. However, all existing approaches consider context information in addition to the base data. Only a few (Höpfner, 2005) researchers have tried to integrate context processing and data processing.

CONCLUSION

We have presented caching, hoarding, and replication as techniques for processing data in mobile information systems. We have clearly differentiated the approaches of each other based on their key characteristics. To our knowledge, this classification is new and aims to harmonize the usage of the terms. Furthermore, we have presented and explained the techniques that must be used in order to keep redundant data coherent. Caching, hoarding and replication techniques are also known from wired networks where they are used in order to increase the availability of data or to decrease the response time of the system. However, in mobile information systems they are mandatory. Neither faster wireless networks nor cheaper data communication contracts solve all problems that are handled by caching, hoarding, and replication strategies.

REFERENCES

Chandra, A. K., & Merlin, P. M. (1977). Optimal implementation of conjunctive queries in relational databases. *In Proceedings of the ninth annual ACM symposium on Theory of computing*, Boulder, Colorado, United States, (pp. 77–90), New York, NY, USA.

Cai, J., Tan, K.-L., & Ooi, B. C. (1997). On incremental cache coherency schemes in mobile computing environments. In A. Gray, & P.-Å Larson (Eds.), *Proceedings of the Thirteenth International Conference on Data Engineering*, April 7-11, 1997 Birmingham U.K, S. 114–123. IEEE Computer Society, Los Altos, CA, USA

Chekuri, C., & Rajaraman, A. (1997). Conjunctive Query Containment Revisited. In F. N. Afrati & P. G. Kolaitis, (Eds.), *Proceedings of the 6th International Conference on Database Theory - ICDT '97*, Delphi, Greece, January 8-10, 1997, volume 1186 of Lecture Notes in Computer Science, (pp. 56–70), Heidelberg. Springer-Verlag.

Dar, S., Franklin, M. J., Jonsson, B., Srivastava, D., & Tan, M. (1996). Semantic Data Caching and Replacement. In T. M.Vijayaraman, A. P. Buchmann, C. Mohan, & N. L. Sarda (Eds.),

Proceedings of 22th International Conference on Very Large Data Bases (VLDB'96), (pp. 330–341), San Francisco, CA, USA. Morgan Kaufmann.

Godfrey, P., & Gryz, J. (1999). Answering queries by semantic caches. In T. Bench- Capon, G. Soda, and A. M. Tjoa (Eds.), *Database and Expert Systems Applications: Proceedings of the 10th International Conference, DEXA'99*, Florence, Italy, August/September 1999, volume 1677 of Lecture Notes in Computer Science, (pp. 485–498), Heidelberg. Springer-Verlag.

Gray, J., Helland, P., O'Neil, P. E., & Shasha, D. (1996). The Dangers of Replication and a Solution. In H. V. Jagadish, & I. S. Mumick (Eds.), *Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data*, Montreal, Quebec, Canada, June 4-6, 1996, *SIGMOD Record, 25*(2),173–182. ACM Press, New York, NY, USA, Juni 1996.

Guo, S., Sun, W., & Weiss, M. A. (1996). Solving satisfiability and implication problems in database systems. *ACM Transactions on Database Systems (TODS), 21*(2), 270–293.

Gupta, A., Mumick, I. S., & Subrahmanian, V. S. (1993). Maintaining views incrementally. *ACM SIGMOD Record, 22*(2), 157–166

Höpfner, H. (2005). Towards update relevance checks in a context aware mobile information system. In A. B. Cremers, R. Manthey, P. Martini, & V. Steinhage (Eds.), *Proceedings of the 35rd annual conference of the German computer society*, number P-68 in Lecture Notes in Informatics (LNI) - Proceedings, (pp. 553–557), Bonn, Germany. Gesellschaft für Informatik, Köllen Druck+Verlag GmbH.

Höpfner, H. (2006). Update Relevance under the Multiset Semantics of RDBMS. In T. Kirste, B. König-Ries, K. Pousttchi, K. Turowski (Eds.), *Proceedings of the 1st conference on mobility and mobile information systems*, number P-76 in Lecture Notes in Informatics (LNI) - Proceedings, (pp. 33–44), Bonn, Germany. Gesellschaft für Informatik, Köllen Druck+Verlag GmbH.

Höpfner, H., & Bunse, C. (2007). Resource Substitution for the Realization of Mobile Information Systems. *In proceedings of the 2nd international conference on software and data technologies, Volume: software engineering,* July 22-25, 2007, Barcelona Spain, pages 283-289, INSTICC Press

Höpfner, H., & Sattler, K.-U. (2003). Towards Trie-Based Query Caching in Mobile DBS. In B. König-Ries, M. Klein, & P. Obreiter (Eds.), *Post-Proceedings of the Workshop Scalability, Persistence, Transactions - Database Mechanisms for Mobile Applications, number P-43 in Lecture Notes in Informatics (LNI) - Proceedings*, (pp. 106–121), Bonn, Germany. Gesellschaft für Informatik, Köllen Druck+Verlag GmbH.

Höpfner, H., Türker, C., & König-Ries, B. (2005). *Mobile Datenbanken und Informationssysteme — Konzepte und Techniken*. dpunkt.verlag, Heidelberg, Germany. in German.

IBM (2004). *IBM DB2 Everyplace Application and Development Guide Version 8.2*. IBM Corporation.

Keller, A. M., & Basu, J. (1996). A predicate-based caching scheme for client-server database architectures. *The VLDB Journal, 5*(1), 35–47.

Kubach, U., & Rothermel, K. (2001). Exploiting location information for infostation-based hoarding. In C. Rose, (Eds.), *Proceedings of the 7th Annual International Conference on Mobile Computing and Networking*, July 16-21, 2001, Rome, Italy Rome, Italy, (pp. 15–27), New York, NY, USA. ACM Press.

Kuenning, G. H., & Popek, G. J. (1997). Automated Hoarding for Mobile Computers. In W. M. Waite (Eds.), *Proceedings of the 16th ACM Symposium on Operating Systems Principles*, (pp. 264–275), New York, NY, USA. ACM Press.

Lee, K. C. K., Leong, H. V., & Si, A. (1999). Semantic query caching in a mobile environment. *ACM SIGMOBILE Mobile Computing and Communications Review, 3*(2), 28–36.

Microsoft (2001). http://msdn.microsoft.com/library/. Microsoft Corporation.

Oracle (2004). *Oracle Database Lite, Administration and Deployment Guide 10g (10.0.0).* Oracle Corporation.

Peissig, J. (2004). guidePort – An Information and Guidance System. In K. Kyamakya, (Ed.), *Proceedings of 1st Workshop on Positioning, Navigation and Communication 2004 (WPNC 04),* University of Hanover, Germany, March 26, 2004, number 0.1 in Hannoversche Beitr¨age zur Nachrichtentechnik, (pp. 1–17), Aachen. NICCIMON, IEEE, VDI, Shaker Verlag GmbH.

Ren, Q., & Dunham, M. H. (1999). Using clustering for effective management of a semantic cache in mobile computing. In S. Banerjee, P. K. Chrysanthis, & E. Pitoura, (Eds.), *Proceedings of the 1st ACM International Workshop on Data Engineering for Wireless and Mobile Access,* (pp. 94–101), New York, NY, USA. ACM Press.

Ren, Q., & Dunham, M. H. (2003). Semantic Caching and Query Processing. *Transactions on Knowledge and Data Engineering, 15*(1), 192–210.

Satyanarayanan, M., Kistler, J. J., Kumar, P., Okasaki, M. E., Siegel, E. H., & Steere, D. C. (1990). Coda: A Highly Available File System for a Distributed Workstation Environment. *IEEE Transactions on Computers, 39*(4), 447–459.

Sybase (2004). *Adaptive Server Anywhere Datenbankadministration,* Bestel lnummer: DC03928-01-0902-01. iAnywhere, A Sybase Company.

v. d. Meyden, R. (1997). The complexity of querying indefinite data about linearly ordered domains. *Journal of Computer and System Sciences, 54,* 113–135.

KEY TERMS

Caching: The implicit storage and reuse of requested, received and processed data.

Cache Invalidation: The process of making cached data consistent by discarding cached data in case of updates.

Cache Maintenance: The process of making cached data consistent by patching cached data.

Hoarding: An extension of caching that caches not only the requested data but also additional, potentially necessary data.

Replication: The explicit (user initiated) creation of redundant data that aims for autonomous (server independent) data processing.

Synchronization: The process of making replicated data consistent by incorporating client and server updates.

Section III
Spatial and Temporal Databases

Chapter XXIX
Spatio-Temporal Indexing Techniques[1]

Michael Vassilakopoulos
University of Central Greece, Greece

Antonio Corral
University of Almeria, Spain

INTRODUCTION

Time and space are ubiquitous aspects of reality. Temporal and Spatial information appear together in many everyday activities, and many information systems of modern life should be able to handle such information. For example, information systems for traffic control, fleet management, environmental management, military applications, local and public administration, and academic institutions need to manage information with spatial characteristics that changes over time, or in other words, Spatio-temporal Information. The need for Spatio-temporal applications has been strengthened by recent developments in mobile telephony technology, mobile computing, positioning technology, and the evolution of the World Wide Web.

Research and technology that aim at the development of Database Management Systems (DBMSs) that can handle Spatial, Temporal, and Spatio-temporal information have been developed over the last few decades. The embedding of spatio-temporal capabilities in DBMSs and GISs is a hot research area that will continue to attract researchers and the informatics industry in the years to come.

In spatio-temporal applications, many sorts of spatio-temporal information appear. For example, an area covered by an evolving storm, the changing population of the suburbs of a city, and the changing coastlines caused by ebb and tide. However, one sort of spatio-temporal information is quite common (and in some respects easier to study) and has attracted the most research efforts:

moving objects or points. For example, a moving vehicle, an aircraft, or a wandering animal.

One key issue for the development of an efficient Spatio-temporal DBMS (STDBMS) is the use of spatio-temporal access methods at the physical level of the DBMS. The efficient storage, retrieval, and querying of spatio-temporal information demands the use of specialized indexing techniques that minimize the cost during management of such information.

In this article, we report on the research efforts that have addressed the indexing of moving points and other spatio-temporal information. Moreover, we discuss the possible research trends within this area of rising importance.

BACKGROUND

The term Spatial Data refers to multidimensional data, like points, line segments, regions, polygons, volumes, or other kinds of geometric entities, while the term Temporal Data refers to data varying in the course of time. Since in database applications the amount of data that should be maintained is too large for main memory, external memory (hard disk) is considered as a storage means. Specialized access methods are used to index disk pages and in most cases have the form of a tree. Numerous indexing techniques have been proposed for the maintenance of Spatial and Temporal Data. Two good sources of related information are the survey by Guenther and Gaede (1998) and the survey by Saltzberg and Tsotras (1999) for spatial and temporal access methods, respectively.

During last years, several researchers have focused on spatio-temporal data (spatial data that vary in the course of time) and the related indexing methods for answering spatio-temporal queries. A spatio-temporal query is a query that retrieves data according to a set of spatial and temporal relationships. For example, "find the vehicles that will be in a distance of less than 5km from a specified point within the next 5 minutes". A number of recent short reviews that summarize such indexing techniques (especially, indexing of moving points) have already appeared in the literature. There are several ways for categorizing (several viewpoints, or classifications of) spatio-temporal access methods. In the rest of this section, we report on the approach followed by each of these reviews and on the material that the interested reader would find there.

In the book "Spatiotemporal Databases: the ChoroChronos Approach", that was authored within the ChoroChronos project and edited by Sellis et al. (2003), chapter 6 is entitled "Access Methods and Query Processing Techniques" and reviews spatio-temporal access methods that have appeared up to 2001. The main classification followed in this chapter is between methods belonging in the R-tree family and methods belonging in the Quadtree family. The principle guiding the hierarchical decomposition of data distinguishes between these two indexing approaches. The two fundamental principles, or hierarchies are:

- the data space hierarchy: a region containing data is split (when, for example, a maximum capacity is exceeded) to sub-regions in a way that depends on these data (for example, each of two sub-regions contains half of the data), and
- the embedding space hierarchy: a region containing data is split (when a certain criterion holds) to sub-regions in a predefined way (for example, a square region is always split in four quadrant sub-regions)

R-trees are data-driven, while Quadtrees are space-driven access methods.

The June 2002 (Vol.25 No.2) issue of the IEEE Data Engineering Bulletin (http://sites.computer.org/debull/A02june/issue1.htm) is devoted to "Indexing of Moving Objects" and is an excellent source of quite updated information. Pfoser (2002) reviews techniques that index the trajectories of

moving objects that follow unconstrained movement (e.g. vessels at sea), constrained movement (e.g. pedestrians) and movement in transportation networks (e.g. trains, or cars). These techniques allow the answering of queries concerning the past of the objects.

On the other hand, Papadopoulos et al. (2002) review techniques that index mobile objects and allow answering of queries about their future positions. The basis of these techniques is the duality transform, that is mapping of data from one data space to another data space, where answering of queries is easier, or more efficient (mapping of a line segment representing a trajectory, to a point in space of equal dimensionality).

Agarwal and Procopiuc (2002) classify indexing techniques according to the consideration of time as another dimension (called time oblivious approach that can be used for answering queries about the past), the use of kinetic data structures (that can be used for answering present queries or even queries that arrive in chronological order) and the combined use of the two techniques (that can be used for efficiently answering queries about the near past or future).

Jensen and Saltenis (2002) discuss a number of techniques that may lead to improved update performance of moving-objects indices. This paper is a good source of possible future research trends.

Chon et al. (2002) report on managing object trajectories by following a partitioning approach that is best suited to answering time-dependent shortest path queries (where the cost of edges varies with time).

Mokbel et al. (2003) review numerous Spatio-Temporal Access Methods classifying them according their ability to index only the past, only the present and the present together with the future status of data. A very descriptive figure that displays the evolution of spatio-temporal access methods with the underlying spatial and temporal structures is included.

Tzouramanis et al. (2004) review and compare four temporal extensions of the Linear Region Quadtree and can store and manipulate consecutive raster images and answer spatio-temporal queries referring to the past.

The book by Güting and Schneider (2005) is a comprehensive presentation of the issues related to moving object databases and includes a chapter dedicated to spatio-temporal indexing.

MAIN TRUST OF THE CHAPTER

In this section, among the numerous spatio-temporal indexing techniques that have appeared in the literature, we briefly review the most fundamental ones.

Quadtree Based Methods

The is a four-way tree where each node corresponds to a subquadrant of the quadrant of each father node (the root corresponds to the whole space). These trees subdivide space in a hierarchical and regular fashion. They are mainly designed for main memory, however several alternatives for secondary memory have been proposed. The most widely used Quadtree is the Region Quadtree that stores regional data in the form of raster images. More details appear in Gaede (1998).

Tayeb et al. (1998) used the PMRquadtree for indexing future trajectories of moving objects. The PMR tree is a tree based on quadtrees, capable of indexing line segments. The internal part of the tree consists of an ordinary region quadtree residing in main memory. The leaf nodes of this quadtree point to the bucket pages that hold the actual line segments and reside on disk. Each line segment is stored in every bucket whose quadrant (region) it crosses. This causes the problem that data replication is introduced (every trajectory, a semi-infinite line, is stored in all the quadrants that it crosses).

Raptopoulou et al. (2004) used a new quadtree based structure, called XBR tree, for indexing past trajectories of moving objects. XBR trees (External Balanced Regular trees) are secondary memory balanced structures that subdivide space in a quadtree manner into disjoint regions. In this paper, XBR trees are shown to excel over PMR trees, when used for indexing past trajectories.

Tzouramanis et al. (2003 & 2004) present and compare four different extensions of the Linear Region Quadtree (the Time-Split Linear Quadtree, the Multiversion Linear Quadtree, the Multiversion Access Structure for Evolving Raster Images and Overlapping Linear Quadtrees) for indexing a sequence of evolving raster data (e.g. images of clouds as they evolve in the course of time). A Linear Region Quadtree is an external memory version of the Region Quadtree, where each quadrant is represented by a codeword stored in a B+-tree. In this paper, temporal window queries (e.g. find the regions intersecting the query window within a time interval) are studied.

Meka and Singh (2005) present a distributed index structure for efficiently tracking moving objects in sensor networks and especially for tracking a chemical plume over a large infrastructure network. The sensor network is hierarchically decomposed in a quad-tree like fashion and a hierarchical distributed index is built, preserving plume's locality in space and time. The proposed structure is scalable with the size of the plume.

R-Tree Based Methods

An R-tree is a balanced multiway tree for secondary storage, where each node is related to a Minimum Bounding Rectangle (MBR), the minimum rectangle that bounds the data elements contained in the node. The MBR of the root bounds all the data stored in the tree. The most widely used R-tee is the R*-tree. More details appear in Gaede (1998).

Hu et al. (1990) presented an R-tree variation called RTtree. The RTtree, couples time intervals with spatial ranges in each node of the tree: each MBR is accompanied by the time interval during which the related object or node is valid. Queries that embed time may require traversal of the whole tree.

In the same paper the MRtree was presented. This R-tree variation employs the idea of overlapping between a sequence of trees (like Overlaping Linear Quadtrees, mentioned above) by storing common subtrees between consecutive trees only once. This tree suffers from reduced performance for temporal window queries. Additionally, a small change in a common subtree causes replication of this subtree between consecutive trees. Nascimento and Silva (1998) presented the HRtree that is very similar to the MRtree. Tao and Papadias (2001a) presented an improved version called the HR+tree that avoids replication of subtrees to some extend. In the HR+tree a node is allowed to have multiple parents.

Theodoridis et al. (1996) presented the 3D Rtree that treats time as one of the three dimensions. Nascimento et al. (1999) presented the 2+3 Rtree: a 2D R-tree is used for current 2 dimensional points and a 3D R-tree for the historical 3 dimensional trajectories. Depending on the query time, both trees may need to be searched.

Tao and Papadias (2001b) presented the MV3R-tree. This consists of a Multiversion R-tree (like the Multiversion Linear Quadtree, mentioned above) to process timestamp queries, and a 3D Rtree to process long interval queries.

Pfoser et al. (2000) presented the STRtree (Spatio-Temporal R-tree), an R-tree with a different insert/split algorithm. This algorithm aims keeping the line segments belonging to the same trajectory together, as much as possible (trajectory preservation). In the same paper, the TBtree, Trajectorybundle tree, was introduced. This is an R-tree like structure that strictly preserves trajectories. A leaf node can only contain segments belonging to the same trajectory. The TB-tree outperforms the STR-tree in trajectory-based queries (queries involving the topology

of trajectories and derived information, such as speed and heading of objects).

Hadjieleftheriou et al. (2004) use a multi-version structure, the Multi-Version R-tree (MVR-tree), to index the history of objects that move and have extents that vary over time. Two problems that arise are excessive dead space and overlap and a number of techniques are proposed for dealing with them.

Saltenis et al. (2000) proposed the TPRtree (Time Parameterized Rtree) for indexing the current and future positions of moving points. This tree introduced the idea of parametric bounding rectangles in the Rtree (bounding rectangles that expand according to the maximum and minimum velocities of the objects that they enclose). To avoid the case where the bounding rectangles grow to be very large, whenever the position of an object is updated, all the bounding rectangles on the nodes along the path to the leaf at which this object is stored are recomputed. Tao et al. (2003) presented an improvement called TPR*tree that uses new insertion and deletion algorithms that aim at minimizing a certain cost function. Moreover, Tao et al. (2004) introduced the STP tree, which is a generalization of the previous two trees to arbitrary polynomial functions describing the movement of objects.

Several researchers have focused on indexing objects moving in Spatial Networks. Frentzos (2003) developed an access method for objects moving on fixed networks, called Fixed Network R-Tree (FNR-Tree). This is a forest of 1-dimensional R-Trees on top of a 2-dimensional R-Tree. The 2D R-Tree indexes the Spatial Network infrastructure (e.g. roads consisting of line segments), while the 1D R-Trees index the time interval of each object's movement on a network edge. De Almeida and Güting (2005) propose an R-tree like index structure for moving objects on networks (trajectory data), called Moving Objects in Networks Tree (MON-tree). This tree indexes both an edge oriented and a route oriented network model.

Gutiérrez et al. (2005) introduce spatio-temporal access method called SEST-Index that indexes two types of data: snapshots of spatial objects after a number of changes occur in the positions of the objects and events that produce these changes. This allows processing of time slice, interval and event queries. The SEST- Index uses an R-tree for storing snapshots and a data structure termed log for storing events that occur between consecutive snapshots.

Lim et al. (2007) propose two indexing structures, called MAR-tree (based on the TB-tree) and the SRTB-tree (an enhanced MAR-tree). The purpose of the introduction of these structures is to exhibit trajectory preservation and at the same time reduce the dead space in nodes and overlapping between nodes, in order to improve performance of range and timeslice queries.

Pelanis et al. (2006) present an indexing technique capable of capturing the positions of moving objects at all points in time, historic, present and near future. The proposed technique is called R^{PPF} –tree (Past, Present, and Future R-tree). It consists of a partially persistent R-tree used to store the past and the present trajectories of the moving objects and a partially persistent TPR-tree used to store the present and the future trajectories.

Transformation Based Methods

Kollios et al. (1999) used the duality transformation to index the current and future positions of moving objects. A moving point in 1D space is represented by a trajectory (line segment) that is transformed to a point in the 2D space (accordingly a moving point in 2D space is transformed to a point in 4D space). The resulting points are indexed by a kdtree based spatial index (a structure more suitable than R-trees, since the distribution of these points is highly skewed). A spatio-temporal range query is transformed into a polygon query in the dual space.

Agarwal et al. (2000) also use a duality transformation. A moving object in the 2D space,

is represnted by a trajectory in 3D space. This trajectory is projected into the (x, t) and (y, t) planes. The duals 2D points of each of these projections are indexed separately. The answer of a spatio-temporal range query is the union of two spatio-temporal range queries in the two planes. A kinetic data structure (Basch et al. 1997) is used to index each dual space.

Other Methods

Pfoser and Jensen (2003) took advantage of the fact that a movement on a specific network edge of a spatial network is 1-dimensional and proposed a technique that reduces the dimensionality of the data (trajectory data in a Spatial Network). The size of the data is reduced and the resulting lower-dimensional indexing method of trajectories of objects moving on a Spatial Network can be supported by a typical database management system.

Li and Lin. (2006) develop an access method, called a Topology-based Mixed Index Structure (TMIS), to index trajectories of objects moving on Spatial Network. TMIS consists of a combination of simple and classical index structures linked by the topology of a network and can support several queries.

Chang et al. (2006) propose a signature-based indexing scheme for trajectories of objects moving on road networks and present experimental results that show better trajectory than other trajectory indexing schemes, like the TB-tree, FNR-tree and MON-tree.

Botea et al. (2007) propose the Practical Index for Spatio-Temporal (PIST) for historical spatio-temporal data (points). This is an indexing technique, rather than a new indexing structure, and it can be fully integrated within existing RDBMSs. PIST partitions the data according to its spatial location and then creates temporal indexes over each partition. By experimentation, PIST has been shown to outperform other spatio-temporal data management methods.

Lin et al. (2005) propose an indexing technique for efficiently supporting queries about the past, present, and future positions of moving objects, called the Broad B^x indexing technique (BB^x-index). This index is based on the B^x tree and stores linearized moving-object locations in a forest of B+-trees and supports spatial and temporal queries.

FUTURE TRENDS

Apart from the development of new, or improved indexing techniques, the following make up a non-exhaustive list of the research trends that are likely to be further addressed in the future.

- The evolution of indexing techniques for other sorts of spatio-temporal data (e.g. evolving boundaries, non-point objects that exhibit not only a changing position, but rotate, change shape, color, transparency, etc) apart from moving objects, or evolving regions.
- The development of access methods and query processing techniques for answering other kinds of queries (e.g. spatio-temporal joins) apart from range queries that have been mainly addressed so far.
- Efficiency and qualitative comparison of the indexing techniques proposed in the literature.
- The adoption of several of the update techniques proposed in Jensen and Saltenis (2002), like the utilization of buffering, the use of all available main memory, the consideration of movement constrains, etc
- Consideration of distributed access methods for spatio-temporal data, as well as addressing of concurrency and recovery issues and incorporation of uncertainty of movement, as proposed in Agarwal and Procopiuc (2002).

CONCLUSION

In this paper, we have reviewed the issues and techniques related to access methods for spatio-temporal data. This research area (and especially indexing of moving objects) has attracted many researchers during last years. Although, this is a relatively new research area, numerous techniques have been developed. However, this is still a hot and demanding research area, where many challenges need to be addressed.

REFERENCES

Agarwal, P. K., & Procopiuc C. M. (2002). Advances in indexing for mobile objects. *IEEE Data Engineering Bulletin, 25*(2), 25-34.

Agarwal, P. K., Arge, L., & Erickson, J. (2000). Indexing moving points. *Proc. of the 19th ACM Symposium on Principles of Database Systems (PODS'2000)*, 175-186. Madison, WI.

Basch, J., Guibas, L. J., & Hershberger, J. (1997). Data structures for mobile data. *Proc. of the ACMSIAM Symposium on Discrete Algorithms (SODA'1997),* 747-756.

Botea, V., Mallett, D., Nascimento, M. A., & Sander, J. (2007). PIST: An efficient and practical indexing technique for historical spatio-temporal point data. *Geoinformatica*, onlinefirst.

Chang, J-W., Um, J-H., & Lee, W-C. (2006). A new trajectory indexing scheme for moving objects on road networks. *Prof of. the 23rd British National Conference on Databases (BNCOD'* 2006), 291-294.

Chon, H. D., Agrawal, D., & Amr, El Abbadi (2002). Data management for moving objects. *IEEE Data Engineering Bulletin, 25*(2), 41-47.

De Almeida, V. T., & Güting, R. H. (2005). Indexing the trajectories of moving objects in networks. GeoInformatica, *9*(1), 33-60.

Frentzos, E. (2003). Indexing objects moving on fixed networks. *Proc. of the 8th International Symposium on Spatial and Temporal Databases (SSTD'2003)*, 289-305. Santorini, Hellas.

Gaede, V., & Günther, O. (1998). Multidimensional access methods. *ACM Computing Surveys, 30*(2), 170-231.

Gutiérrez, G. A., Navarro, G., Rodríguez, A., González, A. F., & Orellana, J. (2005). A spatio-temporal access method based on snapshots and events. *Proc. Of ACM GIS*, 115-124.

Güting, R. H., & Schneider, M. (2005). Moving objects databases. Morgan Kaufmann Publishers.

Hadjieleftheriou, M., Kollios, G., Tsotras, V. J., & Gunopoulos, D. (2006). Indexing spatiotemporal archives. *VLDB Journal, 15*(2), 143-164

Jensen, C. S., & Saltenis, S. (2002). Towards increasingly update efficient moving-object indexing. *IEEE Data Engineering Bulletin, 25*(2), 35-40.

Kollios, G., Gunopoulos, D., & Tsotras, V. J. (1999). On indexing mobile objects. *Proc. of the 18th ACM Symposium on Principles of Database Systems (PODS'1999)*, 261-272. Philadelphia, PA.

Li, X., & Lin, H. (2006). Indexing network-constrained trajectories for connectivity-based queries. *International Journal of Geographical Information Science, 20*(3), 303-328.

Lim D., Cho, D. and Hong B. (2007). Indexing Moving Objects for Trajectory Retrieval on Location-Based Services. *IEICE Transactions on Information and Systems 2007 E90-D(9)*:1388-1397

Lin, D., Jensen, C. S., Ooi, B. C., & Saltenis, S. (2005). Efficient indexing of the historical, present and future positions of moving objects. *Proc. of Mobile Data Management (MDM'2005),* 59-66.

Meka, A., & Singh, A. K. (2005). DIST: A distributed spatio-temporal index structure for sensor networks. *Proc. Int. Conf. on Information and Knowledge Management* (CIKM' 2005), 139-146.

Mokbel, M. F., Ghanem T. M., & Aref, W. G. (2003). Spatio-temporal access method. *IEEE Data Engineering Bulletin, 26*(2): 40-49.

Nascimento, M. A., & Silva, J. R. O.(1998). Towards historical R-trees. *Proc. of the ACM Symposium on Applied Computing (SAC'1998),* 235-240. Atlanta, GA.

Nascimento, M. A., Silva, J. R. O., & Theodoridis, Y. (1999). Evaluation of access structures for discretely moving points. *Proc. of the International Workshop on SpatioTemporal Database Management (STDBM'1999),* 171-188.

Papadopoulos, D., Kollios, G., Gunopulos, D., & Tsotras, V. J. (2002). Indexing mobile objects using duality transforms. *IEEE Data Engineering Bulletin, 25*(2), 18-24.

Pfoser, D. (2002). Indexing the trajectories of moving object. *IEEE Data Engineering Bulletin, 25*(2): 3-9.

Pfoser, D., & C. S. (2003). Indexing of network constrained moving objects. *In proc. ACM-GIS 2003,* 25-32.

Pelanis, M., Saltenis, S., & Jensen, C. S. (2006). Indexing the past, present and anticipated future positions of moving objects. *ACM Transactions on Database Systems 31*(1), 255-298

Pfoser, D., Jensen, C., & Theodoridis, Y. (2000). Novel approaches to the indexing of moving object trajectories. *Proc. of the 26th International Conference on Very Large Databases (VLDB'2000),* 189-200. Cairo, Egypt.

Raptopoulou, K., Vassilakopoulos, M., & Manolopoulos, Y. (2004). Towards quadtree-based moving objects databases. To appear in *Proc. of the Eighth East-European Conference on Advances in Databases and Information Systems (ADBIS'2004).* Budapest, Hungary.

Saltenis, S., Jensen, C. S., Leutenegger, S. T., & Lopez, M. A. (2000). Indexing the positions of continuously moving objects. *Proc. of the ACM SIGMOD International Conference on Management of Data (SIGMOD'2000),* 331-342. Dallas, TX.

Saltzberg, B., & Tsotras, V. J. (1999). Comparison of access methods for time-evolving data. *ACM Computing Surveys, 31*(2), 158-221.

Sellis et al. (Eds.) (2003). Access methods and query processing techniques. *Chapter in the book Spatiotemporal Databases: The ChoroChronos Approach, LNCS* 2520. Springer Verlag.

Tao, Y., & Papadias, D. (2001a). Efficient historical Rtrees. *Proc. of the International Conference on Scientific and Statistical Database Management (SSDBM'2001),* 223-232. George Mason University, Fairfax, Virginia.

Tao, Y., & Papadias, D. (2001b). MV3RTree: A spatiotemporal access method for timestamp and interval queries. *Proc. of the 27th International Conference on Very Large Data Bases, (VLDB'2001),* 431-440. Roma, Italy.

Tao, Y., Papadias, D., & Sun, J. (2003). The TPR*-Tree: An optimized spatiotemporal access method for predictive queries. *Proc. of the 29th International Conference on Very Large Data Bases, (VLDB'2003),* 790-801. Berlin, Germany.

Tao, Y., Faloutsos, C., Papadias, D., & Liu, B. (2004). Prediction and indexing of moving objects with unknown motion patterns. *Proc. of the ACM Conference on the Management of Data (SIGMOD'2004).* Paris, France, to appear.

Tayeb, J., Ulusoy, O., & Wolfson, O. (1998). A quadtree based dynamic attribute indexing method. *The Computer Journal, 41*(3), 185-200.

Theodoridis, Y., Vazirgiannis, M., & Sellis, T. (1996). Spatio-temporal indexing for large multimedia applications. *Proc. of the 3rd IEEE International Conference on Multimedia Computing and Systems (ICMCS'1996),* 441-448. Hiroshima, Japan.

Tzouramanis, T., Vassilakopoulos, M., & Manolopoulos, Y. (2003). Overlapping linear quadtrees and spatio-temporal query processing. *The Computer Journal, 43*(4), 325-343.

Tzouramanis, T., Vassilakopoulos, M., & Manolopoulos, Y. (2004). Benchmarking access methods for time-evolving regional data. *Data & Knowledge Engineering, 49* (3), 243-286.

Xu, X., Han, J., & Lu, W. (1990). RTTree: An improved RTree indexing structure for temporal spatial databases. *Proc. of the International Symposium on Spatial Data Handling (SDH'1990),* 1040-1049. Zurich, Switzerland.

KEY TERMS

Access Method or Indexing: A technique of organizing data that allows the efficient retrieval of data according to a set of search criteria. R-trees and Quadtrees are two well-known families of such techniques.

Constrained (Unconstrained) Movement: Movement (of a moving object) that is (is not) confined according to a set of spatial restrictions.

Movement in Transportation Networks: Movement (of a moving object) that is confined on a transportation network (such as rails, or roads).

Moving Object or Moving Point: A data element that is characterized by its position in space that varies in the course of time (this is a kind of spatio-temporal datum).

Spatio-Temporal Data: Multidimensional data, like points, line segments, regions, polygons, volumes, or other kinds of geometric entities that vary in the course of time.

Spatio-Temporal Data Base Management System: A Data Base Management System that offers spatio-temporal data types and is able to store, index and query spatio-temporal data.

Spatio-Temporal Query: A set of conditions embedding spatial and temporal relationships that defines the set of spatio-temporal data to be retrieved.

Trajectory: The track followed by a moving object in the course of time (due to the change of its position).

ENDNOTE

[1] Supported by the projects "Spatio-temporal data warehouses based on ontologies" (TIN2005-09098-C05-03) of the Spanish Ministry of Science and Technology and "Efficient Image Databases" of the bilateral agreement between Bulgaria and Greece (General Secretariat of Research and Technology of Greece).

Chapter XXX
Query Processing in Spatial Databases

Antonio Corral
University of Almeria, Spain

Michael Vassilakopoulos
University of Central Greece, Greece

INTRODUCTION

Spatial data management has been an active area of intensive research for more than two decades. In order to support spatial objects in a database system several important issues must be taken into account such as: spatial data models, indexing mechanisms and efficient query processing. A spatial database system (SDBS) is a database system that offers spatial data types in its data model and query language and supports spatial data types in its implementation, providing at least spatial indexing and efficient spatial query processing (Güting, 1994).

The main reason that has caused the active study of spatial database management systems (SDBMS) comes from the needs of the existing applications such as geographical information systems (GIS), computer-aided design (CAD), very large scale integration design (VLSI), multimedia information systems (MIS), data warehousing, multi-criteria decision making, location-based services, etc.

Some of the most important companies in the commercial database industry (Oracle, Informix, Autodesk, etc.) have products specifically designed to manage spatial data. Moreover, re-

search prototypes as Postgres and Paradise offer the possibility to handle spatial data. The main functionality provided by these products includes a set of spatial data types such as the point, line, polygon and region; and a set of spatial operations, including intersection, enclosure and distance. The performance enhancement provided by these operations includes spatial access methods and query algorithms over such indexes (e.g. spatial range queries, nearest neighbor search, spatial joins, etc). We must also cite the Open Geographic Information Systems (OGIS) consortium (http://www.opengis.org/), which has developed a standard set of spatial data types and operations and SQL3/SQL99, which is an object-relational query language that provides the use of spatial types and operations.

In a spatial database system, the queries are usually expressed in a high-level declarative language such as SQL; therefore specialized database software has to map the query in a sequence of spatial operations supported by spatial access methods (Shekhar & Chawla, 2003). Spatial query processing refers to the sequence of steps that a SDBMS will initiate to execute a given spatial query. The main target of query processing in the database field is to process the query accurately and quickly (consuming the minimum amount of time and resources on the computer), by using both efficient representations and efficient search algorithms. Query processing in a spatial environment focuses on the design of efficient algorithms for spatial operators (e.g. selection operations, spatial joins, distance-based queries, etc.). These algorithms are both CPU and I/O intensive, despite common assumptions of traditional databases that the I/O cost will dominate CPU cost (except expensive distance-based queries) and therefore an efficient algorithm is one that minimizes the number of disk accesses. In this article we focus on spatial query processing and not spatio-temporal query processing, where the queries refer to both spatial and temporal characteristics of the data (Manolopoulos et al., 2005).

BACKGROUND IN SPATIAL QUERIES AND PROCESSING

From the query processing point of view, the following three properties characterize the differences between spatial and relational databases (Brinkhoff et al., 1993): (1) unlike relational databases, spatial databases do not have a fixed set of operators that serve as building blocks for query evaluation; (2) spatial databases deal with extremely large volumes of complex objects, which have spatial extensions and cannot be sorted in one dimension; (3) computationally expensive algorithms are required to test the spatial operators, and the assumption that I/O costs dominate CPU costs is no longer valid.

We generally assume that the given spatial objects are embedded in d-dimensional Euclidean space (E^d). An object *obj* in a spatial database is usually defined by several non-spatial attributes and one attribute of some spatial data type (point, line, polygon, region, etc.). This spatial attribute describes the geometry of the object $obj.G \subseteq E^d$, i.e. the location, shape, orientation and size of the object. The most representative spatial operations, which are the basis for the query processing in spatial databases (Gaede & Günther, 1998; Shekhar & Chawla, 2003), are: (1) update operations; (2) selection operations (point and range queries); (3) spatial join; and (4) spatial aggregate queries, and (5) feature-based spatial queries.

- *Update operations:* Standard database operations such as modify, create, etc.
- *Point Query (PQ):* Given a query point $p \in E^d$, find all spatial objects O that contain it.
- *Range Query (RQ):* Given a query polygon P, find all spatial objects O that intersect P. When the query polygon is a rectangle, this is called a window query.
- *Spatial Join Query (SJQ):* Given two collections R and S of spatial objects and a spatial predicate θ, find all pairs of objects

(O, O') ∈ R×S (O∈R and O'∈S), where θ(O.G, O'.G) evaluates to true. Some examples of the spatial predicate θ are: *intersects, contains, is_enclosed_by, distance, northwest, adjacent, meets,* etc. For spatial predicates such as *contains, encloses,* or *adjacent,* for example, the intersection join is an efficient filter that yields a set of candidate solutions typically much smaller than the Cartesian product RxS. Recently, an interesting review of the most representative spatial join techniques has been published in (Jacox & Samet, 2007). An extension of the intersection join is the multiway spatial join, which involves an arbitrary number of spatial inputs (Mamoulis & Papadias, 2001). Very interesting distance join queries are being actually studied, for example, closest pair query (Corral et al., 2000), buffer query (Chan, 2003), nearest neighbors join (Böhm & Krebs, 2002), iceberg queries (Shou et al., 2003), distance join queries of multiple inputs (Corral et al., 2004), etc.

- *Spatial Aggregate Queries (SAQ):* This kind of spatial query involves specifying a region of space and asking for the value of some aggregate function (e.g. count, sum, min, max, average) for which we have measurements for this given region (Papadias et al., 2001), and an extension is top-K OLAP queries (Mamoulis et al., 2005). Spatial aggregates are usually variants of the nearest neighbor problem (Shekhar & Chawla, 2003).
- *The Nearest Neighbor Query (NNQ):* Given a spatial object O', find all spatial objects O having a minimum distance from O'. The distance between extended spatial data objects is usually defined as the distance between their closest points (common distance functions for points include the Euclidean and the Manhattan distance). An interesting variant of NNQ is the reverse nearest neighbor query (RNNQ), which reports the points that have the query point as their nearest neighbor (Korn & Muthukrishnan, 2000). Another interesting extension of NNQ is the All-Nearest Neighbor (ANN) query, which takes as input two sets of spatial objects and computes for each spatial object in the first set the nearest neighbor in the second set (Zhang et al., 2004; Chen & Patel, 2007).
- *Feature-Based Spatial Queries (FBSQ):* Recently, algorithms to solve the problem of finding top-K sites based on their influence on feature points were proposed in (Xia et al., 2005). Related to top-K influential sites query are the optimal location queries, the goal of which is to find the location in space that minimizes an objective function where all feature points have the same quality. The maximum influence optimal location query (Du et al., 2005) finds the location with the maximum influence, whereas the minimum distance optimal location query (Zhang et al., 2006) searches for the location that minimizes the average distance from each feature point to its nearest site. Finally, the top-K preference query ranks the K spatial objects in a set with highest scores (Yiu et al., 2007), where the score of an object is defined by the quality of features in its spatial neighborhood.

The simple spatial queries are often processed using filter and refine techniques to minimize both the CPU and I/O cost (Brinkhoff et al., 1994), and other complex spatial queries combine simple ones (Mamoulis et al., 2004). Approximate geometry such as the minimal orthogonal bounding rectangle (MBR) of an extended spatial object is first used to filter out many irrelevant objects quickly. An MBR is characterized by *min* and *max* points of hyper-rectangles with faces parallel to the coordinate axes. Using the MBR instead of the exact geometrical representation of the spatial object, its representational complexity is reduced to two points, where the most important object features (position and extension) are maintained. The R-

tree (Guttman, 1984) is a spatial access method that represents the spatial objects by their MBR, and it is a height-balanced tree. An excellent book on R-trees is (Manolopoulos et al., 2006). Therefore, in the filter step many candidates are eliminated using the spatial predicate and the MBRs of the spatial objects. In the refinement step, the exact geometry of each spatial object from the candidate set (result of the filter step) and the exact spatial predicate are examined. This step usually requires the use of CPU-intensive algorithms, like computational geometry algorithms for spatial operations (Rigaux et al., 2001). Strategies for range-queries include a scan and index-search in conjunction with the plane-sweep algorithm (Brinkhoff et al., 1993). Strategies for the spatial join include the nested loop, tree matching (Brinkhoff et al., 1993; Huang et al., 1997), when indices are present on all participating inputs and space partitioning (Patel & DeWitt, 1996; Lo & Ravishankar, 1996; Arge et al., 1998) in absence of indexes. For the case when one spatial input is indexed, the most representative join strategies have been proposed by Lo & Ravishankar (1994) and Mamoulis & Papadias (2003).

Nearest neighbor queries (NNQ) are common in many applications, for example, GIS, pattern recognition, document retrieval, learning theory, etc. As the previous spatial queries, NNQ algorithms are also two-step algorithms (filter-refine paradigm), and they follow branch-and-bound techniques, using distance functions and pruning heuristics in order to reduce the search space. The most representative algorithms to perform NNQ over spatial data have been proposed by Roussopoulos et al. (1995) and Hjaltason & Samet (1999) on R-trees. The first query algorithm follows a depth-first traversal, whereas the second one is an incremental algorithm following a best-first search on the R-tree. These algorithms can be extended to find K-nearest neighbors by slight modification of the pruning rules to retain the K best candidates.

PERSPECTIVE AND NEW IMPORTANT ISSUES

We have reviewed the most representative spatial queries, that are mainly based on the overlap predicate for range queries and spatial join queries. However, there is a need to develop and evaluate query strategies for many other frequent spatial queries that can be demanded by the users in a spatial database system. Table 1 summarizes some of these new spatial queries, which include queries on objects using predicates other than overlap and queries on fields such as slope analysis as well as queries on networks, etc. (Shekhar et al., 1999).

The ever-increasing demand and easy availability of the Internet have prompted the development of Web-based Geographic Information Systems (WGIS), for easy sharing of spatial data and executing spatial queries over the Internet using Web environments, like web servers, web clients, common data formats (HTML, XML), common communication protocols (http), uniform resource locator (URL), etc.. In a WGIS architecture (Shekhar & Chawla, 2003), the Geo-Spatial Database Access Layer (GSDAL) allows to access the spatial data using a SDBMS, where efficient query processing is required. In order to improve Web-Based Spatial Database Systems (WSDBS), new important issues for SDBMSs and Web Technology need to be addressed. For example, to offer SDBMS services on the Web, to evaluate and improve the Web for SDBMS clients and servers, to use Web data formats for spatial data (e.g. WMS (Web Map Service) and GML (Geography Markup Language)), to employ safe communication protocols, to allocate adequate search facilities on the Web, to develop compatibility between several WSDBS, to improve maintenance and integrity of data, etc. A recent contribution is this field is (Chen et al., 2006), which proposes a web-search engine for evaluating textual geographic queries by considering the spatial context of the searched documents.

Table 1. A list of new spatial queries

Buffer	Find the areas 500 meters away from power lines
Voronoize	Classify households as to which supermarket they are closest to
Neighborhood	Determine slope based on elevation
Network	Find the shortest path from the warehouse to all delivery stops
Allocation	Where is the best place to build a new restaurant?
Transformation	Triangulate a layer based on elevation
Ranking	Find the top-K hotels with the largest number of nearby restaurants
Chromatic	Find the type of monument nearest to the Alhambra
Aggregate	Find the total number of (restaurant, hotel) pairs that are within 1 km from each other
Multi-way	Find all cities within 100 km of Madrid crossed by a river which intersects an industrial area
Feature-based	Find the two most influential supermarkets in Madrid

FUTURE TRENDS

Most of the spatial query algorithms have been studied over point datasets in the 2-dimensional space, and a natural extension is to study their application on datasets containing objects with spatial extent (lines, polygons, regions, 3D objects, etc.). In order to carry out this work, geometric algorithms (based on computational geometry) have to be designed and analyzed for spatial operators using these complex spatial objects (Rigaux et al., 2001).

Many open research areas exist at the logical level of query processing, including query-cost modeling and queries related to spatial networks. Cost models are used to rank and select the promising processing strategies, given a spatial query and a spatial dataset. Traditional cost models may not be accurate in estimating the cost of strategies for spatial operations, mainly due to the distance metric. Cost models are needed to estimate the selectivity of spatial search and join operations toward comparison of execution-costs of alternative processing strategies for spatial operations during query optimization. Preliminary work in the context of the R-tree, using fractal-models for NNQ (Belussi & Faloutsos, 1998), using the concept of Minkowski sum for RQ and NNQ (Böhm, 2000); using the concept of density of a rectangle set for RQ and SJQ (Theodoridis et al., 2000) have been developed, and in (Corral et al., 2006) cost models for distance join queries have been proposed, but more work is needed.

Spatial network databases are an important component of SDBS, since this is the kernel of many important real-life applications as transportation planning, air traffic control, urban management, electric, telephone and gas networks, river transportation, etc. Previous efforts in this research area have been focused on disk-based graph representation as the connectivity clustered access method (CCAM) (Shekhar & Chawla, 2003), nearest neighbor queries in road networks by transforming the problem to a high dimensional space (Shahabi et al., 2002), and a flexible architecture that integrates network

representation (preserving connectivity and locations) and Euclidean restriction for processing the most common spatial queries (range search, nearest neighbors, spatial joins and closest pairs) (Papadias et al., 2003). Interesting research has been developed in this field, but much more work is needed mainly in measuring the CPU and I/O cost for network operations, in new queries as spatial skyline (Sharifzadeh & Shahabi, 2006) or aggregate nearest neighbor (Yui et al., 2005), or in moving objects environment (Mouratidis et al., 2006; George et al., 2007), etc.

Feature-based spatial query (Xia et al., 2005; Zhang et al., 2006; Yiu et al., 2007) is a recent line of researching, where all previous approaches have employed indexes and the use of non-indexed data (access methods based on hashing as in (Zhang et al., 2004) for all-nearest neighbors queries) is an interesting open problem in this topic.

CONCLUSION

Spatial query processing refers to the sequence of steps that a SDBMS will initiate to execute a given spatial query. The main target of query processing in the database field is to process the query accurately and quickly, by using both efficient representations and efficient search algorithms. Query processing in a spatial environment focuses on the design of efficient algorithms for spatial operators (e.g. selection operations, nearest neighbor search, spatial joins, etc.). Spatial query operations can be classified into five groups: *point*, *range*, *spatial join*, *spatial aggregate* and *feature-based*. For spatial query processing, the filter-refine paradigm is used over spatial access methods to minimize both the CPU and I/O cost. Future research trends include the study of new spatial queries (especially on spatial networks), the study of issues related to Web-Based Spatial Database Systems and work on cost models for estimating the selectivity of spatial queries.

ACKNOWLEDGMENT

This research was supported by the "Spatio-temporal data warehouses based on ontologies" (TIN2005-09098-C05-03) project of the Spanish Ministry of Science and Technology and "Efficient Image Databases" bilateral agreement between Bulgaria and Greece (General Secretariat of Research and Technology of Greece).

REFERENCES

Arge, L., Procopiuc, O., Ramaswamy, S., Suel, T., & Vitter, J. S. (1998). Scalable Sweeping-Based Spatial Join. *In Proceedings of 24th International Conference on Very Large Data Bases (VLDB 1998)*, New York City, New York, USA, 24-27 August, 1998, (pp. 570-581). San Francisco: Morgan Kaufmann.

Belussi, A., & Faloutsos, C. (1998). Self-Spacial Join Selectivity Estimation Using Fractal Concepts. *ACM Transactions on Information Systems, 16*(2), 161-201.

Böhm, C. (2000). A cost model for query processing in high dimensional data spaces. *ACM Transactions on Database Systems, 25*(2), 129-178.

Böhm, C., & Krebs, F. (2002). High Performance Data Mining Using the Nearest Neighbor Join. *In Proceedings of the 2002 IEEE International Conference on Data Mining (ICDM 2002)*, Maebashi City, Japan, 9-12 December, 2002, (pp. 43-50), IEEE Computer Society Press.

Brinkhoff, T., Kriegel, H. P., & Seeger, B. (1993). Efficient Processing of Spatial Joins Using R-Trees. *In Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data (SIGMOD 1993)*, Washington D.C., Washington, USA, 26-28 May, 1993, (pp. 237-246), ACM Press.

Brinkhoff, T., Kriegel, H. P., Schneider, R., & Seeger, B. (1994). Multi-Step Processing of Spatial Joins. *In Proceedings of the 1994 ACM SIGMOD International Conference on Management of Data (SIGMOD 1994)*, Minneapolis, Minnesota, USA, 24-27 May, 1994, (pp. 197-208), ACM Press.

Chan, E. P. F. (2003). Buffer Queries. *IEEE Transactions Knowledge Data Engineering, 15*(4), 895-910.

Chen, Y. Y., Suel, T., & Markowetz, A. (2006). Efficient Query Processing in Geographic Web Search Engines. *In Proceedings of the 2006 ACM SIGMOD International Conference on Management of Data (SIGMOD 2006)*, Chicago, Illinois, USA, June 27-29, 2006, (pp. 277-288), ACM Press.

Chen, Y., & Patel, J. M. (2007). Efficient Evaluation of All-Nearest-Neighbor Queries. *In Proceedings of the 23rd IEEE International Conference on Data Engineering (ICDE 2007)*, Istanbul, Turkey, 15-20 April, 2007, (pp. 1056-1065). IEEE Computer Society Press.

Corral, A., Manolopoulos, Y., Theodoridis, Y., & Vassilakopoulos, M. (2000). Closest Pair Queries in Spatial Databases. *In Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data (SIGMOD 2000)*, Dallas, Texas, USA, 16-18 May, 2000, (pp. 189-200), ACM Press.

Corral, A., Manolopoulos, Y., Theodoridis, Y., & Vassilakopoulos, M. (2004) Multi-way Distance Join Queries in Spatial Databases. *GeoInformatica, 8*(4), 373-400.

Corral, A., Manolopoulos, Y., Theodoridis, Y., & Vassilakopoulos, M. (2006). Cost models for distance joins queries using R-trees. *Data and Knowledge Engineering, 57*(1), 1-36.

Du, Y., Zhang, D., & Xia, T. (2005). The Optimal-Location Query. *In Proceedings of the Symposium on Spatial and Temporal Databases, (SSTD 2005)*, Angra dos Reis, Brazil, 22-24 August, 2005, LNCS 3633, (pp. 163-180). Springer.

Gaede, V., & Günther, O. (1998). Multidimensional Access Methods. *ACM Computing Surveys, 30*(2), 170-231.

George, B., Kim, S., & Shekhar, S. (2007). Evaluation of Iceberg Distance Joins. *In Proceedings of the 10th International Symposium on Spatial and Temporal Databases (SSTD 2007)*, Boston, MA, USA, July 16-18, 2007, Lecture Notes in Computer Science 4605, (pp. 460-477), Springer.

Güting, R. (1994). An Introduction to Spatial Database Systems. *VLDB Journal, 3*(4), 357-399.

Guttman, A. (1984). R-trees: A Dynamic Index Structure for Spatial Searching. *In Proceedings of the 1984 ACM SIGMOD International Conference on Management of Data (SIGMOD 1984)*, Boston, Massachusetts, 18-21 June, 1984, (pp. 47-57), ACM Press.

Hjaltason, G. R., & Samet, H. (1999). Distance Browsing in Spatial Databases. *ACM Transactions on Database Systems, 24*(2), 265-318.

Huang, Y. W., Jing, N., & Rundensteiner, E. A. (1997). Spatial Joins Using R-trees: Breadth-First Traversal with Global Optimizations. *In Proceedings of 23rd International Conference on Very Large Data Bases (VLDB 1997)*, Athens, Greece, 25-29 August, 1997, (pp. 396-405), Morgan Kaufmann, San Francisco.

Jacox, E. H., & Samet, H. (2007). Spatial joins techniques. *ACM Transactions on Database Systems, 32*(1), 7 1-44.

Korn, F., & Muthukrishnan, S. (2000). Influence Sets Based on Reverse Nearest Neighbor Queries. *In Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data (SIGMOD 2000)*, Dallas, Texas, USA, 16-18 May, 2000, (pp. 201-212), ACM Press.

Lo, M. L., & Ravishankar, C. V. (1994). Spatial Joins Using Seeded Trees. *In Proceedings of the 1994 ACM SIGMOD International Conference on Management of Data (SIGMOD 1994)*, Minneapolis, Minnesota, USA, 24-27 May, 1994, (pp. 209-220), ACM Press.

Lo, M. L., & Ravishankar, C. V. (1996). Spatial Hash-Joins. *In Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data (SIGMOD 1996)*, Montreal, Quebec, Canada, June 4-6, 1996, (pp. 247-258), ACM Press.

Mamoulis, N., & Papadias, D. (2001). Multiway spatial joins. *ACM Transactions on Database Systems, 26*(4), 424-475.

Mamoulis, N., & Papadias, D. (2003). Slot Index Spatial Join. *IEEE Transactions Knowledge Data Engineering, 15*(1), 211-231.

Mamoulis, N. Papadias, D., & Arkoumanis, D. (2004). Complex Spatial Query Processing. *GeoInformatica, 8*(4), 311-346.

Mamoulis, N., Bakiras, S., & Kalnis, P. (2005). Evaluation of Top-k OLAP Queries Using Aggregate R-trees. *In Proceedings of the Symposium on Spatial and Temporal Databases, (SSTD 2005)*, Angra dos Reis, Brazil, 22-24 August, 2005, LNCS 3633, (pp. 236-253), Springer.

Manolopoulos, Y., Papadopoulos, A. N., & Vassilakopoulos, M. (2005). *Spatial Database: Technologies, Techniques and Trends.* Idea Group Publishing.

Manolopoulos, Y., Nanopoulos, A., Papadopoulos, A. N., & Theodoridis, Y. (2006). *R-Trees: Theory and Applications.* Advanced Information and Knowledge Processing Series, Springer.

Mouratidis, K., Yiu, M. L., Papadias, D., & Mamoulis, N. (2006). Continuous Nearest Neighbor Monitoring in Road Networks. *In Proceedings of the Very Large Data Bases Conference (VLDB 2006)*, Seoul, Korea, 12-15 September, 2006, (pp. 43-54). Morgan Kaufmann, San Francisco.

Papadias, D., Zhang, J., Mamoulis, N., & Tao, Y. (2003). Query Processing in Spatial Network Databases. *In Proceedings of the Very Large Data Bases Conference (VLDB 2003)*, Berlin, Germany, 9-12 September, 2003, (pp. 802-813). Morgan Kaufmann, San Francisco.

Papadias, D., Kalnis, P., Zhang, J., & Tao, Y. (2001). Efficient OLAP Operations in Spatial Data Warehouses. *In Proceedings of the Symposium on Spatial and Temporal Databases, (SSTD 2001)*, Redondo Beach, CA, USA, 12-17 July, 2001, LNCS 2121, (pp. 443-459). Springer.

Patel, J. M., & DeWitt, D. J. (1996). Partition Based Spatial-Merge Join. *In Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data (SIGMOD 1996)*, Montreal, Quebec, Canada, June 4-6, 1996, (pp. 259-270), ACM Press.

Rigaux, P., Scholl, M. O., & Voisard, A. (2001). *Spatial Databases: With Application to GIS*. Morgan Kaufmann.

Roussopoulos, N., Kelley, S., & Vincent, F. (1995). Nearest Neighbor Queries. *In Proceedings of the 1995 ACM SIGMOD International Conference on Management of Data (SIGMOD 1995)*, San Jose, California, USA, 22-25 May, 1995, (pp. 71-79). ACM Press.

Shahabi, C., Kolahdouzan, M., & Sharifzadeh, M. (2002). A Road Network Embedding Technique for K-Nearest Neighbor Search in Moving Object Databases. *In Proceedings of the 1996 ACM Symposium on Advances in Geographic Information Systems (ACM GIS 1996)*, McLean, VA (near Washington, DC), USA, 8-9 November, 2002, (pp. 94-100), ACM Press.

Shekhar, S., Chawla, S., Ravada, S., Fetterer, A., Liu, X., & Lu, C. T. (1999). Spatial Databases

- Accomplishments and Research Needs. *IEEE Transactions Knowledge Data Engineering, 11*(1), 45-55.

Shekhar, S., & Chawla, S. (2003). *Spatial Databases: A Tour*. Prentice Hall.

Sharifzadeh, M. and Shahabi, C. (2006) The Spatial Skyline Queries. *In Proceedings of the Very Large Data Bases Conference (VLDB 2006),* Seoul, Korea, 12-15 September, 2006, (pp. 751-762). Morgan Kaufmann, San Francisco.

Shou, Y., Mamoulis, N., Cao, H., Papadias, D., & Cheung, D. W. (2003). Evaluation of Iceberg Distance Joins. *In Proceedings of the 8th International Symposium on Spatial and Temporal Databases (SSTD 2003),* Santorini Island, Greece, 24-27 July, 2003, Lecture Notes in Computer Science 2750, (pp. 270-288), Springer.

Theodoridis, Y., Stefanakis, E., & Sellis, T. (2000). Efficient Cost Models for Spatial Queries Using R-Trees. *IEEE Transactions Knowledge Data Engineering, 12*(1), 19-32.

Xia, T., Zhang, D., Kanoulas, E., & Du, Y. (2005). On Computing Top-t Most Influential Spatial Sites. *In Proceedings of the Very Large Data Bases Conference (VLDB 2005),* Trondheim, Norway, August 30 - September 2, 2005, (pp. 946-957). Morgan Kaufmann, San Francisco.

Yiu, M. L., Dai, X., Mamoulis, N., & Vaitis, M. (2007). Top-k Spatial Preference Queries. *In Proceedings of the 23rd IEEE International Conference on Data Engineering (ICDE 2007),* Istanbul, Turkey, 15-20 April, 2007, (pp. 1076-1085). IEEE Computer Society Press.

Yui, M. L., Mamoulis, N., & Papadias, D. (2005). Aggregate Nearest Neighbor Queries in Road Networks. *IEEE Transactions Knowledge Data Engineering, 17*(6): 820-833.

Zhang, D., Du, Y., Xia, T., & Tao, Y (2006). Progressive Computation of the Min-Dist Optimal-Location Query. *In Proceedings of the Very Large Data Bases Conference (VLDB 2006),* Seoul, Korea, 12-15 September, 2006, (pp. 643-654). Morgan Kaufmann, San Francisco.

Zhang, J., Mamoulis, N., Papadias, D., & Tao, Y. (2004). All-Nearest-Neighbors Queries in Spatial Databases. *In Proceedings of the 16th Scientific and Statistical Database Management Conference (SSDBM 2004),* Santorini Island, Greece, 21-23 June, 2004, (pp. 297-306), IEEE Computer Society Press.

KEY TERMS

Buffer Query: This spatial query involves two spatial datasets and a distance threshold δ. The answer is a set of pairs of spatial objects from the two input datasets that are within distance δ from each other.

Filter-Refine Paradigm: Algorithms that follow this paradigm are two-step algorithms. *Filter step:* an approximation of each spatial object is used to produce a set of candidates (and, possibly, a set of actual answers), which is a superset of the answer set consisting of actual answers and false hits. *Refinement step:* each candidate from the filter step is then examined with respect to its exact geometry in order to produce the answer set by eliminating false hits.

Iceberg Distance Join: This spatial query involves two spatial datasets, a distance threshold δ and a cardinality threshold K ($K \geq 1$). The answer is a set of pairs of objects from the two input datasets that are within distance δ from each other, provided that the first object appears at least K times in the join result.

K Closest Pairs Query: This spatial query involves two spatial datasets and a cardinality threshold K ($K \geq 1$). It discovers the K distinct pairs

of objects from the two input datasets that have the K smallest distances between them.

K Nearest Neighbors Join: This spatial query involves two spatial datasets and a cardinality threshold K ($K \geq 1$). The answer is a set of pairs from the two input datasets that includes, for each of the spatial objects of the first dataset, the pairs formed with each of its K nearest neighbors in the second dataset.

Spatial Data Types: Spatial data types provide a fundamental abstraction for modeling the structure of geometric entities in space (geometry) as well as their relationships (topology), e.g. points, lines, polygons, regions, etc. A *spatial object* is an object with at least one attribute of a spatial data type.

Spatial Database System (SDBS): A spatial database system is a database system that offers spatial data types in its data model and query language and supports spatial data types in its implementation, providing at least spatial indexing and efficient spatial query processing.

Spatial Operators: Spatial operators represent the spatial relationships between spatial objects. The most representative spatial relationships are: (1) Topological relationships, such as adjacent, inside, disjoint, etc. are invariant under topological transformations like translation, scaling, and rotation; (2) Direction relationships, for example, above, below, north_of, southwest_of, etc.; and (3) Metric relationships, e.g. distance < 100.

Spatial Query Processing: It focuses on extracting information from a large amount of spatial data without actually changing the spatial database. It is different to the concept of query optimization that focuses in finding the best query evaluation plan that minimizes the most relevant performance measure (e.g. CPU, I/O, etc.).

Spatial Query: It is a set of spatial conditions characterized by spatial operators that form the basis for the retrieval of spatial information from a spatial database system.

Top-K Most Influential Site Query: Given a set of sites S, a set of weighted objects O, a spatial region Q and an integer K ($K \geq 1$), the top-K most influential site query retrieves K sites in Q with the largest influence. Where, the influence of a site $s \in S$ is the total weight of objects in O that have s as the nearest site.

Chapter XXXI
Automatic Data Enrichment in GIS Through Condensate Textual Information

Khaoula Mahmoudi
High School of Communications – Tunis (SUPCOM), Tunisia

Sami Faïz
National Institute in Applied Sciences and Technology (INSAT), Tunisia

INTRODUCTION

Geographic Information Systems (GIS) (Faïz, 1999) are being increasingly used to manage, retrieve, and store large quantities of data which are tedious to handle manually. The GIS power is to help managers make critical decisions they face daily. The ability to make sound decisions relies upon the availability of relevant information. Typically, spatial databases do not contain much information that could support the decision making process in all situations. Besides, Jack Dangermond, president of a private GIS software company, argued that *"The application of GIS is limited only by the imagination of those who use it"*. Hence, it is of primary interest to provide other data sources to make these systems rich information sources.

To meet these information requirements, data enrichment strategies have been undertaken, broadening the amount of information the users can access. To enrich data stored in the geographic database (GDB), we extract knowledge from online textual documents corpora. This is accomplished by using multi-document summarization (Barzilay et al., 2005). To obtain complementary data in a reasonable time, we propose a distributed multi-document summarization approach. We do so, because the summary generation among a set of documents can be seen as a naturally distributed problem. Hence, each document is considered as an agent, working towards a concise representation

of its own document, which will then be included as part of the corpus summary. In conformity with the multi-agent paradigm (Ferber, 1999), we use a set of cooperating autonomous agents: an *Interface* agent, *Geographic* agents, and *Task* ones. These agents collaborate to jointly lead the system to an optimal summary. The approach we propose is modular. It consists of: thematic delimitation, theme identification, delegation, and text filtering. Besides these main steps, we propose a refinement process that can be executed in the case of unsatisfactory results.

The reminder of the chapter is organized as follows. In section II, we present a review of the enrichment literature. Section III is dedicated to our data enrichment process. In section IV, we describe the refinement process. The implementation of our approach is reported in section V. Finally, in section VI, we present the evaluation of our system.

BACKGROUND

Typically the database enrichment can be classified as; spatial enrichment and semantic one.

Spatial enrichment is interested to the spatial aspect of the GDB. One use of this enrichment is its application within the overall generalization process. In this context, the newly acquired information are used to provide geometrical knowledge and procedural knowledge in terms of generalisation algorithms and operations order, to guide the choice of the generalization solution (Plazanet, 1996).

For the semantic enrichment, it aims to link unstructured data to the already available structured thematic data (also referred as aspatial, descriptive or semantic) stored in the GDB. The works classified under this category are: Metacarta (MetaCarta, 2005), GeoNode (Hyland et al., 1999), Persus (David, 2002).

MetaCarta's technology provides a bridge between document systems and GIS systems. MetaCarta, allows the semantic enrichment through the Geographic Text Search (GTS). GTS allows to link textual documents to geographic entities localised in digital maps to add supplementary data to GDB. GTS is offered as an extension to the GIS *ArcGIS*.

For GeoNode (Geographic News On Demand Environment), given a sequence of news stories, GeoNode, can identify different events that happen at particular time and place. GeoNode makes use of the Information Extraction technique and more precisely the *Alembic* system to accomplish the enrichment. The latter allows to determine the named entities for geospatial visualizations. The GIS *ArcView* supports GeoNode.

Persus is initially conceived as Digital Library focusing on historical documents relative to past events. Persus has incorporated tools allowing to GIS to make use of the historical collection to bring out knowledge to enrich the GDB. The processing of the collection consists in determining the terms, the toponyms, the dates and estimate the co-occurrence of the dates and emplacements to determine eventual events. This mean of enrichment is explored by the *TimeMap* GIS.

What distinguish the data enrichment approach we propose, is that it goes beyond merely providing and linking the relevant documents related to geographic entities, we instead process the documents to locate the gist of information embedded into them. Besides, our approach exploits the spatial relations inherent to GIS to refine the enrichment results.

MAINTHRUST: OUR DATA ENRICHMENT PROCESS

The data enrichment process we propose (Mahmoudi & Faïz, 2006$_a$, Mahmoudi & Faïz, 2006$_b$, Faïz & Mahmoudi, 2005) emanate from informational requirements claimed by GIS users. Hence,

the user submitting a query to the GDB, and being unsatisfied, can launch our enrichment process. The different steps of our enrichment approach are detailed in the sequel.

Thematic Segmentation

At this level, each *Task* agent processes the text of its document to determine the different segments that are topically homogenous. Before starting, each *Task* agent pre-processes its document by discarding the auxiliary information and stop-words (prepositions, articles…). Besides, texts are stemmed to reduce words to stems.

To perform the segmentation, we have used in previous works (Faïz & Mahmoudi, 2005) the TextTiling (Hearst, 1997) algorithm which sub-divided the text into pseudo-sentences of a fixed length. The pseudo-sentences are gathered into blocks of a fixed size (figure 1, step 1). Between each pair of adjacent blocks (at a gap), the cohesion score is computed according to the cosine similarity measure. These scores are plotted against the sequence of gaps (step 4). The computed scores serve to compute the depth scores to depict the segment boundaries (step 5).

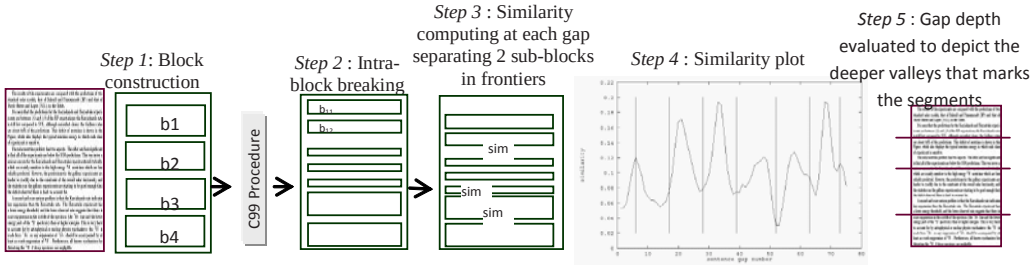
In fact, the main drawback of TextTiling stems from the initial cutting (step 1). Sometimes, while dividing the text into blocks, we put roughly some sentences within the same block whereas they are logically heterogeneous. Hence, during the similarity computing we obtain high coherence values revealing that the two adjacent blocks belong to the same segment.

To deal with this situation, we perform an intra-block division aiming to discard the intruders (heterogeneous sentences) from each block. To perform this, we apply the C99 procedure (Choi, 2000) to each block of step 1. C99 is known to be the more relevant algorithm for the small-sized texts. C99 first constructs a similarity matrix between sentences of the text, using the cosine similarity measure. Then a ranking is done, determining for each pair of sentences, the rank of its similarity measure compared to its *n* x *m-1* neighbours, *n* x *m* being the chosen ranking mask with *n* and *m* being number of sentences. The rank is the number of neighbouring elements with a lower similarity value. To determine the topic boundaries, C99 adopts a divisive clustering.

$$sim = \frac{\sum_t w_{t,b_1} w_{t,b_2}}{\sqrt{\sum_t w_{t,b_1}^2 \sum_t w_{t,b_2}^2}}$$

To accomplish our segmentation process, we pass each block of step one as short text to the C99 procedure. At the end of this operation either we preserve the block as it, hence confirming its homogeneity, or we get other thematic breakings (step 2) of the block into sub-blocks (b1 subdivided into b11 and b12). Then, the sub-blocks in frontier

Figure 1. Adaptation of the textTiling (wtb; frequency of token t in block b)



are passed to the step 3 and will carry on the other steps. For the non-frontal sub-blocks, they will be classified definitively as segments and will not undergo the others segmentations steps.

Once the segmentation is over, each *Task* agent maintains a segmented document, for which it will perform a theme identification task.

Theme Identification

Each *Task* agent looks for annotating each of its segments with the most salient concept. We maintain only the nouns that are considered as being the more meaningful.

Two cases are possible. First, the frequencies are heterogeneously distributed. Then, the term with the highest frequency is affected as segment topic. We assume that the term that characterizes a text topic is usually stressed in by having the highest frequency among all terms.

Nevertheless, this is not always the case, it is frequent to have a text as a set of terms which together contribute to develop a subject. Here, the terms are homogeneously distributed and none of them is remarkably reproduced. In this situation, we use the semantic network ***WordNet*** (Miller, 1990). By doing so, we intend to depict the term to which almost all the terms are related to. Then, we adjust the frequencies to the number of links, a noun maintains with the other segment nouns. Hence, the most referenced word via semantic relations will be assigned as a text topic. For the sake of clarity, we consider the following text:

"The state run Tunisian Radio and Television Establishment operates two national TV channels and several radio networks. Until 2003 the state had a monopoly on radio broadcasting".

None of the nouns is highly occurring and by the way we can't deduce the text theme. We exploit the *WordNet* to depict the semantic relations among the nouns. The main relations are reproduced in what follows:

Hypernymy (TV)= broadcasting
Hypernymy (radio)= broadcasting
Synonym (TV)=television
Topic-domain (network)= broadcasting
…

Then we infer that *broadcasting* is the noun that reveals the main idea of the text.

Delegation

By receiving the topics set tackled throughout the corpus, the *Interface* agent (or each *Geographic* agent in case of a multitude of geographic entities) gathers them into groups. Hence, it maintains for each theme, the potential *Task* agents list. For each topic, the *Interface* agent affects one *Task* agent as a delegate. Each delegate will be responsible of generating a condensate textual representation of the themes under its jurisdiction. While assigning the delegates, the *Interface* agent has to distribute equitably the work overload and to minimize the communication overhead. Formally, the cost function to be minimized is:

$$\text{Delegation}(D) = \alpha\text{workload} + \beta\text{communication}$$

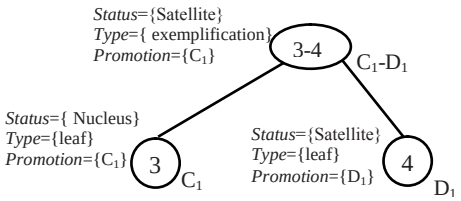
By the way each delegate agent solicits its acquaintances (*Task* agents tackling the themes under its jurisdiction) to send the segments dealing with the same themes. By collecting the segments, each delegate maintains a so-called *generated document* for each topic under its responsibility.

Text Filtering

At this stage the aim is to discard all subsidiary information from the resulting *generated documents*, not needed to express the main ideas tackled in the documents. To perform the text filtering we have been inspired from the Rhetorical Structure Theory (Marcu, 1999, Mann et al., 1988).

To explain our issue we detail Figure 2.

Figure 2. RS-Tree of the text



Text : [A number of reforms were introduced in the past few years,C1] [for instance, the promotion of water users' associations, the increase in the price of irrigation water, etc.D1]

To filter the texts, we build for each segment a rhetorical structure tree which is a binary one whose leaves denote elementary textual units (C1, D1) and whose internal nodes correspond to contiguous text spans (C1-D1). Each node has associated a status (nucleus; C1 or satellite; D1), a type (the rhetorical relation that holds between the text spans that node spans over; exemplification), and a salience or promotion set (the set of units constituting the most important parts of the text that is spanned by that node).

By sweeping the resulting tree in a top-bottom fashion, we depict the units that are most important for the comprehension of the texts. In figure 2, the cue phrase is "*for instance*", the unit to which the cue belongs (D1) can be discarded because it is just an exemplification supporting a previous unit (C1).

The units derived from each *generated document* form a summary of the theme. Hence, each delegate generates a set of partial summaries to be included as a part of the final corpus summary. Such information is dispatched towards the GIS user by the *Interface* agent to judge its relevancy.

REFINEMENT PROCESS

Sometimes, the data enrichment process is fruitless. This may be engendered from the IR step. This step can be blemished by a lack of information or a certain ambiguity; hence the decision can't be made. In the first case, we have to deduce the information from the available ones. In the case of ambiguity, especially when the same noun refers to more than one geographic entity; the GIS user is overwhelmed by a large set of non-relevant documents.

To obtain the desirable information relative to the target entities, one has to describe the latter in an accurate way. By doing so, we have more chance to retrieve the relevant documents.

In fact, because the geographic entities are not disparate rather they maintain spatial relationships with the other entities, one can exploit these relationships to better locate them. The idea is to examine the vicinity of the geographic entity in order to disclose the geographic entities that have some relationships with it. Our argument is the first law of geography that describes the geographic systems nature in which "*everything is related to everything else, but near things are more related than distant things*" (Bin et al., 2006).

To perform this refinement, we exploit some spatial relationships: adjacency, proximity, connectivity and the nearest neighbor. Besides, we exploit the overlaying operation. Overlay analysis is the process of superimposing some layers, such that the resultant map contains the data from all maps for the selected features.

By considering all these spatial relations, many scenarios can occur. In the sequel some scenarios are reported.

For a GIS user looking for information about the river contamination state. Whenever, the user fails to found such information in the documents relative to the river, it will be interesting to examine the connected rivers. The state of the connected branches may be worth informing about the river state. Hence, we enlarge our investigation space to look for information through documents relative to the related entities according to the connectivity topological relation. By the way, the GIS user can deduce the decision about the river state or the measurement to undertake in order to prevent eventual contamination.

Another spatial use case is to profit from the overlay analysis. Assume that a GIS user is handling a GDB relative to birds' dispersion all over the world, and that our data enrichment process is not satisfactory enough for one of the reasons abovementioned. The user can refine the results by overlaying the layer of birds' dispersion and of the countries to define the area where both inputs overlap and retains a set of attributes for each. The points of intersection are the keys of the search that are likely to return the relevant documents. In other words, the couple of identifiers relative to the bird and the country contribute as the components of the key used for the IR, which lead to better locate the entity and can enhance the results.

IMPLEMENTATION

To perform the data enrichment we have developed the SDET (*Semantic Data Enrichment*) tool (Mahmoudi & Faiz, 2007, Mahmoudi & Faiz, 2006$_c$). The implementation was performed using Java language to comply with our distributed architecture. The SDET functionalities have been integrated into OpenJUMP GIS. The SDET tool provides the GIS user with means to manage textual corpora relative to geographic entities. The document management ranges from text segmentation and theme identification to text condensation (figure 3.a). Besides, our tool offers a refinement facility, triggered in demand. One example is shown in figure 3.b relative to the exploitation of the adjacency relation. In this situation, a user is looking for information about Uganda and the relationships maintained with its neighbors that he ignores. Such information is not stored in our GDB relative to the world countries. By exploiting the adjacency relation, the system highlights graphically the neighboring countries meanwhile provides the attributes that will serve as part of the search key. Besides, to guide more the GIS user, we report the statistics relative to the number of web documents reporting information of the association of each neighbor with Uganda. Hence, the user can be adverted about the amount of data to handle and by the way we assist him in choosing the geographic entities attributes that are likely to lead to relevant documents.

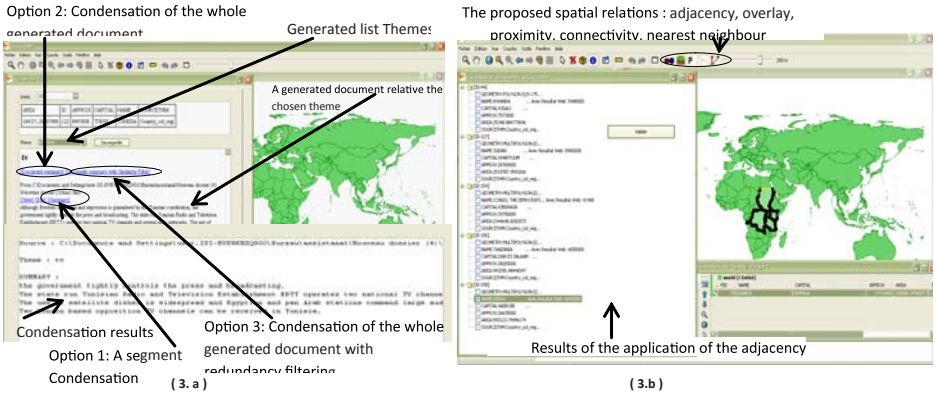
EVALUATION

To evaluate our enrichment system, we have used ROUGE (Recall-Oriented Understudy for Gisting Evaluation) as an intrinsic summary evaluation tool. ROUGE was defined by the DUC (Document Understanding Conference) community (organized by NIST; National Institutes of Standards and Technology (USA)) dedicated to the summary systems evaluation.

ROUGE (Lin, 2004) includes measures to automatically determine the quality of a summary by comparing it to other (ideal) summaries created by humans by using the n-grams. ROUGE-N is an n-gram recall between a candidate summary and a set of reference summaries. ROUGE-N is computed as follows:

$$\text{ROUGE-N} = \frac{\sum_{s \in \{references\}} \sum_{n-gram \in s} count_{match}(n-gram)}{\sum_{s \in \{references\}} \sum_{n-gram \in s} count(n-gram)}$$

Figure 3. The overall enrichment process (3.a) - the refinement process (3.b)



Where *n* stands for the length of the *n-gram* and $Count_{match}(n\text{-}gram)$ is the number of *n-grams* co-occurring in a candidate summary and a set of reference summaries.

Rouge-2, Rouge-SU-4 and BE have been imposed during DUC'2007. ROUGE-2 is the number of successive word pairs (number of *bigrams)* en common between system summary and the models. ROUGE-SU-4; corresponds to recall in skip units with maximum size 4. BE have been proposed as an extension to the basic ROUGE measure. In this approach, we break down each reference (and system) sentence into a set of minimal semantic units called *Basic Elements* (BE). BE are word pairs with their relation labelled as subject, object... For example, in the following sentence "*two Libyans were indicted for the Lockerbie bombing in 1991"* we depict the BE: *indicted|libyans|obj*, where *obj* is the relation *object* between *indicted* and *libyans.* The references and the system summary are compared in terms of BE.

In the manner of DUC, we have defined two *baseline* references. These are summaries created automatically in terms of the following rules: one *baseline* selects the 150 first words from the most recent document in the corpus, another *baseline* selects the first sentences within the (1, …, n) documents of the collection sorted by the chronological order till reaching the 150 words. These rules stem from textual documents studies which has approved the importance of the document beginning in comparison with its end. Hence, the text first lines cover generally the most important part of the gist of the text.

To perform our evaluation, we have generated summaries using: the system MEAD (Radev et al., 2003) and the *baseline* methods (*baseline1, baseline2*). We have compared with ROUGE the summaries obtained with SDET and those generated using the other systems with the human summaries (see table 1).

A high score is the best and it is an indicator of the most relevant system. SDET is classified at the first position with the best evaluation scores. The two baselines bring the lower scores. In fact, while the hypothesis behind the two baselines is always or quasi-always valid, it is not sufficient for a multi-document task which needs a special care of the similarities and differences through out the documents corpus.

Table 1. Evaluation results using the package ROUGE/BE

Système	Rouge-2	Rouge-SU4	BE
SDET	0,47908	0,30106	0,23048
MEAD	0,35194	0,20117	0,10580
Baseline1	0,20200	0,12132	0,09851
Baseline2	0,17240	0,08104	0,05141

FUTURE TRENDS

Many extensions can be performed to pursue the development of a more suitable approach to the GIS user expectations. As one perspective, we can cite the summary updating issues. In other words, we will focus on conceiving means that can take into account the new information resulting from an updating of the existing documents or the newly coming documents without generating summaries from scratch.

CONCLUSION

With the increasing demand for the geographical information expressed by the GIS users, the data enrichment processes play a key role to extend the dataset already stored within the system.

In this context, we propose an approach to provide complementary data that enrich the descriptive aspect of the GDB.

Along this paper we have detailed our semantic data enrichment approach. Furthermore, a refinement process was presented. It is provided to GIS users as a mean to describe with more accuracy the geographic entities and by the way to increase the chance to reach the pertinent documents.

The main SDET tool functionalities as well as their integration to OpenJUMP GIS, were presented also. Finally, the evaluation of the summaries generated by our system was detailed. The evaluation has show that our system is satisfactory enough.

REFERENCES

Barzilay, R., & McKeown, K. (2005). Sentence fusion for multidocument news summarization. *Computational Linguistics, 31*(3), 297-328.

Bin, J., & Itzhak, O. (2006). Spatial topology and its structural analysis based on the concept of simplicial complex. *9th AGILE Conference on Geographic Information Science*, Visegrád, Hungary (pp. 204-212).

Choi, F. (2000). Advances in domain independent linear text segmentation. In *NAACL'00*.

David, A. S. (2002). Detecting events with date and place information in unstructured text. *In Proceedings of the 2nd ACM+IEEE Joint Conference on Digital Libraries*, Portland, OR, (pp. 191-196).

Faïz, S. (1999). Systèmes d'informations géographiques : Information qualité et datamining. *Editions C.L.E,* 362.

Faïz, S., & Mahmoudi, K. (2005, September). Semantic enrichement of geohraphical databases. In L. Rivero, J. Doorn, V. et Ferraggine (Eds.) *Encyclopedia of database technologies and applications*, Idea Group, Etats-Unis, (pp. 587-592).

Ferber, J. (1999). *Multi-agent systems: An introduction to distributed artificial intelligence.* 1st ed. Addison-Wesley Professional.

Hearst, M. A. (1997). TextTiling: Segmenting text into multi-paragraph subtopic passages. *Computational linguistics, 23*(1), 33-46.

Hyland, R., Clifton, C., & Holland, R. (1999). GeoNODE: Visualizing news in geospatial environments. *In Proceedings of the Federal Data Mining Symposium and Exhibition '99, AFCEA*, Washington D.C.

Lin, C-Y. (2004). ROUGE: A package for automatic evaluation of summaries. *Proceedings of Workshop on Text Summarization Branches Out, Post-Conference Workshop of ACL,* Barcelona, Spain, 74-81.

Mahmoudi, K., & Faïz, S. (2006_a). *Une approche distribuée pour l'extraction de connaissances: Application à l'enrichissement de l'aspect factuel des BDG.* Revue des Nouvelles Technologies de l'Information, Editions Cépaduès, Janvier, (pp. 107-118).

Mahmoudi, K., & Faïz, S. (2006_b). L'apport de l'information spatiale pour l'enrichissement des bases de données. INFORSID'2006, Hammamet, Tunisie, juin, (pp. 323-338).

Mahmoudi, K., & Faïz, S. (2007). L'outil SDET pour le complètement des données descriptives liées aux bases de données géographiques. Actes 7èmes Journées d'Extraction et Gestion des Connaissances (EGC'07), Session Demo, Namur Belgique, Janvier, (pp. 179-180).

Mahmoudi, K., & Faïz, S. (2006_c). SDET: A semantic data enrichment tool application to geographical databases. *International Conference On Signal-Image Technology & Internet–Based Systems (SITIS'2006), IEEE, ACM*, Hammamet, Tunisie, Décembre, (pp. 88- 97).

Mann, W. C., & Thompson, S. A. (1988). Rhetorical structure theory: Toward a functional theory of text organization. *An Interdisciplinary Journal for the Study of Text 8*(2), 243-281.

Marcu, D. (1999). Discourse trees are good indicators of importance in text. Mani and Maybury (eds.), *Advances in Automatic Text Summarization*, the MIT Press, (pp. 123-136).

Miller, G. (1990). Wordnet: An on-line lexical database. *International Journal of Lexicogra-phy (special issue), 3*(4), 235-312.

MetaCarta (2005): GTS versus MetaCarta GeoTagger. MetaCarta, Inc.

Plazanet, C. (1996). *Enrichissement des bases de données géographiques: analyse de la géométrie des objets linéaires pour la généralisation cartographique (application aux routes).* PhD thesis, Université de Marne-la-Vallée.

Radev D., Qi, J., Otterbacher, H., & Tam (2003). *Mead reducs: Michigan at DUC 2003.* Edmonton, Alberta, Canada: ACL, (pp. 160-167).

KEY TERMS

GIS: Geographic Information System (GIS) is a computer system capable of capturing, storing, analyzing, and displaying geographically referenced information. These latter are stored in a Geographic database (GDB).

MDS: Multi-Document Summarization (MDS) is the process of distilling the most important information from a corpus of documents.

Rhetorical Structure Tree: A binary tree, which describes the rhetorical structure of every coherent discourse.

Segmentation: Text segmentation is the process of segmenting a text stream into topically coherent segments.

Chapter XXXII
Similarity Search in Time Series

Maria Kontaki
Aristotle University, Greece

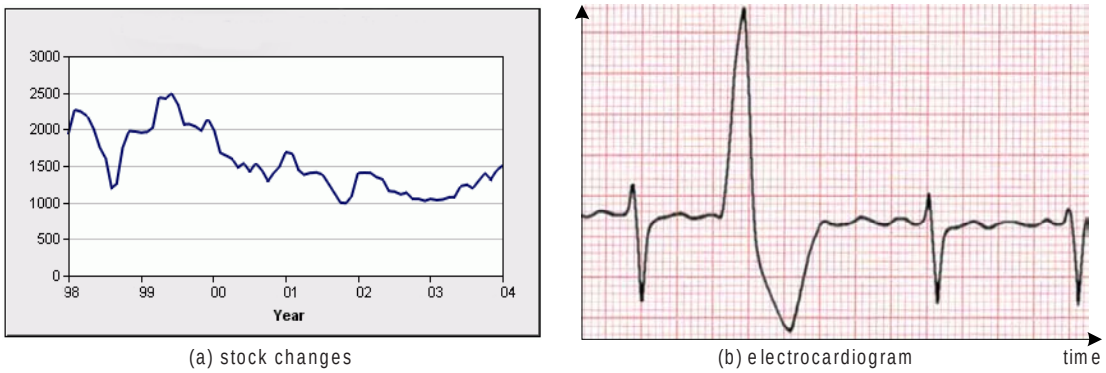
Apostolos N. Papadopoulos
Aristotle University, Greece

Yannis Manolopoulos
Aristotle University, Greece

INTRODUCTION

In many application domains, data are represented as a series of values in different time instances (*time series*). Examples include stocks, seismic signals, audio, and so forth. Similarity search in time series databases is an important research direction. Several methods have been proposed to provide efficient query processing in the case of static time series of fixed length. Research in this field has focused on the development of effective transformation techniques, the application of dimensionality reduction methods, and the design of efficient *indexing schemes*. These tools enable the process of *similarity queries* in time series databases. In the case where time series are continuously updated with new values (*streaming time series*), the similarity problem becomes even more difficult to solve, since we must take into consideration the new values of the series. The dynamic nature of streaming time series precludes the use of methods proposed for the static case. To attack the problem, significant research has been performed towards the development of effective and efficient methods for streaming time series processing. In this article, we introduce the most important issues concerning similarity search in static and streaming time series databases, presenting fundamental concepts and techniques that have been proposed by the research community.

Figure 1. Examples of time series data



BACKGROUND

Time series are used in a broad range of applications, modeling data that change over time. For example, stock changes, audio signals, seismic signals, electrocardiograms, can be represented as time series data. In fact, any measurement that changes over time can be represented as a time series. Two simple time series examples are depicted in Figure 1.

We differentiate between two types of time series, namely: 1) *static time series*, and 2) *streaming time series*. In the first case, we assume that the time series is composed of a finite number of sample values, whereas in the second case the size of the series is increasing since new values are appended. For example, if the data correspond to stock prices for the year 2004, then we can use static time series to capture the stock prices of the time period of interest. On the other hand, if there is a need for continuous stock monitoring as time progresses, then streaming time series are more appropriate.

Streaming time series is a special case of streaming data, which nowadays are considered very important and there is an increasing research interest in the area. Traditional database methods can not be applied directly to data streams. Therefore, new techniques and algorithms are required to guarantee efficient and query processing in terms of the CPU time and the number of disk accesses. The most important difficulty that these techniques must address is the continuous change, which poses serious restrictions.

The purpose of a time series database is to organize the time series in such a way that user queries can be answered efficiently. Although user queries may vary according to the application, there are some fundamental query types that are supported:

- *whole-match queries*, where all time series have the same length, and
- *subsequence-match queries*, where the user's time series is smaller than the time series in the database, and therefore we are interested in time series which contain the user's time series.

In contrast to traditional database systems, time series databases may contain erroneous or noisy data. This means that the probability that two time series have exactly the same values in the same time instances is very small. In such a case, *exact search* is not very useful, and therefore *similarity search* is more appropriate. There are three basic types of similarity queries:

- *similarity range query*: given a user time series *Q* and a distance *e*, this query retrieves all time series that are within distance *e* from *Q*.
- *similarity nearest-neighbor query*: given a user time series *Q* and a integer *k*, this query retrieves the *k* series that are closer to *Q*.
- *similarity join query*: given two sets of times series *U*,*V* and a distance *e*, this query retrieves all pairs (*u*,*v*) $u \in U$ and $v \in V$ such that the distance between *u* and *v* is less or equal to *e*.

It is evident from the above definitions, that in order to express the similarity between two time series, a distance measure *D* is required. The values of the distance measure (also termed *dissimilarity measure*) usually range between 0 and 1. If two time series *u* and *v* are similar, then the value *D*(*u*,*v*) should be close to 0, whereas if they are dissimilar *D*(*u*,*v*) should be close to 1. Similarity search can be applied for whole-match queries and subsequence-match queries as well, for static or streaming time series.

SIMILARITY SEARCH IN TIME SERIES

We begin our study with methods proposed for static time series. Streaming time series are considered later in this section. The efficient processing of similarity queries requires the addressing of the following important issues:

- the definition of a meaningful distance measure *D* in order to express the dissimilarity between two time series objects,
- the efficient representation of time series data, and
- the application of an appropriate indexing scheme in order to quickly discard database objects that can not contribute to the final answer.

Assuming that each time series has a length of *n*, then it is natural to think that each time series is represented as a vector in the *n*-dimensional space. In such a case, the similarity between two time series *u* and *v* can be expressed as the *Euclidean* distance:

$$D(u,v) = \sqrt{\sum_{i=1}^{n} (u[i] - v[i])^2}$$

where *u*[*i*], *v*[*i*] is the value of *u* and *v* for the *i*-th time instance. The Euclidean distance has been widely used as a dissimilarity measure in time series literature (Agrawal 1993, Faloutsos 1994, Chan 1999, Kontaki 2004), because of its simplicity. The Euclidean distance is a member of L_p norm, one of the most popular families of distance functions (Agrawal 1995, Yi 2000). L_p norm is defined as follows:

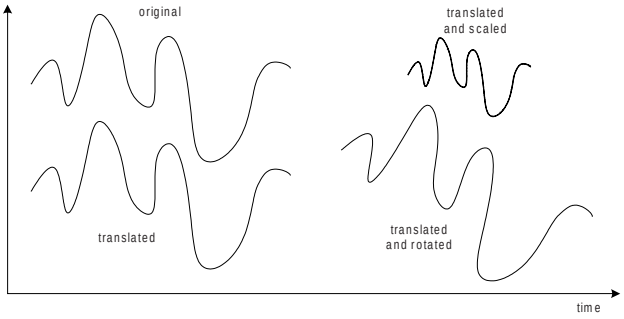
$$L_p(u,v) = \left(\sum_{i=1}^{n} | u[i] - v[i] |^p \right)^{\frac{1}{p}}$$

It is known as city-block or Manhattan norm when p=1 and Euclidean norm when p=2. In the case when $p = \infty$, it is called maximum norm.

Several alternative distance functions have been proposed to allow translation, rotation and scaling invariance. Consider for example the time series depicted in Figure 2. Note that although all time series have the same shape, they will be considered different if the Euclidean distance is used to express similarity. Translation, rotation and scaling invariance is studied in (Agrawal 1995, Yi 1998, Chan 1999, Yi 2000).

The main shortcoming of the Euclidean distance is that all time series must be of equal length, which is a significant restriction. If time series are sampled using different time intervals, then their length will not be the same, and therefore the Euclidean distance can not be applied. In order to express similarity between time series of different lengths, other more sophisticated

Figure 2. Examples of translation, rotation and scaling of time series



distance measures have been proposed (Yi 1998, Park 2000). One such distance measure is *Time Warping* (TW) that allows time series to be stretched along the time axis. The time warping distance maps each element of a time series *u* to one or more elements of another time series *v*. Given two time series *u* and *v* the time warping distance $D_{TW}(u,v)$ is defined as follows (several variations have been proposed):

$$D_{TW}(u,v) = |u[1]-v[1]| + \min \begin{cases} D_{TW}(u, v[2:*]) \\ D_{TW}(u[2:*], v) \\ D_{TW}(u[2:*], v[2:*]) \end{cases}$$

where *u*[2:*] and *v*[2:*] denote the suffix of *u* and *v* respectively. The TW distance has been used extensively in pattern matching, for voice, audio and other types of signals (e.g., electrocardiograms). The computation cost of the TW is much higher than that of the Euclidean distance. The TW distance is not metric and thus it is not indexable. To overcome this drawback, lower bounds of the TW distance have been proposed that obey the *triangular inequality* (Park 2000).

Many string similarity functions such as the *Edit Distance* (Bozkaya 1997) and the *Longest Common Subsequence* (LCSS) (Vlachos 2002) have been modified in order to match similar time series. These similarity functions can be used in time series with different lengths or different sampling rates because some elements may be unmatched. In (Chen 2004), the *Edit Distance with Real Penalty* (ERP) was introduced and the distance function uses a constant gap of edit distance and real distances between elements as the penalty to handle local time shifting. In (Chen 2005), the *Edit Distance on Real sequence* (EDR), a modification of Edit Distance, has been proposed for fast similarity between moving objects trajectories.

As we will discuss next, an important issue is the representation of time series. Experimental evaluation shows that the efficiency of similarity functions highly depends on the representation of time series. Thus, new improved distance functions are proposed for specific representations. The *distPLA* distance is introduced in (Chen 2007) and is appropriate for PLA (*Piecewise Linear Approximation*) representation. Table 1 lists five popular distance functions. For each function, the table lists the time complexity, the capability to handle local time shifting and if it respects the metric space properties or not.

The number of samples of each time series may range from a few to hundreds or thousands. Therefore, representing each time series as an *n*-dimensional vector may result in performance degradation during query processing, due to high computation costs of the distance function. It is preferable to compute the distance as efficiently as possible, towards increased processing perfor-

Table 1. Popular distance functions

Distance Function	Local Time Shifting	Metric	Time Complexity
L_p norm	No	Yes	$O(n)$
TW	Yes	No	$O(n^2)$
LCSS	Yes	No	$O(n^2)$
ERP	Yes	Yes	$O(n^2)$
EDR	Yes	Yes	$O(n^2)$

mance. To attack this problem, *dimensionality reduction* is applied to the time series, to transform them to a more manageable representation. Figure 3 illustrates an example.

One of the first dimensionality reduction techniques applied to time series is the *Discrete Fourier Transform* (DFT), which transforms a time series to the frequency domain. For many real time series (e.g. stock changes) it is observed that most of the information is concentrated in the first few DFT coefficients. Therefore, each time series can be represented by using only a few real numbers, instead of using all the values in the time domain. DFT has been successfully used for whole-match and subsequence-match in static or streaming time series (Agrawal 1993, Faloutsos 1994, Yi 2000, Kontaki 2004).

Time series have many fluctuations, noise and outliers, thus the efficiency of the similarity functions can be reduced dramatically. In order to smooth time series, *piecewise techniques* have been used. The underlying idea is to represent time series using an abstractive representation that is more simple and closer to the "human sense". The most popular representation is the *Piecewise Linear Approximation* (PLA) which approximates a time series with line segments. Given a sequence S of length n, PLA can use one line segment, $S' = a \cdot t + b$ ($t \in [1, n]$), to approximate S, where a and b are two coefficients in a linear function such that the reconstruction error of S is minimized. A different technique has been proposed in (Chakrabarti 2002), the *Adaptive Piecewise Constant Approximation* (APCA). More specifically, APCA divides the time series into disjoint segments of different lengths and computes the mean of each segment. Thus, each segment can be represented by two reduced coefficients, the mean value and the length of the segment.

Among other dimensionality reduction techniques we note the *Singular Value Decomposition* (SVD), the *Discrete Wavelet Transform* (DWT), *FastMap, Piecewise Aggregate Approximation* (PAA) and *Chebyshev Polynomians* (CP), which have been successfully applied in time series databases. Table 2 reports seven popular dimensionality reduction techniques in terms of the time complexity, space complexity and the capability to be indexed. The length of each time series is n, the total number of time series in database is N and m is the reduced dimensionality.

Figure 3. Applying dimensionality reduction using transformation

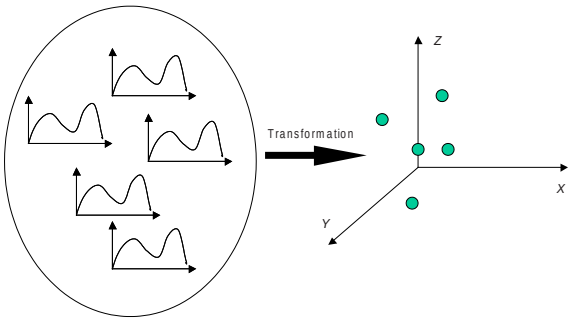


Table 2. Popular dimensionality reduction techniques

Dimensionality Reduction Technique	Indexable	Time Complexity	Space Complexity
DFT	Yes	$O(n \cdot log(n))$	$O(n)$
SVD	Yes	$O(N \cdot n^2)$	$O(N \cdot n)$
DWT	Yes	$O(n)$	$O(n)$
PLA	No	$O(n)$	$O(n)$
APCA	Yes	$O(n)$	$O(n)$
PAA	Yes	$O(n)$	$O(n)$
CP	Yes	$O(m \cdot n)$	$O(n)$

Note that dimensionality reduction is a lossy operation, since after the transformation of the time series some information is lost. This means that the transformed data is an approximation of the original data, and the latter must be retained to be available for reference. The original and the transformed data are used for query processing by means of the *filter-refinement* processing technique:

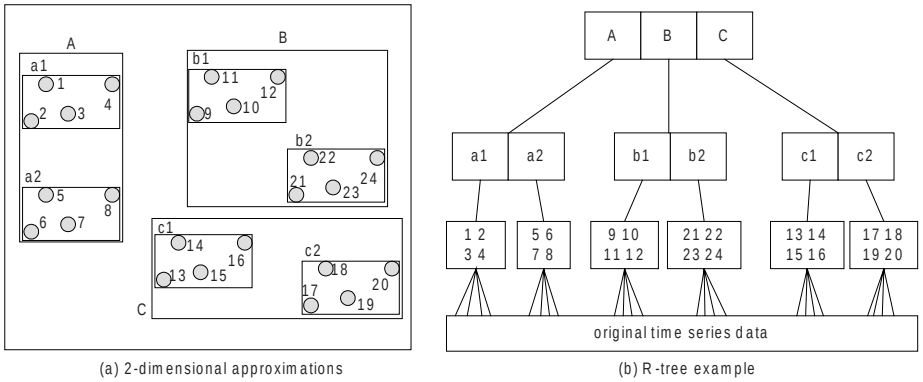
- During the *filter step*, the approximations (transformed data) are investigated to quickly discard time series that can not contribute to the final answer. The result of this step is a set of *candidate* time series that may be part of the final answer.
- Candidates are investigated further in the *refinement step*, by accessing the original time series in the time domain. Candidate objects that do not satisfy query constraints are characterized as *false alarms*, and are discarded.

It is important that the filter step be very efficient with respect to the processing performance and the number of candidates determined. The number of candidates depends on the transformation technique used to produce the approximations, whereas the processing performance depends heavily on the indexing scheme applied. If the time series approximations are vectors in a multi-dimensional space, then *spatial access methods* can be used to organize data hierarchically. One of the most influential spatial access methods is the R-tree (Guttman 1984) and the R*-tree (Bechmann 1990), which is one of its successful variations. The multi-dimensional approximations are organized in an efficient way, to speed-up query processing. The R*-tree manages to discard quickly a large portion of the database, and therefore helps significantly in the efficient processing of the filter step. Efficient algorithms have been proposed for similarity range search, similarity nearest-neighbor search and similarity join. An R*-tree example for a 2-dimensional point dataset is illustrated in Figure 4.

If the dimensionality of the transformed data is still large after the application of the dimensionality reduction method, then indexing schemes for high-dimensional data can be used. These schemes are influenced by the R*-tree access method and they use several optimization techniques to attack the dimensionality curse problem. Among these sophisticated access methods we highlight the TV-tree (Lin 1995) and the X-tree (Berchtold 1996).

Moreover, new index structures are designed to be appropriate specifically for similarity search applications. In (Hoyle 2005), the SASH (*Spatial Approximation Sample Hierarchy*) index is introduced, for approximate similarity queries in high-dimensional datasets. The SASH index is

Figure 4. An R-tree example for 2-dimensional approximations*



(a) 2-dimensional approximations (b) R-tree example

a multi-level structure of random samples. Each remaining object is connected to several of their approximate nearest neighbors from within the sample. Recently the multi-probe LSH index has been proposed (Lv 2007), improving the *locality sensitive hashing* (LSH) indexing techniques.

A whole-match query requires that the query time series and the time series in the database have the same length. This restriction is not realistic for some applications: 1) A query posed by the user is usually smaller than the time series in the database and 2) the *sampling rate* can be different for each sequence. Thus, subsequence match queries, that are generalization of the whole-match queries, are more appropriate for many applications.

There are several approaches in the literature that attack the subsequence matching problem. In (Faloutsos 1994), the proposed scheme for whole match queries of (Agrawal 1993) was extended. The method uses a sliding window over the data sequence, maps each window to the frequency domain, and keeps the first few coefficients. Thus, a data sequence is mapped into a trail in the frequency domain. The trails are divided into subtrails, which are stored in R*-tree. The drawback of this method is that the length of the query cannot be smaller than the sliding window size. Alternatively, other approaches use L_p norm, DTW and LCSS as distance functions, PLA and APCA as representation of the time series, R-tree and suffix-tree as index structure (Keogh 1999, Park 2000, Yi 2000, Vlachos 2003).

So far we have focused on 1-dimensional time series, since there is only one measurement that changes over time. However, there are applications that require the manipulation of *multi-dimensional* time series. For example, consider an object that changes location and we are interested in tracking its position. Assuming that the object moves in the 2-dimensional space, there are two values (x and y coordinates) that change over time. Some interesting research proposals for multi-dimensional time series can be found in (Vlachos 2002, Vlachos 2003, Chen 2005).

Although a significant amount of research work has been performed for static time series, the field of streaming time series is quite immature, since the data stream model has been recently introduced (Babou 2001, Babcock 2002, Gilbert 2003). The difficulty in a streaming time series database is that new values for the time series are continuously arrive. This characteristic yields existing techniques for static time series inefficient, because the index must be updated continuously for every new value.

Similarity queries in streaming time series have been studied in (Gao 2002) where whole-match queries are investigated by using the Euclidean distance as the similarity measure. A prediction-based approach is used for query processing. The distances between the query and each data stream are calculated using the predicted values. When the actual values of the query are available, the upper and lower bound of the prediction error are calculated and the candidate set is formed using the predicted distances. Then, false alarms are discarded. The same authors have proposed two different approaches, based on pre-fetching (Gao 2002b, Gao 2002c).

The aforementioned research efforts examine the case of whole-match queries, where the data are static time series and the query is a streaming time series. In (Liu 2003) the authors present a method for query processing in streaming time series where both the query object and the data are streaming time series. The VA-stream and VA+-stream access methods have been proposed, which are variations of the VA-file (Weber 1998). These structures are able to generate a summarization of the data and enable the incremental update of the structure every time a new value arrives. The performance of this approach is highly dependent on the number of bits associated with each dimension.

In (Kontaki 2004, Kontaki 2007) a different approach is followed to provide a flexible technique for similarity range queries when both data and queries are streaming time series. The proposed method (IDC-Index) is based on the R-tree which is used as the indexing scheme for the underlying time series approximations. The dimensionality reduction technique applied to the original time series is based on an incremental computation of the DFT which avoids re-computation. Moreover, the R-tree is equipped by various deferred update policies to avoid index adjustments every time a new value for a streaming time series is available. Experiments performed on synthetic random walk time series and on real time series data have shown that the proposed approach is very efficient in comparison to previously proposed methods.

A novel representation, called *Multi-scaled Segment Mean* (MSM), appropriate for streaming time series has been proposed in (Lian 2007). A time series is divided into disjoined segments, each of which is represented by the mean of its values. The representation consists of multiple levels in which the number of segments varies. High-level representation is more abstractive, so the number of segments is low. Conversely, low-level representation has a large number of segments. The MSM representation can be computed incrementally and therefore it is suitable for streaming environments.

APPLICATIONS

The retrieval of similar time series has wide usage in many new application domains such as:

- **Financial data analysis:** Similarity search can be used in stock market data to discover stocks with certain shapes or user-defined movement trends. Therefore, analysts can have an indication of a stock's future performance. A typical query of this domain would be 'find other companies which have similar sales patterns with this company'.
- **Moving object tracking:** Similarity search can be used to discover patterns between trajectories. For example, by the examination of the trajectories of animals, it is possible to determine migration patterns. In a store monitoring system, finding the customers' movement may help in the arrangement of merchandise.
- **Sensor network monitoring:** Event detection is one of the most important applications of similarity search in sensor networks. For example, assume a building with a network of sensors collecting data, such as temperature. Since emergency events (e.g. fire

alarm) usually correspond to some specific patterns, similarity search helps to discover emergency situations accurate and fast.

- **Scientific databases:** Other examples of databases, that similarity search has important contribution in data analysis, are: geological, environmental, astrophysics databases, weather data, biological data (e.g. diagnosis of pre-defined patterns of an electrocardiogram).

FUTURE TRENDS

The research interest in the last years has focused to the streaming time series. Apart from the investigation of more efficient techniques for similarity search, there is significant work performed towards *data mining* of streaming data. The challenge is to overcome the difficulty of continuous data change, and apply *clustering* algorithms to streaming time series. Some interesting results have been reported (Guha 2003).

Another important research direction is the management of *continuous queries* over streaming time series. In this case users pose queries that must be continuously evaluated for a time interval. Therefore, when a new value for a time series arrives, the value must be used to determine which queries are satisfied. Continuous queries over data streams are studied in (Chandrasekaran 2002, Gao 2002, Gao 2002b, Gao 2002c, Babcock 2002).

A new promising direction for similarity search is the *approximate similarity queries*. To speed up query processing, the method sacrifices the quality of the answer. There is a tradeoff between the processing time and the quality. Approximate similarity queries have been studied in (Houle 2005, Lv 2007). Additionally, new types of queries will be proposed for specific application domains. Recently, two novel types of similarities queries have been proposed, the *threshold query* (Assfalg 2006) and the *k-n-match query* (Tung 2006).

CONCLUSION

Time series data are used to model values that change over time, and they are successfully applied to diverse fields such as online stock analysis, computer network monitoring, network traffic management, seismic wave analysis. To manipulate time series efficiently, sophisticated processing tools are required from the database viewpoint.

Time series are categorized as static or streaming. In static time series each sequence has a static length, whereas in the streaming case new values are continuously appended. This dynamic characteristic of streaming time series poses significant difficulties and challenges in storage and query processing.

A fundamental operation in a time series database system is the processing of similarity queries. To achieve this goal, there is a need for a distance function, an appropriate representation and an efficient indexing scheme. By using the filter-refinement processing technique, similarity range queries, similarity nearest-neighbor queries and similarity join queries can be answered efficiently.

REFERENCES

Agrawal, R., Faloutsos, C., & Swami, A. (1993). Efficient similarity search in sequence databases. *Proceedings of FODO*, 69-84. Evanston, Illinois, USA.

Agrawal, R., Lin, K.-I., Sawhney, H. S., & Swim, K. (1995). Fast similarity search in the presence of noise, scaling, and translation in time-series databases. *Proceedings of VLDB*, 490-501. Zurich, Switzerland.

Assfalg, J., Kriegel, H., Kroger, P., Kunath, P., Pryakhin, A., & Renz, M. (2006). Threshold similarity queries in large time series databases.

Proceedings of IEEE ICDE, 149. Atlanta, Georgia, USA.

Babcock, B., Babu, S., Datar, M., Motwani, R., & Widom, J. (2002). Models and issues in data stream systems. *Proceedings of ACM PODS,* 1-16. Madison, Wisconsin, USA.

Berchtold, S., Keim, D., & Kriegel H.-P. (1996). The X-tree: An index structure for high-dimensional data. *Proceedings of VLDB,* 28-39. Bombay, India.

Beckmann, N., Kriegel, H.-P., Schneider, R., & Seeger, B. (1990). The R*-tree: An efficient and robust access method for points and rectangles. *Proceedings of ACM SIGMOD,* 322-331. Atlantic City, NJ.

Babu, S., & Widom, J. (2001). Continuous queries over data streams. *SIGMOD Record, 30*(3), 109-120.

Bozkaya, T., Yazdani, N., & Ozsoyoglu, M. (1997). Matching and indexing sequences of different lengths. *Proceedings of CIKM*, 128-135. Las Vegas, NV, USA.

Chakrabarti, K., Keogh, E., Mehrotra, S., & Pazzani, M. (2002). Locally adaptive dimensionality reduction for indexing large time series databases. *ACM TODS, 27*(2), 188-228.

Chan, K., & Fu, A. W. (1999). Efficient time series matching by wavelets. *Proceedings of IEEE ICDE*, 126-133. Sydney, Australia.

Chandrasekaran, S., & Franklin, M. J. (2002). Streaming queries over streaming data. *Proceedings of VLDB*, 203-214. Hong Kong, China.

Chen, L., Özsu, M. T., & Oria, V. (2005). Robust and fast similarity search for moving object trajectories. *Proceedings of ACM SIGMOD,* 491-502. Baltimore, Maryland, USA.

Chen, L., & Ng, R. (2004). On the marriage of edit distance and Lp norms. *Proceedings of VLDB,* 792-803. Toronto, Ontario, Canada.

Chen, Q., Chen, L., Lian, X., Liu, Y., & Yu, J. X. (2007). Indexable PLA for efficient similarity search. *Proceedings of VLDB*, 453-446. Vienna, Austria.

Faloutsos, C., Ranganathan, M., & Manolopoulos, Y. (1994). Fast subsequence matching in time-series databases. *Proceedings of ACM SIGMOD,* 419-429. Minneapolis, Minnesota, USA.

Gao, L., & Wang, X. S.(2002). Continually evaluating similarity-based pattern queries on a streaming time series. *Proceedings of ACM SIGMOD,* 370-381. Madison, Wisconsin, USA.

Gao, L., & Wang, X. S. (2002b). Improving the performance of continuous queries on fast data streams: Time series case. *SIGMOD/DMKD Workshop.* Madison, Wisconsin, USA.

Gao, L., Yao, Z., & Wang, X. S. (2002c). Evaluating continuous nearest neighbor queries for streaming time series via pre-fetching *Proceedings of CIKM,* 485-492. McLean, Virginia, USA.

Gilbert, A. C., Kotidis, Y., Muthukrishnan, S., & Strauss, M. J. (2003). One-pass wavelet decompositions of data streams. *IEEE TKDE, 15*(3), 541-554.

Guha, S., Meyerson, A., Mishra, N., Motwani, R., & O'Callaghan, L. (2003). Clustering data streams: Theory and practice. *IEEE TKDE, 15*(3), 515-528.

Houle, M. E. & Sakuma, J. (2005). Fast approximate similarity search in extremely high-dimensional data sets. *Proceedings of IEEE ICDE,* 619-630, Tokyo, Japan.

Keogh, E. J., & Pazzani, M. J. (1999). An indexing scheme for fast similarity search in large time series databases. *Proceedings of SSDBM,* 56-67, Clevelant, Ohio, USA.

Kontaki, M., & Papadopoulos, A. N. (2004). Efficient similarity search in streaming time sequences. *Proceedings of SSDBM,* 63-72, Santorini, Greece.

Kontaki, M., Papadopoulos, A. N., & Manolopoulos, Y. (2007). Adaptive similarity search in streaming time series with sliding window. *DKE* to appear.

Lian, X., Chen, L., Yu, J. X., Wang, G., & Yu, G. (2007). Similarity match over high speed time-series streams. *Proceedings of IEEE ICDE*, 1086-1095. Istanbul, Turkey.

Lin, K., Jagadish, H. V., & Faloutsos, C. (1994). The TV-tree: An index structure for high dimensional data. *The VLDB Journal, 3*(4), 517-542.

Liu, X., & Ferhatosmanoglu, H. (2003). Efficient k-NN search on streaming data series. *Proceedings of SSTD*, 83-101. Santorini, Greece.

Lv, Q., Josephson, W., Wang, Z., Charikar, M., & Li, K. (2007). Multi-probe LSH: Efficient indexing for high-dimensional similarity search. *Proceedings of VLDB*, 950-961. Vienna, Austria.

Park, S., Chu, W. W., Yoon, J., & Hsu, C. (2000). Efficient searches for similar subsequences of different lengths in sequence databases. *Proceedings of IEEE ICDE*, 23-32. San Diego, CA, USA.

Tung, A. K., Zhang, R., Koudas, N., & Ooi, B. C. (2006). Similarity search: A matching-based approach. *Proceedings of VLDB*, 631-642. Seoul, Korea.

Vlachos, M., Hatjieleftheriou, M., Gunopoulos, D., & Keogh, E. (2003). Indexing multidimensional time series with support for multiple distance measures. *Proceedings of ACM KDD*, 216-225. Washington, DC, USA.

Vlachos, M., Kollios, G., & Gunopoulos, D. (2002). Discovering similar multidimensional trajectories. *Proceedings of IEEE ICDE*, 673-684. San Jose, California, USA.

Weber, R., Schek, H.-J., & Blott, S. (1998). A quantitative analysis and performance study for similarity-search methods in high-dimensional spaces. *Proceedings of VLDB*, 194-205. New York City, USA.

Yi, B.-K., & Faloutsos, C. (2000). Fast time sequence indexing for arbitrary Lp norms. *Proceedings of VLDB*, 385-394. Cairo, Egypt.

Yi, B.-K., Jagadish, H. V., & Faloutsos, C. (1998). Efficient retrieval of similar time sequences under time wrapping. *Proceedings of IEEE ICDE*, 201-208. Orlando, FL, USA.

KEY TERMS

Data Mining: A research field which investigates the extraction of useful knowledge from large datasets. Clustering and association rule mining are two examples of data mining techniques.

Dimensionality Reduction: It is a technique that is used to lower the dimensionality of the original dataset. Each object is transformed to another object which is described by less information. It is very useful for indexing purposes, since it increases the speed of the filtering step.

Distance Function: It is used to express the similarity between two objects. It is usually normalized in the range between 0 to 1. Examples of distance functions used for time series data are the Euclidean distance and the Time Warping distance.

Filter-Refinement Processing: A technique used in query processing, which is composed of the filter step and the refinement step. The filter step discards parts of the database that can not contribute to the final answer, and determines a set of candidate objects, which are then processed by the refinement step. Filtering is usually enhanced by efficient indexing schemes for improved performance.

Similarity Queries: These are queries that retrieve objects which are similar to a query object. There are three basic similarity query

types, namely, similarity range, similarity nearest-neighbor and similarity join.

Streaming Time Series: It is composed of a sequence of values, where each value corresponds to a time instance. The length changes, since new values are appended.

Time Series: It is composed of a sequence of values, where each value corresponds to a time instance. The length remains constant.

Chapter XXXIII
Internet Map Services and Weather Data

Maurie Caitlin Kelly
Pennsylvania State University, USA

Bernd J. Haupt
Pennsylvania State University, USA

Ryan E. Baxter
Pennsylvania State University, USA

INTRODUCTION

Internet map services (IMSs) are redefining the ways in which people interact with geospatial information system (GIS) data. The driving forces behind this trend are the pervasiveness of GIS software and the emerging popularity of mobile devices and navigation systems utilizing GPS (Global Positioning System), as well as the ever-increasing availability of geospatial data on the Internet. These forces are also influencing the increasing need for temporal or real-time data. One trend that has become particularly promising in addressing this need is the development of IMS. IMS is changing the face of data access and creating an environment in which users can view, download, and query geospatial and real-time data into their own desktop software programs via the Internet. In this section, the authors will provide a brief description of the evolution and system architecture of an IMS, identify some common challenges related to implementing an IMS, and provide an example of how IMSs have been developed using real-time weather data from the National Digital Forecast Database (NDFD). Finally, the authors will briefly touch on some emerging trends in IMS, as well as discuss the future direction of IMS and their role in providing access to real-time data.

BACKGROUND

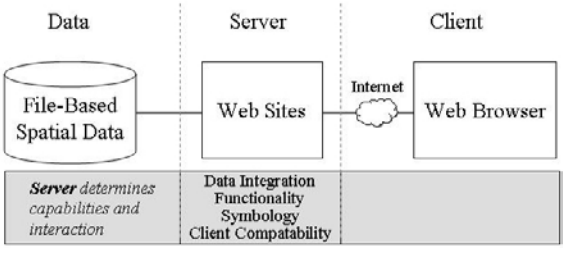
The origins of IMS can be traced to the geospatial data sharing initiatives of the 1990s when spatial data sharing began in earnest. In the early 1990s, the World Wide Web (WWW) and browsers altered users' perception of the Internet. Suddenly users were able to see images and interact with the Internet through graphics and scripts and even

access digital data. In the United States the creation of the National Spatial Data Infrastructure (NSDI) initiated an effort to develop standards for sharing data. By 1995 the vision for the NSDI to promote the sharing of data within the federal government and enhance sharing with state and local governments was solidified (Federal Geographic Data Committee, 1995).

Spatial data, which is defined as any data containing a locational component, have become widely available to the public through efforts of spatial data clearinghouses, governmental Web sites, and even nonprofit organizations, many of which were early NSDI initiatives. Through these spatial data clearinghouses, static spatial data were accessed traditionally via file transfer protocol (FTP). In cases such as these, the onus was on the user or client to identify, download, and manipulate data to comply with the desktop GIS environment. Temporal and real-time data have a more complex history. This type of information, which includes such diverse data types as traffic, hydrologic, and weather data, is difficult to maintain and provide access to due to its dynamic nature. For example, weather data, which encompasses everything from radar to precipitation to wind speed, are changing continuously and present the user with numerous data formats and types with which to contend (Van der Wel, Peridigao, Pawel, Barszczynska, & Kubacka, 2004). The past few years have seen advances on this front. For example, the National Weather Service (NWS) of the U.S. National Oceanic and Atmospheric Administration (NOAA) has developed several online applications that allow viewing of real-time weather information in a Web-based GIS environment. In addition, some data sets are being provided in a format that can be integrated into desktop GIS programs and tools have been created to convert the format and automate update processes within these programs (Liknes, Hugg, Sun, Cullen, & Reese, 2000). However, these tools do not allow for seamless integration of remote real-time databases into the desktop environment, an issue that the emergence of IMS has addressed.

IMS initiatives began in the mid 1990s. As the amount and availability of spatial and temporal data increased, so did the desire for increased interactive capabilities. Initially faced with the same historical constraints as in the past—large file size, long download times, and outdated data—data sites, programmers, and software companies began working toward providing users with the capability to bring data to their desktops without downloading the data: GIS over the Web became the new trend (Gold, 2006). The early steps toward visualization and subsequent desktop utilization of data via the Internet were basic. These applications were comprised of a set of data, predefined by the developer, which users could view via an interactive map interface that was built within a standard HTML (hypertext markup language) page. Analysis and navigation functionality, symbology, and compatibility with client browser software were controlled by the developer and server architecture (Figure 1). In many cases, the initial applications were environmental in nature, focusing on watershed health or water quality in a specific geographic area (Abel, Taylor, Ackland, & Hungerford, 1998). However, there were few if any customization capabilities and limited download capabilities in the first few years of the IMS movement. The user was simply viewing the data and turning data layers on and off. Another component of early IMS development was the use of a map as part of a search engine to access data (Kraak, 2004). As groundbreaking as these Web GIS applications were, they still had limited utility for those who wanted to acquire data and utilize them on their own desktops or who desired more than predefined static data sets.

Figure 1. Diagram of early IMS architecture

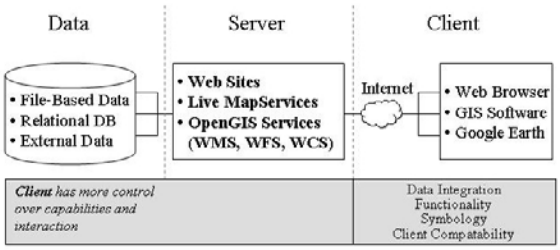


The next step in the evolution of this field would be the development of the capability for users to access exactly the data they needed without ever having to download a single file.

EVOLUTION OF IMS FROM WEB GIS APPLICATION TO MAP SERVICE

There are several different architecture types and performance enhancing capabilities that have emerged to support spatial data distribution and Web-based access over the past few years. The specific development and deployment architecture—from the type of software, database, and hardware used—are essentially a reflection of the internal capabilities of the developers and their organization. However, several major developments have impacted the performance of IMS and have enabled increased functionality and integration of data. From the basic interface of the early Web GIS applications have grown a host of new tools and capabilities that have made data not only viewable but also intrinsically more usable. The evolution of these basic interfaces and capabilities has occurred for a number of reasons. As stated earlier in this article, the Internet itself is an ever-changing landscape in which developers are consistently pushing the technology envelope and users are expecting bigger, better, and faster services and resources. Second, existing Web-based GIS services such as those provided by spatial data clearinghouses or services through sites such as Google Earth have created an environment in which people are more spatially aware. With this awareness comes the expectation of greater access to more relevant and timely data. Third is the extension of the traditional database model to include spatial data. Databases such as Oracle were most commonly utilized to store and access functional or business-related data. However, in the past decade, these databases have come to incorporate spatial and dynamic data and serve as the backbone for the development of access tools and IMS. Finally, there was the evolution of the IMS from a simple Web map that provided only

Figure 2. Diagram of the architecture for a map service



a static image of a data set to services that allow interaction and querying data.

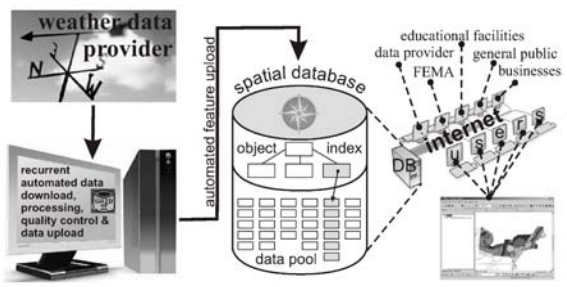
In the earliest incarnations of IMS development, data were housed within the application environment stored predominantly in a flat file structure. Commonly, data were stored on a server and accessed by the Internet mapping server directly. This architecture was adequate for data of relatively small sizes (kilobytes and megabytes). However, file-based data retrieval imposed constraints on performance once individual files and collections grew to sizes in the gigabytes and terabytes. Today, with the exponential growth in availability of detailed vector data (parcels, transportation networks, etc.) and especially high-resolution aerial photography, file sizes have rendered file-based storage impractical and prohibitively slow. Advancements in database technology, for example, the Environmental Systems Research Institute (ESRI) Spatial Database Engine (SDE), which manages data within the data tables in a relational database management system (RDBMS), have enabled traditional databases, such as Oracle and DB2, to house large quantities of both raster and vector spatial data. Also, database vendors themselves have developed spatial data models to extend their products, such as Oracle Spatial and DB2 Spatial Extender. Techniques such as the use of pyramids and new indexing procedures to manage and retrieve data within an RDBMS or SDE environment have also enabled substantially increased performance and speed of IMS (Chaowei, Wong, Ruixin, Menas, & Qi, 2005). In addition, the use of object-oriented

databases (OODBMS), which combine both data and function, are on the rise (Alam & Wasan, 2006). The potential impact of OODBMS is still unknown since it is found most commonly within the commercial or business environment, but as IMS becomes more reliant on database technologies, it is possible that OODBMS will increase the functionality of IMS.

By taking advantage of the data retrieval performance and efficiency of databases, Internet mapping servers are able to deploy applications and services containing terabytes of spatial data within acceptable time frames. In addition, as database management has become more mainstreamed, smaller organizations have developed the capability to build higher functioning IMSs. An example of this type of database is MySQL, which is a more user-friendly and affordable open-source database that utilizes the structured query language (SQL).

Building on the advances in databases and software, the next challenge was to bring the data from the browser environment onto the desktop—without requiring that users download the source data files to their own computers. There are several types of IMSs that have emerged to fill this need. The Open Geospatial Consortium, which is a nonprofit, international organization, has created several types of standards for IMS development including Web mapping services (WMSs), Web feature services (WFSs), and Web coverage services (WCSs). In addition, the two most common types of services are the feature service and image service. The image service allows the user to view a snapshot or picture of the data via a standard image format, such as JPEG or PNG. This type of service is particularly meaningful for those utilizing raster data, aerial photography, or other static data sets. The feature service is the more intelligent of the two as it streams raw spatial geometries (points, lines, polygons) to the client and allows the user to determine the functionality and symbology of the data. IMS is a particularly effective way of serving temporal real-time data since it allows for real-time updates of source data on the server side and customization on the client side. Users can access services through their desktop GIS software with no need to utilize a browser or use a search engine.

Figure 3. Example of the integration of remote temporal data, in this instance NDFD weather data for the continental United States (CONUS), from the NWS/NOAA



IMS and the Challenge of Weather Data

Real-time or temporal data by nature present numerous challenges to the IMS developer and database administrator as they bring with them the aspect of integrating the dimension of time. Weather or climate data are one of the most complex data types in the temporal data environment. In considering the development and deployment of a weather-based IMS, the unique data format, frequency of updates, and numerous data types, that is, those taken as surface points as well as those taken by satellite or radar must be considered (Liknes et al., 2000).

However, the significance of this data in areas such as emergency management cannot be underestimated. Just as important is the timeliness and speed with which the information is updated. There are numerous examples of the use of IMS to facilitate the sharing and display of time-sensitive data. These range from traffic management to earthquake early-warning systems and damage distribution maps (Erdik, Fahjan, Ozel, Alcik, Mert, & Gul, 2003). Climate and weather data pose their own unique problems and often present a heavy burden for those involved in developing

and serving temporal data (Shipley, Graffman, 7 Ingram, 2000).

Working with weather data proves to be difficult for several reasons. First there is the vast amount of output produced by weather stations and apparatus and high-resolution large-scale and regional climate forecast models that produce frequent output in different data formats (such as NetCDF); furthermore, there is the issue of timely data availability and at times unknown data quality. In addition, short-lived climate events like hurricanes and tornados are particularly challenging. The reasons for this are not related to the amount of data, data type, or the frequency of recurrent data updates. It is the temporary nature of the data availability and the immediacy of the event that impact the IMS developer when an extreme climate event occurs. Finally, the types of weather data vary. There are point data that include surface observations like lightning strikes and tornados, wind speed and gusts, and radar such as NEXRAD (Next-Generation Radar). Each type of data can require its own specific manipulation to prepare it for integration in a spatial data environment.

In the example illustrated by Figure 3, regularly updated weather data are automatically downloaded from given Internet FTP locations at predetermined times. The downloaded data are checked for completeness. Once the files are stored on a local server, they are "degribbed" (unpacked). The GRIB2 data files (e.g., temperature, apparent temperature, minimum and maximum temperature, probability of precipitation, precipitation, snow amount, dew point temperature, relative humidity, weather features, sky cover, wave height, wind speed and direction, wind gusts, etc.) contain the actual weather data in shapefile format. Each data set includes metadata and information about each available data layer for each forecasted time (up to 40 possible layers for up to 7 days). The data are then automatically uploaded into a relational spatial database, where they reside until new data are downloaded from the NWS/NOAA. In addition, map services are defined from updated map configuration files (AXL files) in order to provide the user with a map service that shows date and time.

The level of effort in accessing and incorporating temporal data into an IMS environment is difficult to determine by the simplified explanation provided here. Suffice it to say that the actual process of integrating this data requires approximately 5,000 lines of code to ensure that the data downloaded and errors resulting from interrupted dataflow during the data download are reported, and to ensure data availability and completeness, data preparation, automated database upload, that AXL file modifications and updated IMS services work flawlessly, and that possible foreseeable errors are covered by backup error procedures.

FUTURE TRENDS

There are numerous potential influences on spatial and temporal data and IMS development. Object-oriented database technologies are poised to challenge the relational database approach and influence the spatial database realm. As suggested previously, OODBMS has been utilized in business environments but have applicability in the spatial environment as well. In addition, significant work has been done with XML and the geography markup language (GML) that could influence the development of IMS. The creation of the NetCDF markup language or NcML could have a significant impact on weather and geoscience-related data exchange. Societal changes in the way people perceive spatial information and utilize spatially based services to access real-time data will increase the need for faster and more effective IMSs. Most recently, Google developed Google Earth that allows users to find imagery—satellite or aerial photography—of almost any area on earth via the Internet. As the technology that supports IMS improves, the integration of temporal data will increase. Database management software will provide the framework for managing large data collections and supporting the processing of temporal data. Performance factors have improved dramatically in a short period and will continue to do so until issues of loading time become obsolete. Emerging trends such as

image and feature services and the Open GIS efforts to develop advanced open-source WMS and WFS will allow the users greater flexibility and customization options and eventually the line between remotely stored and served data and the desktop will disappear. These improvements and the increased focus on emergency management and climate and weather information will place IMS in the forefront of providing support and information to stakeholders in need of data.

CONCLUSION

In conclusion, advances in serving data via the Internet, specifically Internet map services and Web-based GIS, are meeting the challenge of temporal data dissemination. In a period of only a decade, spatial and temporal data have moved from the realm of FTP to becoming fully integrated into the desktops of users in every walk of life. Spatial and temporal data that had heretofore required download time, manipulation, and consistent monitoring can now be viewed via a browser or imported into your desktop software with a click of a button.

REFERENCES

Abel, D. J., Taylor, K., Ackland, R., & Hungerford, S. (1998). An exploration of GIS architectures for Internet environments. *Computers, Environment, and Urban Systems, 22*(1), 7-23.

Alam, M., & Wasan, S. K. (2006). Migration from relational database to object oriented database. *Journal of Computer Science, 2*(10), 781-784.

Chaowei, P. Y., Wong, D., Ruixin, Y., Menas, K., & Qi, L. (2005). Performance-improving techniques in Web-based GIS. *International Journal of Geographical Information Science, 19*(3), 319-342.

Erdik, M., Fahjan, Y., Ozel, O., Alcik, H., Mert, A., & Gul, M. (2003). Istanbul earthquake rapid response and early warning system. *Bulletin of Earthquake Engineering, 1*(1), 157-163.

Federal Geographic Data Committee. (1995). *Development of a national digital geospatial data framework.* Washington, DC: Author.

Gold, C. M. (2006). What is GIS and what is not? *Transactions in GIS, 10*(4), 505-519.

Kraak, M. J. (2004). The role of the map in a Web-GIS environment. *Journal of Geographical Systems, 6*(2), 83-93.

Liknes, G., Hugg, R., Sun, J., Cullen, W., & Reese, C. (2000, June). *Techniques for conversion of weather data into ESRI formats.* Proceedings of the ESRI International Users Conference, San Diego, CA.

Shipley, S. T., Graffman, I. A., & Ingram, J. K. (2000, June). GIS applications in climate and meteorology. *Proceedings of the ESRI User Conference*, San Diego, CA.

Van der Wel, F., Peridigao, A., Pawel, M., Barszczynska, M., & Kubacka, D. (2004, June). COST 719: Interoperability and integration issues of GIS data in climatology and meteorology. *Proceedings of the 10th EC GI & GIS Workshop: ESDI State of the Art*, Warsaw, Poland.

KEY TERMS

Degrib: NDFD GRIB2 Decoder is the starting point for decoding the NDFD and various World Meteorological Organization (WMO) GRIB2 files. A "degribbed" file is a packed binary file or archive (GRIB2) like any other compressed archive; for more information see http://www.weather.gov/ndfd.

Feature Service: A feature service is a type of map service that delivers raw geometry features via the Internet to clients through desktop software or applications.

Image Service: An image service is a type of map service that generates a snapshot view of the spatial data requested by a client. The snapshot is stored in a standard image file, such as a JPEG, which is subsequently delivered to the client.

Map Service: A map service is a special type of Web-based service that allows spatial information, that is, maps, to be accessed remotely over the Internet and displayed and manipulated in a client software program.

NDFD: The National Digital Forecast Database is a database put together by the National Weather Service (NWS) to provide forecasts of sensible weather elements (e.g., cloud cover, maximum temperature) on a seamless grid. The NDFD data sets are currently given out to the public as GRIB2 files.

NetCDF: Network Common Data Form is an interface for array-oriented data access and a library that provides an implementation of the interface. The NetCDF library also defines a machine-independent format for representing scientific data. Together, the interface, library, and format support the creation, access, and sharing of scientific data that include metadata.

NEXRAD: Next-Generation Radar is an NWS network of about 160 Doppler weather surveillance radars (WSRs) operating nationwide.

Open GIS: Open GIS is the full integration of geospatial data into mainstream information technology by defining a standard syntax for requesting IMS. Particular formats include Web map services (WMSs), which are types of image services, and Web feature services (WFSs), which are types of feature services.

SDE: Spatial Database Engine is a software product from Environmental Systems Research Institute (ESRI) that acts as a middleware between the database and applications allowing GIS data to be stored in and retrieved from a relational database, such as Oracle or DB2.

Chapter XXXIV
Spatial Network Databases[1]

Michael Vassilakopoulos
University of Central Greece, Greece

INTRODUCTION

A Spatial Database is a database that offers spatial data types, a query language with spatial predicates, spatial indexing techniques, and efficient processing of spatial queries. All these fields have attracted the focus of researchers over the past 25 years. The main reason for studying spatial databases has been applications that emerged during this period, such as Geographical Information Systems, Computer-Aided Design, Very Large Scale Integration design, Multimedia Information Systems, and so forth.

In parallel, the field of temporal databases, databases that deal with the management of time-varying data, attracted the research community since numerous database applications (i.e., Banking, Personnel Management, Transportation Scheduling) involve the notion of time.

Since time and space are ubiquitous and in many cases related aspects of reality, the area of spatio-temporal databases, databases that simultaneously deal with the time and space aspects of data, also emerged. Technological advances that became accessible to the average user, such as accurate positioning systems (e.g., GPS), mobile computing, personal digital assistants with communication facilities, mobile and smart phones, and ubiquitous Internet terminals, made the development of applications about moving objects (an important kind of spatio-temporal data) possible. Such applications include vehicle navigation; tracking; and monitoring the positions of air, sea, or land-based equipment such as airplanes, fishing boats, and vehicles. In several of these applications, the objects move on predefined routes such as a transportation system, called a Spatial Network.

A Spatial Network is a collection of interconnected elements that have the shape of curves, or polylines, appearing in geographic applications (e.g., a railway network, an electricity network, a road network). A Spatial Network Database is a Spatial Database where the data modeled, stored, and queried are Spatial Network data. Queries mainly appearing in navigation

applications (e.g., find the route to the gas station that is closest to a traveling car) are the key motivation for the increased attention of the research community to Spatial Networks and Spatial Network Databases and Applications during last years. This chapter reviews the key aspects of Spatial Network Databases.

BACKGROUND

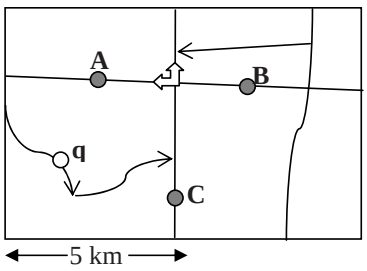
A Spatial Network can be considered a special kind of graph. A graph is a set of nodes and a set of edges connecting such nodes. Within this context, a Spatial Network is a graph in which the edges between nodes are constrained by the location of the nodes in space. Depending on the Spatial Network application, the graph can be directed (e.g., edges modeling one-way roads or gas distribution pipes) or not (e.g., edges modeling two-way roads or pedestrian streets), weighted (e.g., edges having a weight, such as speed limit or the number of lanes of a road), planar (e.g., a graph that can be drawn so edges do not intersect in the plane, such as a river network where river intersections correspond to graph nodes), or nonplanar (e.g., an electricity network where cables may intersect without an electrical junction between them). For example, in Figure 1, a part of a road network is depicted. The arrows indicate one-way roads. The filled circles indicate points of interest (e.g., gas stations A, B, and C), while the nonfilled circle represents the position of a motorist identified by q. The double-arrow symbol represents the permitted changes of direction at the related crossroad.

This graph-theoretic approach leads to a conceptual modeling of Spatial Networks that is independent of specific implementations and data management technologies (e.g., relational or object-oriented database management systems). By considering the representation of a Spatial Network graph through a specific data management technique, such as a set of interrelated tables or a collection of interreferenced objects, we reach a logical model, while the physical model of a Spatial Network describes the storage and indexing techniques used for the representation of the network.

The representation of a node of a Spatial Network may be enhanced by extra information (e.g., constraints or characteristics), such as permitted turns and changes of direction (e.g., the double-arrow symbol in Figure 1 that denotes the permitted changes of direction at the related crossroads); for example, in a navigation application. Similarly, an edge may be enhanced by extra information, such as (within a similar context) length, direction of movement (e.g., the roads with arrows in Figure 1), maximum allowed speed, and flow (traveling vehicle) capacity. Apart from representing the network itself, extra entities of several types may be modeled and represented in relation to the network (to its edges, in most cases). For example, static entities (points of interest) such as gas stations (e.g., the filled circles in Figure 1), stops of a train route, or tourist attractions; or moving/changing entities such as moving cars (represented by their current position) (e.g., the nonfilled circle in Figure 1) or their trajectory.

A Spatial Network Database, like any database, is useful when it enables answering of queries. The most interesting queries are those that refer not only to spatial characteristics of the network or the extra entities but also to the connectivity expressed

Figure 1. An example of a road network



by the network. For example, the motorist q is likely to be interested in reachable gas stations within traveling distance of 10km from his or her position and not about (possibly unreachable) gas stations within 10km of Euclidean distance (in Figure 1, station B is unreachable, although its Euclidean distance from q is less than 10km). Such queries include:

- Queries about the network infrastructure (e.g., list the towns that can be reached from Paris without changing highway).
- Route evaluation (e.g., calculate the volume gas that can flow though a route of pipes).
- Path computation (e.g., find the shortest path between two cities that visits certain tourist spots).
- Nearest neighbor queries (e.g., find the k nearest, in the network, gas stations to a car, in the example depicted in Figure 1, the 1 nearest station to q is C, considering traveling distance, although the Euclidean distance between q and A is smaller).
- Range queries (e.g., find the gas stations that are reachable by a car within traveling distance of 10km, in the example depicted in Figure 1, the answer is stations C and A, since station B is unreachable).
- Skyline queries with spatial predicates (e.g., find all hotels that are cheap and close (in the network) to a specific town, such that there is no other hotel that is at the same time cheaper and closer to this town.
- Closest pair queries (e.g., find the k hotel-restaurant pairs that are closest in the network).
- E-distance joins (e.g., find the hotel-restaurant pairs that are within distance e in the network).
- Continuous queries (e.g., continuously report the k nearest in the network gas stations of a car as it moves).

Several of these queries have been studied in the Spatial Database literature for static objects or for objects freely moving or moving in specific trajectories. In this chapter, we focus on queries where the connectivity of the underlying network affects the answer to the query.

The first overall approach on spatial and spatio-temporal data management in Spatial Networks is the DOMINO framework (Wolfson et al., 1999). Shekkar and Chawla (2003) present the aspects of Spatial Networks in the sixth chapter of their informative book, *Spatial Databases: A Tour*. More specifically, they deal with examples of applications; conceptual, logical, and physical data models; query languages; algorithms for query processing; and access methods. Another overall approach to Spatial Networks is given by De Almeida (2006), who reviews his work during his doctoral studies under the supervision of R.H. Güting. More specifically, he presents a prototype of a complete database management system for storing and querying moving objects in networks. A data model that supports historic movement, query processing supported by indices, an extended model for handling uncertainty, and a complete integrated implementation as an algebra inside the Secondo extensible database system is included. Servicing a different target to the previously mentioned publications, within a limited size, this chapter aims to introduce the reader to the concepts related to Spatial Network Databases and to guide him or her to the most important research advances in this area. In the next section, we review representative material related to models and languages, query processing, and indices of Spatial Networks.

MODELS, QUERIES, INDEXING

Models and Languages

Vazirgiannis and Wolfson (2001) present a model for moving objects on road networks, represented

by electronic maps. Additionally, a small set of expressive predicates is introduced.

Shekkar and Chawla (2003) present conceptual and logical modeling of Spatial Networks based on graphs. Additionally, they present SQL92 and SQL3 features that are suitable for querying Spatial Network Databases.

Speičys, Jensen, and Kligys (2003) present a data model that can support query processing of location-based services for mobile users constrained to road networks. The data model includes a Euclidean as well as a graph representation of road networks, data objects, and query objects. Hage, Jensen, Pedersen, Speičys, and Timko (2003) present a data model of a Spatial Network infrastructure based on multiple interrelated representations. These representations capture either Euclidean or connectivity characteristics and complement each other, providing the foundation for processing location-based queries.

Güting, et al. (2006), apart from a comprehensive survey of related work, present an integrated approach to modeling and querying moving objects in networks that consists of a precise data model and a query language. The model represents a spatially embedded network in terms of routes and junctions. Abstract data types (integrated into a relational environment with interface functions) for a network and for static and moving network positions and regions, along with a rich algebra to work with these types, are provided.

Query Processing and Indexing

Several access methods have been developed for storage and indexing of the Spatial Network infrastructure (the nodes and edges of the network) and the spatial entities related to this infrastructure (e.g., positions of points of interest, positions of moving objects, or trajectories of moving objects). Shekhar and Liu (1997) present an access method called CCAM (connectivityclustered access method) that groups nodes to common disk pages based on connectivity clustering (via graph partitioning, according to their weighted connectivity residue ratio). The resulting structure facilitates aggregate computations on Spatial Network infrastructures. Papadias, Zhang, Mamoulis, and Tao (2003) present a composite structure for the Spatial Network infrastructure that combines connectivity and Euclidean information. Moreover, they develop Euclidean restriction and network expansion approaches that take advantage of location and connectivity and apply these approaches for processing nearest neighbors, range search, closest pairs, and e-distance queries on entity sets (points of interest) indexed by R-trees. Chang, Kim, Kim, and Kim (2006) present a disk-based indexing method for both the Spatial Network infrastructure and the related spatial entities, and propose new query processing algorithms for range search and k nearest neighbors search, depending on the density of points of interest in the Spatial Network. These algorithms take advantage of stored shortest network distances between all nodes in the Spatial Network. Hu, Lee, and Xu (2006) propose indexing a Spatial Network infrastructure by replacing the graph topology with a set of interconnected tree-based structures called SPIEs. A special kind of index for related spatial entities (points of interest), called *nd* index, is constructed on each SPIE to efficient processing of nearest neighbor queries

Several researchers have focused on indexing objects moving in a Spatial Network (network-constrained trajectories). Frentzos (2003) develops an access method for objects moving on fixed networks, called Fixed Network R-Tree (FNR-Tree). This is a forest of one-dimensional R-Trees on top of a two-dimensional R-Tree. The 2D R-Tree indexes the Spatial Network infrastructure (e.g., roads consisting of line segments), while the 1D R-Trees index the time interval of each object's movement on a network edge. Pfoser and Jensen (2003), taking advantage of the fact that a movement on a specific network edge

is one-dimensional, develop a technique that reduces the dimensionality of the data (trajectory data in a Spatial Network). The advantage is the reduced size of the data and the lower-dimensional indexing method of trajectories of objects moving on a Spatial Network that can be supported by a typical database management system. De Almeida and Güting (2005) propose an index structure for moving objects on networks (trajectory data) called Moving Objects in Networks Tree (MON-tree). This tree is based on R-trees and indexes both an edge-oriented and route-oriented network model. Li and Lin. (2006) develop an access method called Topology-based Mixed Index Structure (TMIS) to index trajectories of objects moving on a Spatial Network. TMIS consists of a combination of simple and classical index structures linked by the topology of a network and can support several queries.

The most studied query in Spatial Networks is the nearest neighbor query, which has several variations. First, Jensen, Kolárvr, Pedersen, and Timko (2003) present a data model, implementable using the relational model, and a set of algorithms for processing nearest neighbor queries in Spatial Networks. The next work that deals with nearest neighbor queries (among other types) in Spatial Networks is the previously mentioned work by Papadias, et al. (2003) that is based on the Euclidean restriction and network expansion approaches. Kolahdouzan and Shahabi (2004) propose an approach for the evaluation of k nearest neighbor queries in Spatial Network databases using first-order Voronoi diagrams. A large network is partitioned to small Voronoi regions, and the distances both within and across the regions are precomputed. Huang, Jensen, and Saltenis (2005) present a method for processing k nearest neighbor queries in Spatial Networks, termed the Islands approach, that is based on a flexible and simple technique of balancing recomputation and precomputation, thus managing the trade-off between query and update performance. Deng, Zhou, Shen, Xu, and Lin (2006) investigate the problem of surface k nearest neighbor query processing, where the distance is calculated from the shortest path along a terrain surface, a challenging problem since terrain data can be very large and the cost of finding shortest paths is very high. An efficient solution based on multiresolution terrain models is proposed in this chapter. Huang, Jensen, and Saltenis (2006) present techniques for processing of multiple k nearest neighbor queries in a road-network setting. Progressive techniques that selectively cache query results in main memory and subsequently reuse these for query processing are presented. Cases with a given upper bound on k and no bound on k are examined. Ku, Zimmerman, Wang, and Wan (2005) develop a local-based greedy nearest neighbor algorithm and a global-based adaptive nearest neighbor algorithm that both utilize real-time traffic information of the Spatial Network to provide adaptive nearest neighbor search results, according to minimum travel time. Yoo and Shekhar (2005) address the problem of finding the in-route nearest neighbor for a given route with a destination and a given location. The in-route nearest neighbor is a neighbor that is reachable with the smallest detour distance from the original route on the way to the destination. Yiu, Mamoulis, and Papadias (2005) study the processing of aggregate nearest neighbor queries in road networks; that is, queries that return the object that minimizes an aggregate distance function with respect to a set of query points (e.g., users at specific locations—query points—seek the restaurant that leads to the minimum sum of distances that they have to travel in order to meet). Yiu, Papadias, Mamoulis, and Tao (2006) study reverse *k* nearest neighbor queries in the context of large graphs. A reverse nearest neighbor query returns the data objects that have a query point as their nearest neighbor. In thier paper, eager and lazy techniques are presented. De Almeida and Güting (2006) propose an indexing technique to support an efficient execution of a modified

version of Dijkstra's algorithm for answering k nearest neighbor queries.

Sankaranarayanan, Alborzi, and Samet (2005) propose the use of a framework, called Spatially Induced Linkage Cognizance (SILC), that precomputes the shortest path between all pairs of nodes in a Spatial Network. The encoding used is compact and fast in path and distance retrievals. Using this framework, a variety of spatial queries such as incremental nearest neighbor searches and spatial distance joins can be processed.

Suppose you are looking for a hotel that is cheap and close to the beach. The skyline query returns all hotels that are not worse than any other hotel in both dimensions (note that one of the predicates, the distance from the beach, is a spatial one). A multisource skyline query considers several query points at the same time (e.g., to find hotels that are cheap and close to the university, the beach, and the botanic garden). Deng, Zhou, and Shen (2007) present three algorithms for processing multisource skyline queries in road networks.

Path computation queries, having roots in graph theory, appear also in the context of Spatial Networks. Li, Cheng, Hadjieleftheriou, Kollios, and Teng (2005) present fast approximation algorithms for processing the Trip Planning Query (TPQ) on metric graphs. Given a set of points of interest in space where each point belongs to a specific category, a starting point, and a destination, TPQ retrieves the best trip that starts at the starting point, passes through at least one point from each category, and ends at the destination.

Continuous queries, where the answer is altered as the position of the moving object changes, have also attracted researchers of Spatial Networks. Cho and Chung (2005) and Kolahdouzan and Shahabi (2005) present algorithms for continuous nearest neighbor queries; that is, for finding the nearest (static) points of interest along an entire network path. Mouratidis, Bakiras, and Papadias (2006) refer to continuous monitoring of nearest neighbors on road networks, where the query points and the data objects move frequently and arbitrarily. Two methods are proposed, one that maintains the query results by processing only updates that might affect its result and ignoring the rest, and another that groups together the queries falling in the path between two consecutive intersections in the network and produces their results by monitoring the nearest neighbor sets of these intersections. Liu, Do, and Hua (2006) present distributed processing of dynamic range queries in Spatial Networks. Such a query retrieves the moving objects within a specified network distance of the moving query point. The term *distributed* refers to utilizing computing power of the moving objects to help reduce server load and save wireless bandwidth.

FUTURE TRENDS

Existing models are expected to be enriched by taking into account additional aspects of Spatial Networks infrastructure (even time-varying properties such as traffic data) as well as aspects of the objects moving in the infrastructure (not currently supported by all the models) in order to express real-life situations in more detail. The ability of each model to support answering of more kinds of queries (e.g., traffic jam discovery and prediction) will also be a driving force of enrichment. Model enhancements are expected to be followed by extensions of the query language features for expressing these enhancements. Integration of models and access methods is another issue that is expected to be addressed in the future.

The consideration of 3-D Spatial Networks to express situations where the fluctuation of the terrain elevation is important (such as in Deng et al., 2006) will affect all the components of the Spatial Network framework: models, languages, access methods, queries, and algorithms.

Time-varying (continuous) versions of several queries could be also developed; for example,

Trip Planning queries in a Spatial Network embedding traffic information. Moreover, new query types (variations and extensions of queries appearing in the spatial database or the general database literature) could appear. For example, Tiakas, Papadopoulos, Nanopoulos, Manolopoulos, Stojanovic, and Djordjevic-Kajan (2006) extending similarity queries in the context of Spatial Networks, present techniques for similarity search of trajectories of objects moving in Spatial Networks by using metric-based access methods. The utilization of the computing power of the moving objects (such as in Liu et al., 2006) could lead to distributed processing of queries.

Qualitative (functionality) and efficiency comparison of access methods for Spatial Networks, especially of newly proposed to existing ones; comparison of alternative algorithmic approaches for the same or similar queries and development of generic (and as simple as possible) indexing techniques exhibiting adequate performance and diverse functionality are also possible trends of research.

CONCLUSION

Spatial Networks Databases are emerging from Spatial and Spatio-temporal Databases as a distinct data management technology. In this chapter, we have reviewed the motivation for developing techniques for the management of Spatial Networks, their fundamental concepts; we have reported representative and recent research efforts; and we have discussed possible future research directions. Although numerous research efforts have recently been devoted to Spatial Networks, significant research challenges still remain.

REFERENCES

Chang, J.-W., Kim, Y.-K., Kim, S.-M., & Kim, Y.-C. (2006). *New query processing algorithms for range and k-NN search in spatial network databases.* Proceedings of the ER Workshops, 130–139.

Cho, H.-J., & Chung, C.-W. (2005). *An efficient and scalable approach to CNN queries in a road network.* Proceedings of the VLDB 2005, 865–876.

De Almeida, V.T., & Güting, R.H. (2005). Indexing the trajectories of moving objects in networks. *GeoInformatica, 9*(1), 33–60.

De Almeida, V.T., & Güting, R.H. (2006). *Using Dijkstra's algorithm to incrementally find the k-nearest neighbors in spatial network databases.* Proceedings of the SAC 2006, 58–62.

De Almeida, V.T. (2006). *Moving objects in networks databases.* Proceedings of the EDBT Ph.D. WorkshopEDBT Workshops 2006, 75–85.

Deng, K., Zhou, X., & Shen, H.T. (2007). *Multi-source skyline query processing in road networks.* Proceedings of the ICDE 2007.

Deng, K., Zhou, X., Shen, H.T., Xu, K., & Lin, X. (2006). *Surface k-NN query processing.* Proceedings of the ICDE 2006, 78.

Frentzos, E. (2003). *Indexing objects moving on fixed networks.* Proceedings of the SSTD 2003, 289–305.

Güting, R.H., De Almeida, V.T., & Ding, Z. (2006). Modeling and querying moving objects in networks. *VLDB J., 15*(2), 165–190.

Hage, C., Jensen, C.S., Pedersen, T.B., Speičys, L., & Timko, I. (2003). *Integrated data management for mobile services in the real world.* Proceedings of the VLDB 2003, 1019–1030.

Hu, H., Lee, D.L., & Xu, J. (2006). *Fast nearest neighbor search on road networks.* Proceedings of the EDBT 2006, 186–203.

Huang, X., Jensen, C.S., & Saltenis, S. (2005). *The islands approach to nearest neighbor querying in spatial networks.* Proceedings of the SSTD 2005, 73–90.

Huang, X., Jensen, C.S., & Saltenis, S. (2006). *Multiple k nearest neighbor query processing in spatial network databases.* Proceedings of the ADBIS 2006, 266–281.

Jensen, C.S., Kolárvr, J., Pedersen, T.B., & Timko, I. (2003). Nearest neighbor queries in road networks. Proceedings of the ACM-GIS 2003, 1–8.

Kolahdouzan, M.R., & Shahabi, C. (2004). *Voronoi-based K nearest neighbor search for spatial network databases.* Proceedings of the VLDB 2004, 840–851.

Kolahdouzan, M.R., & Shahabi, C. (2005). Alternative solutions for continuous K nearest neighbor queries in spatial network databases. *GeoInformatica, 9*(4), 321–341.

Ku, W.-S., Zimmermann, R., Wang, H., & Wan, C.-N. (2005). *Adaptive nearest neighbor queries in travel time networks.* Proceedings of the ACM-GIS 2005, 210–219.

Li, F., Cheng, D., Hadjieleftheriou, M., Kollios, G., & Teng, S.-H. (2005). *On trip planning queries in spatial databases.* Proceedings of the SSTD 2005, 273–290.

Li, X., & Lin, H. (2006). Indexing network-constrained trajectories for connectivity-based queries. *International Journal of Geographical Information Science, 20*(3), 303–328.

Liu, F., Do, T.T., & Hua, K.A. (2006). *Dynamic range query in spatial network environments.* Proceedings of the DEXA 2006, 254–265.

Mouratidis, K., Bakiras, S., & Papadias, D. (2006). *Continuous monitoring of top-k queries over sliding windows.* Proceedings of the SIGMOD Conference 2006, 635–646.

Papadias, D., Zhang, J., Mamoulis, N., & Tao, Y. (2003). *Query processing in spatial network databases.* Proceedings of the VLDB 2003, 802–813.

Pfoser, D., & Jensen, C.S. (2003). *Indexing of network constrained moving objects.* Proceedings of the ACM-GIS 2003, 25–32.

Sankaranarayanan, J., Alborzi, H., & Samet, H. (2005). *Efficient query processing on spatial networks.* Proceedings of the ACM-GIS 2005, 200–209.

Shekhar, S., & Chawla, S. (2003). *Spatial databases: A tour.* Prentice Hall.

Shekhar, S., & Liu, D.-R. (1997). CCAM: A connectivity-clustered access method for networks and network computations. *IEEE Trans. on Knowledge & Data Eng., 9*(1), 102–119.

Speičys, L., Jensen, C.S., & Kligys, A. (2003). *Computational data modeling for network-constrained moving objects.* Proceedings of the ACM-GIS 2003, 118–125.

Tiakas, E., Papadopoulos, A.N., Nanopoulos, A., Manolopoulos, Y., Stojanovic, D., & Djordjevic-Kajan, S. (2006). *Trajectory similarity search in spatial networks.* Proceedings of the IDEAS 2006, 185–192.

Vazirgiannis, M., & Wolfson (2001). *A spatiotemporal model and language for moving objects on road networks.* Proceedings of the SSTD 2001, 20–35.

Wolfson, O., et al. (1999). *Tracking moving objects using database technology in DOMINO.* Proceedings of the NGITS 1999, 112–119.

Yiu, M.L., Mamoulis, N., & Papadias, D. (2005). Aggregate nearest neighbor queries in road networks. *IEEE Trans. Knowl. Data Eng., 17*(6), 820–833.

Yiu, M.L., Papadias, D., Mamoulis, N., & Tao, Y. (2006). Reverse nearest neighbors in large graphs. *IEEE Trans. Knowl. Data Eng., 18*(4), 540–553.

Yoo, J.S., & Shekhar, S. (2005). In-route nearest neighbor queries. *GeoInformatica, 9*(2), 117–137.

KEY TERMS

Graph: A set of nodes and a set of edges connecting such nodes.

Spatial Network: A collection of interconnected elements that have the shape of curves, or polylines, appearing in geographic applications (e.g., a railway network, an electricity network, a road network).

Spatial Access Method, or Spatial Index: A technique of organizing spatial data (multidimensional data such as points, line segments, regions, polygons, volumes, or other kinds of geometric entities) that allows the efficient retrieval of data according to a set of search criteria.

Spatial Database: A database that offers spatial data types, a query language with spatial predicates, spatial indexing techniques, and efficient processing of spatial queries.

Spatial Network Database: A spatial database where the data modeled, stored, and queried are Spatial Network data.

Spatial Query: A request expressed as a combination of spatial conditions (e.g., Euclidean distance from a query point) for extracting specific information from a large amount of spatial data without actually changing these data.

***k* Nearest Neighbor Query:** A query of finding *k* entities (data elements) among a group of entities that is nearest (according to a distance function) to a given (query) entity.

Range Query: A query of finding all entities (data elements) among a group of entities with distances (according to specific distance functions) from a given (query) entity that satisfy some given conditions (e.g., with Euclidean distance that is smaller than a given value or with horizontal and vertical distances smaller than two respective given values).

Moving Object or Moving Point: An entity (data element) in space with a varying position in the course of time. At any specific time instance, a moving object is characterized by its position, its direction of movement, and its speed of movement.

Trajectory: The track followed by a moving object in the course of time (due to the change of its position).

ENDNOTE

[1] Supported by the projects "Management of Moving Objects and the WWW" (ARCHIMEDES - EPEAEK II - cofunded by the European Social Fund and National Resources) and "Efficient Image Databases" (bilateral agreement between Bulgaria and Greece – cofunded by the General Secretariat of Research and Technology of Greece and the Bulgarian State).

Chapter XXXV
Supporting Location-Based Services in Spatial Network Databases

Xuegang Huang
Aalborg University, Denmark

INTRODUCTION

Location-based services (LBSs) utilize consumer electronics, mobile communications, positioning technology, and traditional map information to provide mobile users with new kinds of online services. Examples include location-sensitive information services that identify points of interest that are in some sense nearest and of interest to their users and that offer travel directions to their users. Data management is a core aspect of the provisioning of LBSs. The diversity and complexity of LBSs demand novel and advanced data management techniques.

The scenario of network-constrained movement is of particular interest to LBSs as a large amount of users of LBSs are car drivers and mobile users moving in vehicles. We term the transportation networks where LBS users are moving as spatial networks. The databases that manage the spatial network data as well as other relevant data are termed spatial network databases (SNDBs). Data management in SNDBs poses novel challenges to the database research community. Specifically, most existing discussions on spatial databases assume that objects can move arbitrarily in Euclidean space. The fundamental difference between network-constrained space and the Euclidean space is that the distance of two objects cannot be computed by their coordinates but by the shortest path on the network (Jensen, Kolář, Pedersen, & Timko, 2003). This makes it difficult to extend the existing data models, indexing structures, and query processing techniques from spatial databases to SNDBs.

This article summarizes existing efforts from the database community to support LBSs in spatial

networks. The focus of discussion is on the data models, data structures, and query processing techniques in SNDBs. An example application framework is presented to illustrate the relationship of these topics. Challenges and future trends are also addressed.

BACKGROUND

Due to the natural similarity between spatial networks and graphs, the study of SNDBs can be traced back to the study of graph models in databases (Gyssens, Paredaens, & Gucht, 1990). The graph models discussed in these works cannot be directly used for LBSs and many real-world aspects of networks, such as U-turns (U-turn means a complete reversal of travel direction) and one-way roads, are not considered. There are also past studies on shortest-path computation in large urban transportation networks (Shekhar, Kohli, & Coyle, 1993). Assuming that network data are stored in disk pages, these papers are focused on techniques that optimize the amount of disk accesses in the shortest-path search.

In the study of LBSs, spatial networks have several distinct differences from graphs. Specifically, the study of graph concentrates on the links between graph nodes, while in spatial networks, positions of nodes are of equal importance to links. Second, as an important element in LBSs, data on points of interest are represented in spatial networks using linear referencing. Next, data modeling of spatial networks often needs to consider the pragmatic constraints in real-world road networks, such as U-turns and one-way roads, which are not very common in the study of graphs. The study of moving-object databases is also relevant to our discussion as users of LBSs are often modeled as moving objects in networks.

We proceed to discuss relevant works in three categories, that is, data modeling of spatial networks and network-constrained moving objects, data structures of spatial networks, and spatial and spatiotemporal query processing in spatial networks.

Data Modeling of Spatial Networks and Network-Constrained Moving Objects

Vazirgiannis and Wolfson (2001) first considered modeling and querying for moving objects in road networks. In this paper, the network model is basically an undirected graph, with nodes representing road junctions and edges representing roads between these junctions. The paper also considers trajectories as basic elements to represent moving objects in databases.

The paper by Hage, Jensen, Pedersen, Speičys, and Timko (2003) presents an industrial study on the data modeling of spatial networks for supporting LBSs. The paper presents four interrelated representations of the road networks. The kilometer-post representation provides a convenient and precise way of relating a location in the physical world with a location stored in a database. The link-node representation is based on the mathematical graph model. The geographical representation contains the geographical coordinates of the roads. The segment representation models the network as a collection of polyline segments that intersect at road intersections. Other representations of the network can be mapped to and from the segment representation, which ensures that all these representations are interrelated and locations in one representation can be transformed to another in very convenient ways.

Next, the paper by Speičys, Jensen, and Kligys (2003) describes a formalized model that incorporates the ideas described in Hage et al. (2003). This paper proposes the 2-D representation and the graph representation, both of which are similar to a graph while the 2-D representation is focused on capturing the detail of road network properties and the graph representation is used to support efficient computations. This paper also proposes the connection matrix (also the co-edge relationship in the graph representation) to reflect the real-world situations on road junctions where turning from one road to another is restricted. On top of the network, there are data points and query points, which serve as basic elements of LBSs.

Data points represent points of interest and query points represent mobile users who send query requests to LBS servers.

The paper by Jensen et al. (2003) extends the two network models in Speičys et al. (2003) to include moving objects. The paper explores how to represent the moving data point in both the 2-D and the graph representations and how to compute nearest-neighbor queries on the moving data points. To capture the dynamic characteristics of transportation networks, Ding & Güting (2004) developed a data model for dynamic networks. The data model presented in the paper is a graph that describes how the availability of nodes or edges can change over time. Güting, Almeida, and Ding (2006) propose a comprehensive framework including the data modeling of spatial networks and a query language. Compared to previous works, this paper defines the network as a set of routes and a set of junctions between these routes. This definition is similar to the segment representation in Hage et al., (2003) but with thorough extensions.

Data Structures of Spatial Networks and Network-Constrained Moving Objects

The growth of urban transportation networks in many cities worldwide makes the networks larger and denser. Thus, the storage size of spatial networks can be too large to fit in computer memory. It becomes necessary to consider disk-based data structures for saving spatial network data and relevant points of interest data in the network.

We start our discussion from the CCAM (Shekhar & Liu, 1997) method. This structure assumes a graph-oriented model of the network and considers extending the adjacency list structure for a graph to a disk-based structure. Specifically, the CCAM method adopts a heuristic graph-partitioning approach from the area of VLSI design. This approach clusters nodes of a network into pages based on the connectivity relationship so as to maximize the chances that a pair of connected nodes that are more likely to be accessed together are allocated to the same disk page. Then a B^+ tree is created to index the network nodes based on their *Z* orders. The CCAM method is applicable to shortest-path and many other graph traversal algorithms.

The CCAM structure is then extended for query processing in spatial networks. Papadias, Zhang, Mamoulis, and Tao (2003) propose a data structure that supports spatial queries on both networks and the 2-D space. This structure includes three parts: the adjacency component, which is adapted from CCAM, the polyline component that stores the detail polyline data of each network edge, and the network R tree, which is an R tree indexing on the polylines' minimum bounding rectangles (MBRs). Data about points of interest are stored in another R tree based on their location in space. This data structure enables the spatial search such as *k*-nearest neighbor (kNN) queries on both networks and the Euclidean space. The adjacency component has been adapted by quite a few papers studying query processing techniques in spatial networks, such as Kolahdouzan and Shahabi (2004) and Huang, Jensen, and Šaltenis (2005).

In applications such as fleet management and traffic monitoring, it is often necessary to manage the moving location data of mobile users. The research on moving objects databases has been focused on indexing structures for objects moving in Euclidean space and only a little attention has been paid on indexing structures for network-constrained moving objects. These Euclidean-space indexing structures, such as the TPR tree (Šaltenis & Jensen, 2002), are more applicable for LBS applications than an indexing structure for network-constrained moving objects. One reason is that it becomes unnecessary to consider indexing structures for the current and near future movement of objects in networks since the management of such movement becomes easy when only coordinates of moving objects are considered and map matching from an object's coordinate to a network location is very simple. We consider these Euclidean-space indexing structures orthogonal to our discussion as we are focused on research in SNDBs.

Figure 1. Example map

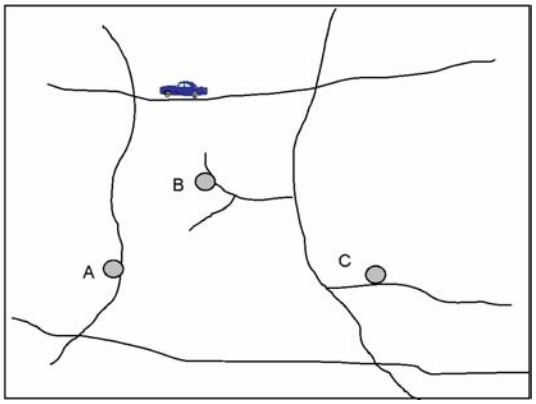
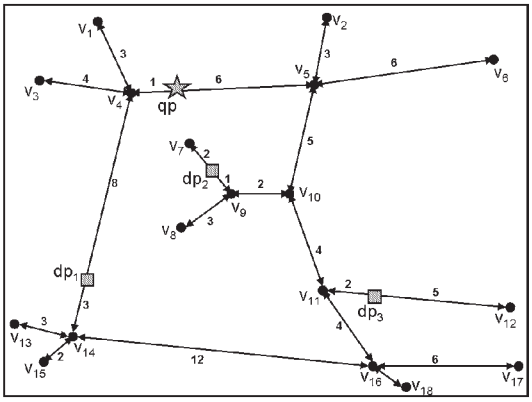


Figure 2. Network data model



Spatial and Spatiotemporal Query Processing in SNDBs

Many types of spatial queries, such as *k*-nearest neighbor query and range search, have been extensively studied in the spatial database literature when Euclidean distance and R-tree indexing of the data are considered. The kNN query returns *k* objects that are closest to a query object in space. The paper by Jensen et al. (2003) first explores *k*-nearest neighbor query in road networks. Then, Papadias et al. (2003) explored the query processing techniques for *k*-nearest neighbor and other query types in spatial networks. The dominant technique presented in that paper is termed incremental network expansion (INE). It makes a network expansion from the query point to incrementally scan the network and find *k*-nearest neighbors. The disadvantage of INE lies in the huge amount of disk accesses when the network is very dense and points of interest are sparsely distributed in the network. To provide optimal performance in terms of disk accesses to network data, several precomputation techniques (Cho & Chung, 2005; Huang et al., 2005; Kolahdouzan & Shahabi, 2004) have been developed on kNN query processing. All these techniques are focused on using precomputations of network distance to optimize the performance of kNN queries. In addition to these techniques, Huang, Jensen, & Šaltenis (2006) discuss main-memory caching strategies for improving the kNN query performance.

The discussion on continuous *k*-nearest neighbor (CkNN) queries in spatial networks is based on techniques for kNN query processing. Two existing papers (Cho & Chung, 2005; Kolahdouzan & Shahabi, 2004) define the CkNN query in networks as to find a set of intervals on the path and the corresponding network *k* nearest neighbors so that, for each interval, the *k* nearest neighbors to any point within that interval are the same. This definition is similar to the in-route nearest neighbor queries (Yoo & Shekhar, 2005) discussed in the literature.

The study of data models, data structures, and query processing techniques in SNDBs provides an architectural way of supporting LBSs in spatial networks. We proceed to demonstrate this architecture with an example LBS framework with data models, data structures, and a query processing algorithm selected from the aforementioned discussion. The next section also presents the definition of kNN query and descriptions of several other query types.

AN EXAMPLE LBS FRAMEWORK

We formulate the LBS as the following: We assume the spatial network is a road network and users of the LBS are road-network constrained; for example, users may drive by car or may be bus passengers. Next, a number of facilities or so-called points of interest, for example, gas sta-

Figure 3. Example data structure

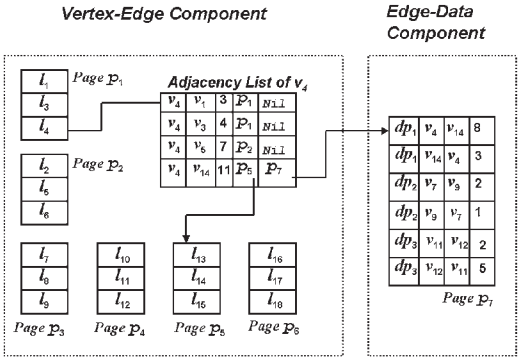


Figure 4. Running example of the INE algorithm

Step	Q_v	Q_{dp}	d_2	Buffer	Disk Access
1	$(v_4,1),(v_3,6)$	Φ	∞	Φ	0
2	$(v_1,4),(v_3,5)$, $(v_5,6),(v_{14},12)$	$(dp_1,9)$	∞	p_1, p_7	2
3	$(v_5,6),(v_{11},12)$	$(dp_1,9)$	∞	p_1, p_7	2
4	$(v_2,9),(v_{10},11)$, $(v_{14},12),(v_6,12)$	$(dp_1,9)$	∞	p_7, p_2	3
5	$(v_{10},11),(v_{14},12)$, $(v_6,12)$	$(dp_1,9)$	∞	p_7, p_2	3
6	$(v_{14},12),(v_6,12)$, $(v_9,13),(v_{11},15)$	$(dp_1,9)$	∞	p_2, p_1	4
7	$(v_6,12),(v_9,13)$, $(v_{15},14),(v_{11},15)$, $(v_{13},15)$	$(dp_1,9)$	∞	p_4, p_5	5
8	$(v_9,13)$, $(v_{15},14),(v_{11},15)$, $(v_{13},15)$	$(dp_1,9)$	∞	p_4, p_5	5
9	$(v_{15},14),(v_{11},15)$, $(v_{13},15),(v_7,16)$, $(v_8,16)$	$(dp_1,9)$ $(dp_2,14)$	14	p_3, p_7	7
10	$(v_{11},15)$, $(v_{13},15),(v_7,16)$, $(v_8,16)$	$(dp_1,9)$ $(dp_2,14)$	14	p_3, p_7	7

tions and supermarkets, are located within the road network. The users send queries such as kNN and CkNN to the LBS server to find points of interest.

We term users query points and the points of interest data points, and proceed to discuss the data model, data structure, and the kNN query processing algorithm.

The Road Network Model

A road network is defined as a two-tuple, $RN = (G, \Im)$, where G is a directed, labeled graph and $\Im$ is a binary, so-called co-edge, relationship on edges. The graph G is itself a two-tuple, (V, E), where V is a set of vertices and E is a set of edges. Vertices model intersections and starts and ends of roads. An edge e models the road in-between an ordered pair of vertices and is a three-tuple, $e = (v_s, v_e, l)$, where $v_s, v_e \in V$ are, respectively, the start and end vertex of the edge. The edge can be traversed only from v_s to v_e. The element l captures the travel length of the edge. A pair of edges e to e' belong to $\Im$, if and only if they represent the same bidirectional part of a road and a U-turn is allowed from e to e'.

Next, a location is a two-tuple (e, pos) where e is the edge where the location is located and pos represents the length from the start vertex of the edge to loc. Then, a data point is modeled as a nonempty set of locations, that is, $dp = \{loc_1, loc_2, ..., loc_k\}$.

A query point qp is modeled as a two-tuple (e, pos) where e is the edge on which the query point is located and pos represents the length from the start vertex of the edge to qp. Given a query point and a value k, the kNN query returns k data points for which no other data points are closer to the query point in terms of road-network distance.

An edge with start and end vertices v_i and v_j is denoted by $e_{i,j}$. Based on the map in Figure 1, Figure 2 illustrates the concepts defined above, for example, edge $e_{3,4} = (v_3, v_4, 4)$, data point $dp_2 = \{(e_{7,9}, 2)(e_{9,7}, 1)\}$, and query point $qp = (e_{5,4}, 6)$.

For the simplicity of our discussion, we assume that the road network of Figure 2 only contains bidirectional edges and there is no U-turn restriction. As pointed out by Huang et al. (2006), the INE algorithm described next still works with these restrictions. Thus, each edge has a corresponding co-edge connecting the two vertices in the opposite direction. Each data point then has two positions: one on each edge that models the road along which the data point is located. Note that in Figure 2 we draw the two co-edges as one edge with two arrows.

The Road-Network Data Structure

A road network is represented physically by two adjacency components on disk (Huang et al., 2005), that is, the vertex-edge component that

captures the network adjacency and the edge-data component that captures the connectivity between edges and data points. The vertex-edge component consists of adjacency lists of vertices. Vertices can be assigned into data pages by the modified graph partitioning algorithm in Shekhar and Liu (1997). As illustrated in Figure 3, the vertex-edge component contains disk pages p_1 to p_6. Page p_1 contains the adjacency lists l_1, l_3, l_4 of vertices v_1, v_3, v_4. The adjacency list l_4 for vertex v_4 is composed of entries standing for edges starting at v_4. The edge-data component stores the list of data points linked to each edge, that is, page p_7.

Each entry in the vertex-edge component has the form (*vsID, veID, L, vePage, dpPage*), where *vsID* and *veID* are the identity attribute value of the start and edge vertices, *L* is the travel distance of the edge, *vePage* is the pointer to the disk page containing the end vertex, and *dpPage* is the pointer to the disk page containing data points on this edge. The value of *dpPage* is set to *Nil* if there are no data points on this edge. In the edge-data component, each entry has the form (*dpID, vsID, veID, vsOffset*), where *dpID* is the identity attribute value of the data point, *vsID* and *veID* are the identity attribute values of the start and end vertices of the edge that contains the data point, and *vsOffset* is the travel distance from the start vertex to the data point along the edge.

The INE Algorithm for kNN Query Processing

The INE algorithm (Papadias et al., 2003) incrementally expands to search for data points through a network. Despite the aforementioned shortcomings, the INE approach is the simplest algorithm to implement and it has been extended by other approaches (such as Huang et al., 2005, 2006) to solve many other query types. Another feature of this algorithm is its robustness to updates; that is, updates to the road network and points of interest do not influence the performance of the query algorithm.

At each step, this algorithm reads the closest vertex *w* from a priority queue Q_v, which stores yet-to-be-visited vertices in the order of their network distance from the query point. Then it puts all nonvisited adjacent vertices of *w* into Q_v and inserts the data points found on the adjacent edges of *w* into a queue Q_{dp} that stores the data points found so far. Let d_k denote the network distance from the query point to the k^{th} nearest neighbor in Q_{dp}. The search terminates when *k* nearest neighbors are found and the distance from the query point to the next vertex to be explored is larger than d_k. Assuming that the memory buffer size is two disk pages, Figure 4 illustrates the steps of the INE algorithm for a 2NN query at $qp = (e_{5,4}, 6)$ in the network of Figure 2.

Other Queries in Spatial Networks

In addition to kNN queries, many other query types can be processed under the same work. The following lists three queries in spatial networks that have been discussed in the literature.

Range search: Given a source point *q*, a value *e*, and a spatial data set *S*, a range query retrieves all objects of *S* that are within network distance *e* from *q* (e.g., find hotels within 5km driving distance).

***k*-closest pair:** Given two data sets *S* and *T* and a value *k*, a closest-pairs query retrieves the *k* pairs (*s*,*t*), $s \in S$, $t \in T$, that are closest in the network (e.g., find the hotel-restaurant pair within the shortest driving distance).

e**-distance join:** Given two spatial data sets *S* and *T* and a value *e*, an *e*-distance join retrieves the pairs (*s*,*t*), $s \in S$, $t \in T$, such that the network distance between *s* and *t* is no less than *e* (e.g., find the hotel-restaurant pairs within 10km driving distance).

CHALLENGES AND FUTURE TRENDS

Quite a few challenges are imposed with the existing works on SNDBs. In particular, we discuss two challenges. First, to compare the existing techniques of kNN queries, a thorough analytical

study on the cost model of these query processing techniques will provide the most solid background for their adaptation to real-world applications. Second, it is now necessary to consider the travel time between road intersections as the edge weight for the computation of *k* nearest neighbors. Since updates occur very often in travel-time networks, all the existing query processing algorithms should be reevaluated in terms of very frequent update operations. The pragmatic constraints of the network, such as U-turn restrictions and one-way roads, should also be addressed in the query algorithms.

Topics in the study of LBSs are also influenced by the industry development as well as the varying business models. For instance, the widespread use of GPS (Global Positioning System) devices makes taxi companies use tracking applications to manage taxi cabs, which in turn motivates researchers to study efficient GPS-based tracking techniques. The area of location-based games offers tremendous opportunities for applications and research topics. In the BotFighter game, by the Swedish company It's Alive, players get points for finding and "shooting" other players via their mobile phones (using SMS or WAP). Only players nearby can be shot. To enable the game, players can request the positions of other nearby players. This game motivates researchers to look for efficient ways to manage the positions of moving mobile users.

As for future trends, we list two emerging research topics posed by very recent research articles.

Advanced Trip Planning Queries

The so-called trip planning query (TPQ) proposed by Li, Cheng, Hadjieleftheriou, Kollios, and Teng (2005) is an interesting exploration into more advanced queries in LBSs. Given a starting location, a destination, and a set of points of interest in space, where each point belongs to a specific category, TPQ retrieves the best trip between the start location and the destination, which passes through at least one point from each category. As LBSs become more advanced and intelligent, queries like kNN and range search are too simple to serve the various requests of mobile users. Studying TPQ and similar queries in SNDBs will provide more advanced query prototypes and techniques to support future LBSs.

Road-Network-Based Tracking of Moving Objects

Civilis, Jensen, and Pakalnis (2005) study the tracking of road-network-based moving objects. The topic of tracking the locations of moving objects in road networks comes from applications such as fleet management and traffic monitoring. Current tracking techniques only consider the use of network data (Civilis et al.) and the users' route information (Brilingaite, Jensen, & Zokaite, 2004) to optimize the tracking efficiency and accuracy. The future trend is to consider tracking moving objects with different positioning technologies and to utilize artificial intelligence techniques to improve the tracking efficiency and accuracy.

CONCLUSION

This article summarizes relevant works in the database literature to support mobile LBSs in spatial networks. Existing papers have focused on the data modeling, data structure, and query processing algorithms in spatial network databases. This article considers a prototype service that finds *k* nearest neighbors to a mobile user in the network and illustrates the relevancy of existing papers by an example application framework. As discussed with the future trends, the challenges and novel requirements of LBSs demand more advanced and efficient systems to support the dynamic data in LBSs. A concrete implementation of SNDBs to the current relational database systems will provide a rigorous platform to evaluate the existing techniques and to construct advanced functionalities for LBSs.

REFERENCES

Brilingaite, A., Jensen, C. S., & Zokaite, N. (2004). Enabling routes as context in mobile services. *Proceedings of 12th ACM Symposium on Advances in Geographic Information Systems* (pp. 127-136).

Cho, H. J., & Chung, C. W. (2005). An efficient and scalable approach to CNN queries in a road network. *Proceedings of 31st International Conference on Very Large Databases* (pp. 865-876).

Civilis, A., Jensen, C. S., & Pakalnis, S. (2005). Techniques for efficient road-network-based tracking of moving objects. *IEEE Transactions on Knowledge and Data Engineering, 17*(5), 698-712.

Ding, Z., & Güting, R. H. (2004). Modelling temporally variable transportation networks. *Proceedings of 9th International Conference on Database Systems for Advances Applications* (pp. 154-168).

Güting, R. H., Almeida, V. T. de, & Ding, Z. (2006). Modeling and querying moving objects in networks. *VLDB Journal, 15*(2), 165-190.

Gyssens, M., Paredaens, J., & Gucht, D. van. (1990). A graph-oriented object database model. *Proceedings of 1990 ACM Conference on Principles of Database Systems* (pp. 417-424).

Hage, C., Jensen, C. S., Pedersen, T. B., Speičys, L., & Timko, I. (2003). Integrated data management for mobile services in the real world. *Proceedings of 29th International Conference on Very Large Databases* (pp. 1019-1030).

Huang, X., Jensen, C. S., & Šaltenis, S. (2005). The islands approach to nearest neighbor querying in spatial networks. *Proceedings of the 9th International Symposium on Advances in Spatial and Temporal Databases* (pp. 73-90).

Huang, X., Jensen, C. S., & Šaltenis, S. (2006). Multiple k nearest neighbor query processing in spatial network databases. *Proceedings of 10th East European Conference on Advances in Databases and Information Systems* (pp. 266-281).

Jensen, C. S., Kolář, J., Pedersen, T. B., & Timko, I. (2003). Nearest neighbor queries in road networks. *Proceedings of 11th ACM International Symposium on Advances in Geographic Information Systems* (pp. 1-8).

Kolahdouzan, M., & Shahabi, C. (2004). Voronoi-based nearest neighbor search for spatial network databases. *Proceedings of 30th International Conference on Very Large Databases* (pp. 840-851).

Kolahdouzan, M., & Shahabi, C. (2005). Alternative solutions for continuous K nearest neighbor queries in spatial network databases. *GeoInformatica, 9*(4), 321-341.

Li, F., Cheng, D., Hadjieleftheriou, M., Kollios, G., & Teng, S. H. (2005). On trip planning queries in spatial databases. *Proceedings of 9th International Symposium on Advances in Spatial and Temporal Databases* (pp. 273-290).

Papadias, D., Zhang, J., Mamoulis, N., & Tao, Y. (2003). Query processing in spatial network databases. *Proceedings of 29th International Conference on Very Large Databases* (pp. 802-813).

Šaltenis, S., & Jensen, C. S. (2002). Indexing of moving objects for location-based services. *Proceedings of 18th International Conference on Data Engineering* (pp. 463-472).

Shekhar, S., Kohli, A., & Coyle, M. (1993). Path computation algorithms for advanced traveller information systems. *Proceedings of 9th International Conference on Data Engineering* (pp. 31-39).

Shekhar, S., & Liu, D. R. (1997). CCAM: A connectivity-clustered access method for networks and network computations. *IEEE Transactions on Knowledge and Data Engineering, 9*(1), 102-119.

Speičys, L., Jensen, C. S., & Kligys, A. (2003). Computational data modeling for network con-

strained moving objects. *Proceedings of 11th ACM International Symposium on Advances in Geographic Information Systems* (pp. 118-125).

Vazirgiannis, M., & Wolfson, O. (2001). A spatio temporal model and language for moving objects on road networks. *Proceedings of 7th International Symposium on Advances in Spatial and Temporal Databases* (pp. 20-35).

Yoo, J. S., & Shekhar, S. (2005). In-route nearest neighbor queries. *GeoInformatica, 9*(2), 117-137.

KEY TERMS

Data Point: In location-based services, data points represent points of interest in the spatial network databases. A data point is often represented by linear referencing the edges where the data point can be accessed from. In location-based games where the location of other moving objects is of interest, data points can also be used to model moving objects.

***K*-Nearest Neighbor Query:** Given a set of data points in space and a query point, a *k*-nearest neighbor query finds the *k* data points that are closest to the query point in terms of their distance to the query point. No other data points are closer to the query point than these *k* data points. The distance between a query point and a data point can be computed based on Euclidean distance metrics or network distance (if a spatial network is included).

Location-Based Service: Location-based service is a type of mobile service offered by mobile phone companies or a third-party service provider to mobile phone subscribers. This service sends custom information to the mobile phone subscribers based on their current geographical location. A typical example is when the service sends information about the nearest gas station to a mobile phone user.

Network Distance: For any two locations in a spatial network, their network distance is the length of the shortest path between these two locations along the network. The shortest path is computed based on the travel weight, such as travel distance or travel time, of network edges.

Point of Interest: A point of interest is a geographical site that someone, such as a mobile phone user, may find useful or interesting. Typical points of interest are restaurants, hotels, tourist sites, and so forth.

Query Point: A query point is a location in a spatial network. It represents the current location of a mobile user in reality. Query points are often modeled by linear referencing the network edge where the current location is.

Spatial Network: A spatial network is a network of spatial elements. The instance of spatial network in location-based service is the transportation network, such as the road network. Elements of a spatial network are, for instance, edges that stand for road segments, nodes that stand for intersections of road segments, and points of interest.

Spatial Network Database: A spatial network database is a collection of spatial element data of a spatial network. These data are logically represented by a network data model and physically stored in a network data structure. Database systems provide special data types and functions to support spatial network databases.

Chapter XXXVI
Spatial Data Integration Over the Web

Laura Diaz
Universitat Jaume I, Spain

Carlos Granell
Universitat Jaume I, Spain

Michael Gould
Universitat Jaume I, Spain

INTRODUCTION

Spatial data are increasingly becoming available on the Internet in applications such as routing portals that involve map-based and satellite imagery backgrounds, allowing a large audience to access and share the rich databases that are currently used by the specialized geographic community. These spatial data are heterogeneous, being available in various formats, and stored in disparate formats (flat files, relational or object-oriented databases, etc.). Some data are structured according to well-established data modeling techniques such as the relational or object-oriented data models; other data, such as data maintained in various information systems, spreadsheets, or Internet repositories, are in proprietary formats, semi-structured, or unstructured. In practice, this situation of multiple models and schemas combined with the difficulty for establishment agreements for data representation in the application domains becomes spatial data in special regarding other types of scientific data, making the interoperability problem a nontrivial task (Lemmens, Wytzisk, de By, Granell, Gould & van Oosterom, 2006). In addition to the scale of data integration, the

complex and heterogeneous query processing and domain-specific computational capabilities supported by these sources make spatial data integration a real challenge (Boulcema, Essid & Lacroix, 2002; Devogele, Parent & Spaccapietra, 1998; Goodchild, Egenhofer, Fegeas & Kottman, 1999).

Historically, due to specialized characteristics and the nature of spatial data, geographic information systems (GISs) were managed separately from existing database systems. As first steps to spatial data integration in the mid-1990s, advances in database technology enabled accommodating spatial data in relational databases, allowing organizations to take the first steps toward enterprise GIS and the elimination of organizational "spatial data islands" (ESRI, 2003). Some examples are the appearance of Oracle Spatial (www.oracle.com), PostgreSQL with the PostGIS extension (www.postgresql.org), and MySQL (www.mysql.com). The early work for spatial data integration in database systems focused on sharing simple spatial features in a relational database. Then, standard data manipulation languages such as SQL (Structured Query Language) began to adopt common spatial functionalities to embed, for example, spatial selections and topological queries in SQL statements. The arrival of the first relational models capable of storing both spatial and attribute data led to spatial databases (Rigaux, Scholl & Voisard, 2001), which provided methods for spatial data modeling, algorithms, access methods, and query processing extending traditional database systems.

The success factor of Web services technology has permitted promoting service integration and interoperability among heterogeneous distributed information sources. The GIS approach to service-oriented architecture (SOA) is represented by the Spatial Data Infrastructure (SDI) paradigm, which offers the possibility to access distributed, heterogeneous spatial data through a set of policies, common rules, and standards that facilitate interconnecting spatial information users in an interoperable way (Granell, Gould, Manso & Bernabé, 2007b).

Several approaches and techniques exist for data integration over the Web (Fileto, 2001); however, some of the most representative are the following: *Gateways* as middleware that allows an application running in one DBMS (Data Base Management System) to access data maintained by another DBMS; and *Data Warehouses*, which are separated databases built specifically for decision support, useful when decisions depend on heavy analysis of large amounts of data collected from a variety of possibly heterogeneous data sources. In the GIS domain, most data integration approaches have included one or both of the more common techniques: *wrapper* (Roth & Schwarz, 1997) and *mediator* (Wiederhold, 1992). Wrappers provide interfaces for accessing concrete data sources, retrieving the results, and translating them into a common scheme. Mediators are software components in the middleware in charge of specifying such a common scheme that provides an integrated interface to a set of wrappers; indeed, underlying data sources. Data integration approaches use mediators to handle client queries, submit them directly to the wrappers, and integrate the results before delivering the response to client applications. GIS systems manage data integration mostly by using wrappers and mediators implementing standard interfaces specified by Open Geospatial Consortium (OGC), which provides open standards and specifications for supporting standardized information exchange, sharing, accessing, and processing geospatial data.

The demand for interoperability has boosted the development of standards and tools to facilitate data transformation and integration. Furthermore, this chapter focuses on interface standards as key to spatial data syntactical integration over the Web. Nevertheless, there are still many challenges to be met, especially those concerned with data semantics and harmonization of interoperating systems.

BACKGROUND: SPATIAL DATA INFRASTRUCTURE

Many standards have been proposed for exchanging geographical data among systems (Albrecht, 1999), although they are not enough to enable interoperability because data conversion among these formats often produces information loss. The adoption of a common geographical data model or at least a framework to unify heterogeneous models constitutes one ingredient to achieve GIS interoperability in the wide sense. A first attempt in this way has been featured by the GML standard offering access and retrieval of spatial data in a standard exchange format.

Geography Markup Language (GML) (Cox, Daisay, Lake, Portele & Whiteside, 2002) describes an encoding specification for spatial data in XML that enables the storage, transport, exchange, processing, and transformation of geographic information. GML provides a variety of object types for describing geography, including features, coordinate reference systems, geometry, topology, time, units of measure, and generalized values. Current database implementations (with spatial extensions) permit storing directly GML-based data, thus enabling interacting with other spatial and nonspatial data sources using an international standard. However, implementers also may decide to convert from some other storage format on demand and use GML only for schema and data transport. In this case, spatial data integration over the Web implies geospatial services, which normally exchange GML data, following the SDI architecture.

Spatial Data Infrastructure (SDI) comprises a set of policy and standards activities promoting creation of a geospatial information infrastructure to assist diverse user communities to collect, share, access, and exploit georeferenced information resources. As depicted in Figure 1, traditional SDI vision focused on data is shifting to a service-based vision (Bernard, Craglia, Gould & Kuhn, 2005) in which geospatial services in the middleware are increasingly used to discover and access geospatial data stored on data repositories, transform it into useful information to users, and then deliver it to them.

Although user applications can access directly to spatial data sources as in traditional database systems (see rightmost arrow in Figure 1), they normally use a set of standard geospatial interface services to discover and access spatial data sources in SDI in order to achieve the spatial data integration. Such geospatial service interfaces, mainly published by OGC, act typically as wrappers to add the level of abstraction needed for the syntactic integration of spatial data. The next section introduces some interesting examples of OGC specifications for data integration in the SDI context.

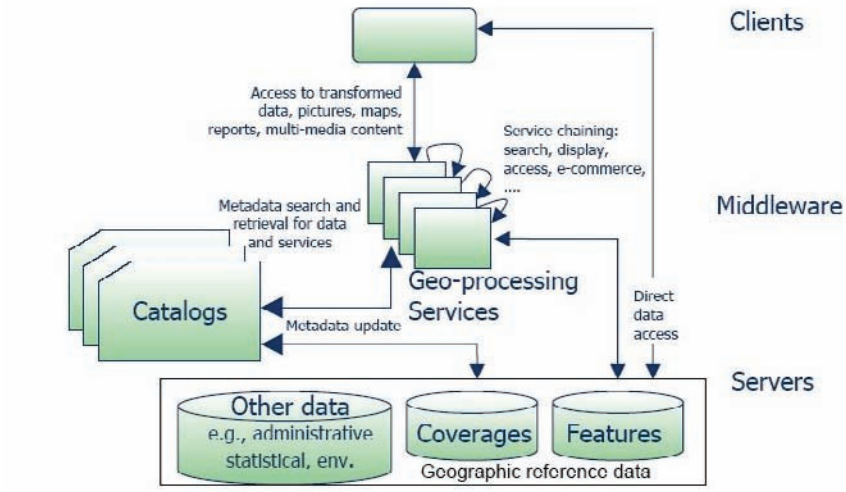
SPATIAL DATA INTEGRATION

Integrating spatial data over the Web implies several components and services linked via The Internet following the SDI architecture. First, this section describes some key service interfaces to spatial data integration. Next, Figure 2 summarizes how such service interfaces are put together to offer spatial access, edition, and processing over heterogeneous and remote data sources.

GEOSPATIAL SERVICE INTERFACES

The OGC Simple Features Access (SFA) is a standard that consists of two parts. The first one describes the common architecture for simple feature geometry (http://www.opengeospatial.org/standards/sfa). A feature has both spatial (geometry valued) and nonspatial attributes. This first part then defines the simple feature geometry object model that is distributed computing platform neutral, which means a common, platform-independent interface to handle geometry

Figure 1. High-level SDI architecture, taken from Smits (2002)



objects in a standard format. All geometry objects, such as *Point*, *Curve*, *Surface*, and *Geometry Collection*, extend the basic functionality of the *Geometry* base class. Each geometric object is also associated with a concrete spatial reference system, which specifies the coordinate space in which the geometric object is present. The second part of the OGC SFA standard (http://www.opengeospatial.org/standards/sfs) is to define standard SQL schema that supports storage, retrieval, query, and update of a geometry object or feature collection, terms already defined in part 1 of this standard.

The OGC Web Feature Service (WFS) (Vretanos, 2005) defines interfaces for data access and manipulation operations on geographic features using HTTP as transport protocol. Via these interfaces, a Web user or service can combine, use, and manage spatial data—the feature information behind a map image—from various data sources by invoking the following WFS operations on geographic features and elements:

- Create a new feature instance
- Delete a feature instance
- Update a feature instance
- Lock a feature instance
- Get or query features based on spatial and nonspatial constraints

The OGC Filter Encoding Implementation Specification (Filter) (http://www.opengeospatial.org/standards/filter) defines an XML encoding for filter expressions to be used in conjunction with other specifications, as the case of WFS services. A filter expression constrains property values to create a subset of a group of objects (features). The goal is to select a determined subset of objects and operate on just such subset by, for example, rendering them in a different color to save them to another format or edit them. An XML encoded filter behaves normally as an SQL *where* structure, because any set of filters could be transformed into an SQL *where* clause for a SQL *select* statement to fetch spatial and nonspatial data stored in any database.

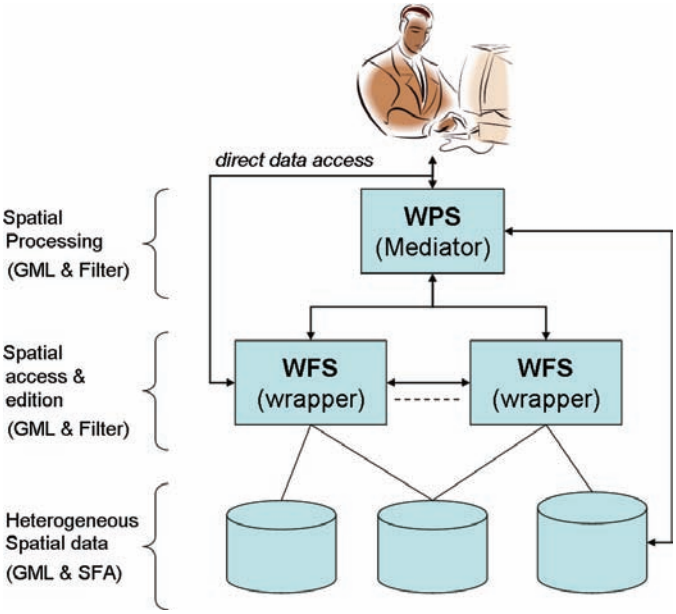
The OGC Web Processing Service (WPS) (Schut, 2007) provides access to calculations or models that operate on spatially referenced data. The data required by the service can be available locally or delivered across a network using data exchange standards such as GML or Geolinked Data Access Service (GDAS). The calculation can be as simple as subtracting one set of spatially referenced numbers from another (e.g., determining the difference in influenza cases between two seasons) or as complicated as a global climate change model. While most OGC specifications and standards are devoted for spatial data abstraction, access, and integration, the WPS specification is about spatial data processing over heterogeneous data sources. The main steps in this process are to identify the spatially referenced data required by the calculation, initiate the calculation, and manage the output from the calculation so it can be accessed by the client. The WPS is targeted at both vector and raster databased processing.

Spatial Access and Edition

As depicted in Figure 2, wrapping spatial databases with OGC WFS Services means that we provide one more level of abstraction to data access and retrieval, since knowing the database-specific model and settings are no longer needed. Indeed, the WFS layer acts as a wrapper to manipulate the underlying data sources and/or spatial databases; it receives requests from remote users, application, or mediators; executes them against the corresponding data sources; and delivers the results to remote users, applications, or mediators.

In most cases, the exchange format of a WFS server is GML. WFS has a rich query interface by using the OGC Filter encoding specification, which describes an XML encoding of the OGC Common Catalogue Query Language (CQL) as a system-neutral representation of a query predicate. As WFS services return GML data, such XML-based representations facilitate data edition by using the numerous XML tools available today.

Figure 2. Components for spatial data integration over the Web



GML data can be easily validated, parsed, edited, and transformed into whatever target language or persistent object.

A WFS service supports *insert, update, delete, query,* and *discovery* operations that are applied to one or more geographic features. A WFS service delivers GML representations of geospatial features in response to queries from HTTP clients. Clients access geographic feature data through WFS by submitting a request for just those features needed for an application.

Spatial Processing

More complex geospatial services have to be specified in order to distribute over the Internet all the functionalities (computation, analysis, etc.) common in our desktop GIS and local data. The first steps toward advanced geoprocessing services online are outlined by the recently published OGC Web Processing Service (WPS) discussion paper (Schut, 2007), which provides interface specifications to enable geospatial Web services to support a limited range of geoprocessing tasks by creating accessible libraries of geoprocessing algorithms under the appearance of geospatial Web service chains (Granell, Gould *Esbrí, 2007a).

WPS services allow users not only to access and visualize distributed data through services but also to realize complex spatial operations (e.g., spatial analysis, shortest path, buffer, etc.). In addition, Figure 2 shows how WPS services may play the role of mediators that provide an integrated interface to various WFS services (wrappers). Client queries containing GML data and optionally filter expressions are handled by WPS services, which submit them directly to the WFS services. WPS services are also in charge of integrating the results (GML) before delivering the response to client applications to perform complex spatial processing. Users or applications can access WFS directly via WFS clients to access and edit the underlying data sources and perform the operations given by the WFS interface.

CHALLENGES AND FUTURE TRENDS

Table 1 lists some challenges that are being solved or will be solved in the future in order to improve spatial data integration over the Web. One of the key areas is that surrounding semantics and spatial data harmonization.

In the semantic context, new approaches and prototypes are emerging (Lemmens et al., 2006; Lutz, 2007; Tanasescu et al., 2007; Tomai & Prastacos, 2006) to improve semantic aspects of spatial data integration by mediating legacy spatial data sources to high-level spatial ontologies through Semantic Web Services (McIlraith, Son & Zeng, 2001). The semantic integration of heterogeneous data requires specific domain knowledge exposed as ontologies that rely on shared concepts, terms, and structuring constructs found in source data. They involve metadata enrichment to support semantic matching among data items from distinct datasets (Stoimenov & Djordjevic-Kajan, 2006). Future developments on spatial data integration with Semantic Web Services will include an increase in the complexity of the integration ontologies in order to allow the unified representation, access, and edition of heterogeneous spatial data sources as envisioning through the *Geospatial* Semantic Web vision (Egenhofer, 2002). This vision requires the development of multiple spatial and terminological ontologies, each with a formal semantics. This will lead to a new framework for geospatial information based on the semantics of spatial and terminological ontologies in which geospatial queries and results are processed semantically. The goal of the Geospatial Semantic Web is to implement components for discovery, querying, and consumption of geospatial content in a distributed architecture, which are based on formal semantic specifications.

Interesting examples of spatial data harmonization are the ongoing projects and current best practice recommendation efforts (e.g., by the INSPIRE Drafting Team on Data Specifica-

Table 1. Challenges and future trends in spatial data integration

• Semantics and data harmonization • Security: access control, authentication • Transactional integrity

tions) devoted to the GMES-INSPIRE vision of seamless interoperability. The GMES (Global Monitoring for Environment and Security) initiative focuses on six application fields: land cover, water resources, ocean/marine applications, atmosphere, risk management, and security. The INSPIRE (Infrastructure for Spatial Information in Europe) infrastructure will then facilitate implementation of services to achieve GMES goals of harmonizing space-based, in situ, and traditional cartographic data sources. In consequence, the GMES program recently funded data harmonization projects such as the RISE project (www.eurogeographics.org/eng/03_RISE.asp) that aim at developing tools and methodologies to enable and facilitate cross-disciplinary interoperability. From this viewpoint, data harmonization pursues the goal of producing geospatial data implementation specifications consistent with international standards encompassing the development of application schema and data product specifications to enable sustainable and interoperable functioning of INSPIRE and GMES.

Other important issues listed in Table 1 are related to security and transactional aspects. Security is used here in the sense of ensuring that communications and transactions among services, both those acting as mediators and as wrappers, are conducted in a protected environment and that messages are reliably delivered to the correct destinations. Other challenges to achieve full interoperability of GIS and related spatial data integration are more related to social and policy restrictions; when accessing distributed data, security and integrity issues will normally arise.

CONCLUSION

GML permits representation of spatial and non-spatial data with an XML-based format, providing spatial services with common data models for spatial data access and interchange. An environment to support data integration must be based on this common data model, which may also be used for data transfer and exchange.

SDIs provide the infrastructure in which spatial wrappers and mediators play a facilitating role. WFS services wrap the data sources, abstracting data from its machine representation, and become accessible to diverse users in a uniform way. As far as spatial processing, WPS services provide an interface to allow not only data access but also data processing and analysis in an interoperable fashion. Adoption of OGC interfaces and standards makes possible spatial data integration in a distributed environment when semantic differences are not too great. Otherwise, future research on semantic interoperability is needed to reach generally acceptable levels of ad hoc spatial data integration.

REFERENCES

Albrecht, J. (1999). Geospatial information standards: A comparative study of approaches in the standardization of geospatial information. *Computers & Geosciences, 25*(1), 9–24.

Bernard, L., Craglia, M., Gould, M., & Kuhn, W. (2005). *Towards an SDI research agenda.* Proceedings of the EC & GIS Workshop, Sardinia.

Boulcema, O., Essid, E., & Lacroix, Z. (2002). *A WFS-based mediation system for GIS interoperability.* Proceedings of the 10th ACM International

Symposium on Advances in Geographic Information Systems, 23–28.

Cox, S., Daisay, P., Lake, R., Portele, C., & Whiteside, A. (Eds.). (2002). OpenGIS geography markup language (GML), Version 3.0. Open Geospatial Consortium. Retrieved from http://www.opengeospatial.org/standards/gml

Devogele, T., Parent, C., & Spaccapietra, S. (1998). On spatial database integration. *Intl. Journal of Geographical Information Science, 12*(4), 335–352.

Egenhofer, M. (2002). *Toward the semantic geospatial Web.* Proceedings of the 10th ACM International Symposium on Advances in Geographic Information Systems, 1–4.

ESRI. (2003). *Spatial data standards and GIS interoperability.* Retrieved from www.isotc211.org/Outreach/Newsletter/Newsletter_04_2004/Appendix_4.pdf

Fileto, R. (2001). *Issues on interoperability and integration of heterogeneous geographical data.* Proceedings of the 3rd Brazilian Symposium on GeoInformatics.

Goodchild, M., Egenhofer, M., Fegeas, R., & Kottman, C. (1999). Interoperating geographic information systems. Norwell: Kluwer Academic Publishers.

Granell, C., Gould, M., & Esbrí, M.A. (2007a). Geospatial Web service chaining. In H. Karimi (Ed.), *Handbook of research on geoinformatics.* Hershey: Information Science Reference.

Granell, C., Gould, M., Manso, M.A., & Bernabé, M.A. (2007b). Spatial data infrastructures. In H. Karimi (Ed.), *Handbook of research on geoinformatics.* Hershey: Information Science Reference.

Lemmens, L, Wytzisk, A., de By, R., Granell, C., Gould, M., & van Oosterom, P. (2006). Integrating semantic and syntactic descriptions to chain geographic services. *IEEE Internet Computing, 10*(5), 42–52.

Lutz, M. (2007). Ontology-based descriptions for semantic discovery and composition of geoprocessing services. *Geoinformatica, 11*(1), 1–36.

McIlraith, S.A., Son, T.C., & Zeng, H. (2001). Semantic Web services. *IEEE Intelligent Systems, 16*(2), 46–53.

Rigaux, P., Scholl, M., & Voisard, A. (2001). *Spatial databases—With applications to GIS.* San Francisco: Morgan Kaufmann.

Roth, M.T., & Schwarz, P. (1997). *Don't scrap it, wrap it! A wrapper architecture for legacy data sources.* Proceedings of the International Conference on Very Large DataBases, 266–275.

Schut, P. (Ed.). (2007). OpenGIS Web processing service specification, version 1.0.0. Open Geospatial Consortium. Retrieved from http://portal.opengeospatial.org/files/?artifact_id=21988&version=1

Smits, P. (Ed.). (2002). INSPIRE architecture and standards position paper. Architecture and Standards Working Group. Retrieved from http://inspire.jrc.it/documents/inspire_ast_pp_v4_3_en.pdf

Stoimenov, L., & Djordjevic-Kajan, S. (2006). *Discovering mappings between ontologies in semantic integration process.* Proceedings of the AGILE Conference on Geographic Information Science, Visegrad, Hungary, 213–219.

Tanasescu, V., et al. (2007). Geospatial data integration with semantic Web services: The eMerges approach. In A. Scharl, & K. Tochtermann (Eds.), *The geospatial Web***. London:** Springer.

Tomai, E., & Prastacos, P. (2006). *A framework for intensional and extensional integration of geographic ontologies.* Proceedings of the AGILE Conference on Geographic Information Science, Visegrad, Hungary, 220–227.

Vretanos, P.A. (2005). Web feature service implementation specification, Version 1.1.0. Open Geospatial Consortium. Retrieved from http://www.opengeospatial.org/standards/wfs

Wiederhold, G. (1992). Mediators in the architecture of future information systems. *IEEE Computer, 25*(3), 38–49.

KEY TERMS

Feature: The fundamental unit of geospatial information. For example, depending on the application, a feature could be any part of the landscape, whether natural (such as a stream or ridge) or artificial (such as a road or power line). A feature object then corresponds to a real-world or abstract entity. Attributes of this feature object describe measurable or describable phenomena about this entity. Feature object instances derive their semantics and valid use or analysis from the corresponding real-world entities' meaning.

GML: Geography Markup Language is an XML grammar defined by OGC to express geographical features. To help users and developers structure and facilitate the creation of GML-based application, GML provides *GML profiles* that are XML schemas that extend the very GML specification in a modular fashion. A GML profile is a GML subset for a concrete context or application but without the need for the full GML grammar, thus simplifying the adoption of GML and facilitating its rapid usage. Some common examples of GML profiles that have been published are *Point Profile* for applications with point geometric data and *GML Simple Features profile*, supporting vector feature requests and responses as the case of WFS.

Mediator: A negotiator who acts as a link between parties, the neutral who carries out the dispute resolution process called mediation.

OGC: Open Geospatial Consortium (http://www.opengeospatial.org), a membership body of 300-plus organizations from the commercial, government, and academic sectors that creates consensus interface specifications in an effort to maximize interoperability among software detailing with geographic data.

SDI: Spatial Data Infrastructure. Many government administrations have initiated coordinated actions to facilitate the discovery and sharing of spatial data, creating the institutional basis for SDI creation. The Global Spatial Data Infrastructure (GSDI) association (http://www.gsdi.org) defines SDI as a coordinated series of agreements on technology standards, institutional arrangements, and policies that enable the discovery and facilitate the availability of and access to spatial data. The SDI, once agreed upon and implemented, serves to connect Geographic Information Systems (GIS) and other spatial data users to a myriad of spatial data sources, the majority of which are held by public sector agencies.

Service: Functionality provided by a service provider through interfaces (paraphrased from ISO 19119—Geographic Information Services).

WFS: The OpenGIS Web Feature Service specification allows a client to retrieve and update geospatial data encoded in Geography Markup Language (GML) from multiple WFS. The specification defines interfaces for data access and manipulation operations on geographic features using HTTP as the distributed computing platform. Via these interfaces, a Web user or service can combine, use, and manage geodata, the feature information behind a map image, from various sources.

Wrapper: A package that changes the interface to an existing package without substantially increasing its functionality.

Section IV
Database Integrity

Chapter XXXVII
Improving Constraints Checking in Distributed Databases with Complete, Sufficient, and Support Tests

Ali Amer Alwan
Universiti Putra Malaysia, Malaysia

Hamidah Ibrahim
Universiti Putra Malaysia, Malaysia

Nur Izura Udzir
Universiti Putra Malaysia, Malaysia

INTRODUCTION

A **database state** is said to be **consistent** if and only if it satisfies the set of integrity constraints. A **database state** may change into a new state when it is updated either by a single **update operation** (insert, delete, or modify) or by a sequence of updates (**transaction**). If a constraint is false in the new state, the new state is inconsistent, therefore the enforcement mechanism can either perform compensatory actions to produce a new consistent state, or restore the initial state by **undo**ing the update operation. The steps generate integrity tests that are **queries** composed from the integrity constraints and the update operations, and run these **queries** against the **database**, which check whether all the integrity constraints of the database are satisfied; are referred to as *integrity checking* (Alwan, Ibrahim, & Udzir, 2007; Ibrahim, 2006; Ibrahim, Gray & Fiddian, 2001), is the main focus of this chapter.

The growing complexity of modern database applications in addition to the need for support of multiple users has further increased the need for a powerful integrity subsystem to be incorporated into these systems. Therefore, a complete integrity subsystem is considered to be an important

part of any modern DBMS. The crucial problem in designing this subsystem is the difficulty of devising an efficient algorithm for enforcing database integrity against updates (Ibrahim, Gray & Fiddian, 2001). Thus, it is not surprising that much attention has been paid to the maintenance of integrity in centralized databases. A naïve approach is to perform the update and then check whether the integrity constraints are satisfied in the new database state. This method, termed *brute force checking*, is very expensive, impractical, and can lead to prohibitive processing costs. Enforcement is costly because the evaluation of integrity constraints requires accessing large amounts of data, which are not involved in the database update transition. Hence, improvements to this approach have been reported in many research papers (Henschen, McCune & Naqvi, 1984; Martinenghi, 2005; McCune & Henschen, 1989; Nicolas, 1982; Qian, 1990; Simon & Valduriez, 1986). The problem of devising an efficient enforcement is more crucial in a distributed environment.

The brute force strategy of checking constraints is worse in the distributed context since the checking would typically require data transfer as well as computation leading to complex algorithms to determine the most efficient approach. Allowing an update to execute with the intension of aborting it at commit time in the event of constraints violation is also inefficient given that rollback and recovery must occur at all sites which participated in the update. Moreover, devising an efficient algorithm for enforcing database integrity against update is extremely difficult to implement and can lead to prohibitive processing costs in a distributed environment (Grefen, 1993; Ibrahim, Gray & Fiddian, 2001). A comprehensive survey on the issues of constraint checking in centralized, distributed, and parallel databases are provided in (Feras, 2005; Ibrahim, 2006).

Works in the area of constraint checking for distributed databases concentrate on improving the performance of the checking mechanism by executing the complete and sufficient tests when necessary. None of the work has to look at the potential of support test in enhancing the performance of the checking mechanism. Also, the previous works claimed that the sufficient test is cheaper than the complete test and its initial integrity constraint. These depend solely on the assumption that the update operation is submitted at the site where the **relations** to be updated is located, which is not necessary the case. Thus, the aim of this chapter is to analyze the performance of the checking process when various types of integrity tests are considered without concentrating on a certain type of test as suggested by previous works. The most suitable test will be selected from the various alternative tests in determining the consistency of the distributed databases. Here, suitable means the test that minimizes the amount of data transferred across the network and the number of sites involved during the process of checking the constraints.

RELATED WORK

For distributed databases, a number of researchers have looked at the problem of semantic integrity checking. Although many research works have been conducted concerning the issues of integrity constraint checking and maintaining in distributed databases but these works are limited due to the fact that (i) they cater limited type of constraints, (ii) their approaches did not use the available information during the process of checking, and (iii) their approaches are able to generate limited type of integrity tests. This is briefly shown in Table 1, where column labeled 1, 2, 3, 4, 5, 6, and 7 represent the work by Simon and Valduriez (1986), Qian (1989), Mazumdar (1993), Gupta (1994), Ibrahim, Gray and Fiddian (2001), Ibrahim (2002), and Madiraju and Sunderraman (2004), respectively.

The work presented in Simon and Valduriez (1986) constructed a simplification method for integrity constraints expressed in terms of as-

Table 1. Summarized of the previous works

Criteria		1	2	3	4	5	6	7
Types of integrity constraints	Domain	√	√	√		√	√	
	Key	√	√	√		√	√	
	Referential	√	√	√	√	√	√	
	Semantic	√	√	√		√	√	√
	Transition						√	
Types of tests	Complete	√	√					
	Sufficient		√	√	√	√	√	√
	Support							

sertions for central databases and extended it to distributed databases. This method produces at assertion definition time, differential pre-tests called compiled assertions, which can be used to prevent the introduction of inconsistencies in the database. The cost of integrity checking is reduced because only data subject to update are checked in this approach.

Qian (1989) argued that most approaches derive simplified forms of integrity constraints from the syntactic structure of the constraints and the update operation without exploiting knowledge about the application domain and the actual implementation of the database. Qian (1989) shows that distributed constraints can be translated into constraints on the fragments of a distributed database, given the definition of the fragmentation, and offers a framework for constraint reformulation. The constraint reformulation algorithm used to derive sufficient conditions can potentially be very inefficient because it searches through the entire space of eligible reformulations for the optimal ones. Using heuristic rules to restrict the reformulation step may miss some optimal reformulation.

The work presented in (Mazumdar, 1993) aims at minimizing the number of sites involved in evaluating the integrity constraints in a distributed environment. In his approach the intention is to reduce the non-locality of constraints by deriving sufficient conditions not only for the distributed integrity constraints given, but also for those arising as tests for particular transactions. His method relies on a standard backchaining approach to find the sufficient conditions.

Gupta (1994) presents an algorithm to generate parameterized local tests that check whether an update operation violates a constraint. This algorithm uses the initial consistency assumption, an integrity constraint assertion that is expressed in a subset of first order logic, and the target relation to produce the local test. This optimization technique allows a global constraint to be verified by accessing data locally at a single database where the modification is made. However, this approach is only useful in situations where each site of a distributed DBMS contains one or more intact relations since it does not consider any fragmentation rules.

Ibrahim, Gray and Fiddian (2001) contribute to the solution of constraint checking in a distributed database by demonstrating when it is possible to derive from global constraints localized constraints. They have proved that examining the semantics of both the tests and the relevant update operations reduces the amount of data

Figure 1. The Company static integrity constraints

Schema:
emp(eno, dno, ejob, esal); dept(dno, dname, mgrno, mgrsal); proj(eno, dno, pno)
Integrity Constraints:
'A specification of valid salary'
IC-1: $(\forall w \forall x \forall y \forall z)(emp(w, x, y, z) \rightarrow (z > 0))$
'Every employee has a unique eno'
IC-2: $(\forall w \forall x1 \forall x2 \forall y1 \forall y2 \forall z1 \forall z2)(emp(w, x1, y1, z1) \wedge emp(w, x2, y2, z2) \rightarrow (x1 = x2) \wedge (y1 = y2) \wedge (z1 = z2))$
'Every department has a unique *dno*'
IC-3: $(\forall w \forall x1 \forall x2 \forall y1 \forall y2 \forall z1 \forall z2)(dept(w, x1, y1, z1) \wedge dept(w, x2, y2, z2) \rightarrow (x1 = x2) \wedge (y1 = y2) \wedge (z1 = z2))$
'The *dno* of every tuple in the *emp* relation exists in the *dept* relation'
IC-4: $(\forall t \forall u \forall v \forall w \exists x \exists y \exists z)(emp(t, u, v, w) \rightarrow dept(u, x, y, z))$
'The *eno* of every tuple in the *proj* relation exists in the *emp* relation'
IC-5: $(\forall u \forall v \forall w \exists x \exists y \exists z)(proj(u, v, w) \rightarrow emp(u, x, y, z))$
'The *dno* of every tuple in the *proj* relation exists in the *dept* relation'
IC-6: $(\forall u \forall v \forall w \exists x \exists y \exists z)(proj(u, v, w) \rightarrow dept(v, x, y, z))$
'Every manager in *dept* '*D*1' earns > £4000'
IC-7: $(\forall w \forall x \forall y \forall z)(dept(w, x, y, z) \wedge (w = `D1') \rightarrow (z > 4000))$
'Every employee must earn ≤ to the manager in the same department'
IC-8: $(\forall t \forall u \forall v \forall w \forall x \forall y \forall z)(emp(t, u, v, w) \wedge dept(u, x, y, z) \rightarrow (w \ ? \ z))$
'Any department that is working on a project P_1 is also working on project P_2'
IC-9: $(\forall x \forall y \exists z)(proj(x, y, P_1) \rightarrow proj(z, y, P_2))$

transferred across the network. The simplified tests have reduced the amount of data that needed to be accessed and the number of sites that might be involved. Ibrahim (2002) extends the work in (Ibrahim, Gray & Fiddian, 2001) by considering the transition constraints.

The work proposed by Madiraju and Sunderraman (2004) focuses on checking the integrity constraints in multiple databases using a mobile agent approach but is limited to general semantic integrity constraints. Other types of integrity constraints that are important and are frequently used in database applications are not being considered.

From these works, it can be observed that most of the previous works proposed an approach to derive simplified form of the initial integrity constraint with the sufficiency property, since the sufficient test is known to be cheaper then the complete test and its initial integrity constraint as it involved less data to be transferred across the network and always can be evaluated at the target site, i.e. only one site will be involved during the checking process. The previous approaches assume that an update operation will be executed at a site where the relation specified in the update operation is located, which is not always true. For example, consider a relation *R* that is located at site 1. An insert operation into *R* is assume to be submitted by a user at site 1 and the sufficient test generated is used to validate the consistency of the database with respect to this update operation, which can be performed locally at site 1. But if the same update operation is submitted at different site, say 2, the sufficient test is no longer appropriate as it will definitely access information from site 1 which is now remote to site 2. Therefore, an approach is needed so that local checking can be performed regardless the location of the submitted update operation. Also, the approach must be able to cater the important and frequently used integrity constraint types. Throughout this paper the *company* database is used, as given in Figure 1. This example has been used in most previous works related to the area of constraint checking (Gupta, 1994; Ibrahim, 2006; Ibrahim, Gray & Fiddian, 2001).

TYPES OF INTEGRITY TESTS

In the database literature, many types and variations of integrity tests have been described. The classifications of integrity tests are based on some of their characteristics, as presented in Table 2. This is explained below:

a. Based on when the integrity test is evaluated: (i) *post-tests* - allow an **update operation** to be executed on a database state, which changes it to a new state, and when an inconsistent result is detected **undo** this update. The method that applies these integrity tests is called the detection method. (ii) *pre-tests* - allow an update to be executed only if it changes the database state to a consistent state. The method that applies these integrity tests is called the prevention method.

b. Based on region: (i) *local tests* – verify the consistency of a database within the local region, i.e. by accessing the information at

Table 2. Types of integrity tests in distributed databases

Integrity test based on input	Integrity test based on region	Integrity test based on detection/ prevention methods	Integrity test based on its properties
Non-support test	Global test – spans remote sites	Post-test – evaluated after an update is performed	Sufficient test
			Necessary test
			Complete test
		Pre-test – evaluated before an update is performed	Sufficient test
			Necessary test
			Complete test
	Local test – spans local sites	Post-test – evaluated after an update is performed	Sufficient test
			Necessary test
			Complete test
		Pre-test – evaluated before an update is performed	Sufficient test
			Necessary test
			Complete test
Support test	Global test – spans remote sites	Post-test – evaluated after an update is performed	Sufficient test
			Necessary test
			Complete test
		Pre-test – evaluated before an update is performed	Sufficient test
			Necessary test
			Complete test
	Local test – spans local sites	Post-test – evaluated after an update is performed	Sufficient test
			Necessary test
			Complete test
		Pre-test – evaluated before an update is performed	Sufficient test
			Necessary test
			Complete test

the local site. The method that adopts these integrity tests is called the *local method.* (ii) *global tests* - verify the consistency of a database outside the local region, i.e. by accessing the information at the remote site(s). The method that adopts these integrity tests is called the *global method*.

c. Based on its properties (McCarroll, 1995): (i) *sufficient tests* - when the test is satisfied, this implies that the associated constraint is satisfied and thus the update operation is safe with respect to the constraint. (ii) *necessary tests* - when the test is not satisfied, this implies that the associated constraint is violated and thus the update operation is unsafe with respect to the constraint. (iii) *complete tests* - has both the sufficiency and the necessity properties.

d. Based on the input used to generate the test: (i) *non-support tests* – these integrity tests are generated based on the update operation and the integrity constraint to be checked. (ii) *support tests* - any tests that are derived using other integrity constraints as the support to generate the tests. Both the non-support tests and support tests can have the characteristics as mentioned above, namely: pre-test or post-test, depending on when the test is evaluated, local-test or global test, depending on the region it covers during the evaluation, and complete, sufficient or necessary depending on the properties of the test.

Table 3 summarizes the integrity tests generated for the integrity constraints listed in Figure 1 using the simplification method proposed by us. It is not the intention of this paper to present the simplification method as readers may refer to (Alwan, Ibrahim & Udzir, 2007). Based on Table 3, integrity tests 1, 2, 5, 8, 11, 12, 15, 16, 19, 20, 21, 23, 24 and 26 are the complete tests, while integrity tests 9, 13, 17, 22, 25 and 27 are the sufficient tests, where these tests are derived using the **update operation** and the integrity constraint to be checked as the input. Thus, they are the non-support tests. Support tests are 3, 4, 6, 7, 10, 14 and 18, where these tests are generated based on the update operation and other integrity constraint as the support. For example test 3 and test 18 are generated using *IC*-5 as the support.

For the following analysis consider the worst case scenario where the *emp*, *dept* and *proj* tables are located at three different sites, *S*1, *S*2 and *S*3, respectively. We assume that there are 500 employees (500 tuples), 10 departments (10 tuples) and 100 projects (100 tuples). We also assume that the following update operations are submitted by three different users at three different sites.

a. A user *A* submits an insert operation, insert(*emp*(*E*1, *D*4, *J*5, 4500)) at *S*1.
b. A user *B* submits an insert operation, insert(*emp*(*E*2, *D*4, *J*6, 3000)) at *S*2.
c. A user *C* submits an insert operation, insert(*emp*(*E*3, *D*3, *J*5, 4500)) at *S*3.

The integrity constraints that might be violated by these update operations are *IC*-1, *IC*-2, *IC*-4 and *IC*-8. This is based on the well-known update theorem (Nicolas, 1982).

Table 4 compares the initial integrity constraints and the various types of integrity tests with regards to the amount of data transferred across the network and the number of sites involved for the simple update operations and allocation as given in (a), (b) and (c) above. The parameters and the measurement used in this comparison are taken from Ibrahim, Gray, and Fiddian (2001). The symbol δR denotes the size of the relation *R*.

a. T provides an estimate of the amount of data transferred across the network. It is measured based on the following formula, $T = \sum_{i=1}^{n} dt_i$, where dt_i is the amount of data transferred from site *i* to the target site.
b. σ gives a rough measurement of the amount of nonlocal access necessary to evaluate

Table 3. Integrity tests of the integrity constraints of figure 1

IC-i	Update template	Integrity test
IC-1	insert(*emp*(*a*, *b*, *c*, *d*))	1. $d > 0$
IC-2	insert(*emp*(*a*, *b*, *c*, *d*))	2. $(\forall x2 \forall y2 \forall z2)(\neg emp(a, x2, y2, z2) \vee (x2 \neq b) \vee (y2 \neq c) \vee (z2 \neq d))$
		3. $(\forall y \forall z)(\neg proj(a, y, z))$
		4. $(\forall x \forall y \forall z)(\neg dept(x, y, a, z))$
IC-3	insert(*dept*(*a*, *b*, *c*, *d*))	5. $(\forall x2 \forall y2 \forall z2)(\neg dept(a, x2, y2, z2) \vee (x2 \neq b) \vee (y2 \neq c) \vee (z2 \neq d))$
		6. $(\forall x \forall y \forall z)(\neg emp(x, a, y, z))$
		7. $(\forall y \forall z)(\neg proj(y, a, z))$
IC-4	insert(*emp*(*a*, *b*, *c*, *d*))	8. $(\exists x \exists y \exists z)(dept(b, x, y, z))$
		9. $(\exists t \exists v \exists w)(emp(t, b, v, w))$
		10. $(\exists y \exists z)(proj(y, b, z))$
	delete(*dept*(*a*, *b*, *c*, *d*))	11. $(\forall t \forall v \forall w)(\neg emp(t, a, v, w))$
IC-5	insert(*proj*(*a*, *b*, *c*))	12. $(\exists x \exists y \exists z)(emp(a, x, y, z))$
		13. $(\exists v \exists w)(proj(a, v, w))$
		14. $(\exists x \exists y \exists z)(dept(x, y, a, z))$
	delete(*emp*(*a*, *b*, *c*, *d*))	15. $(\forall v \forall w)(\neg proj(a, v, w))$
IC-6	insert(*proj*(*a*, *b*, *c*))	16. $(\exists x \exists y \exists z)(dept(b, x, y, z))$
		17. $(\exists u \exists w)(proj(u, b, w))$
		18. $(\exists x \exists y \exists z)(emp(x, b, y, z))$
	delete(*dept*(*a*, *b*, *c*, *d*))	19. $(\forall u \forall w)(\neg proj(u, a, w))$
IC-7	insert(*dept*(*a*, *b*, *c*, *d*))	20. $(\forall x \forall y \forall z)(\neg dept(a, x, y, z) \vee (a \neq \text{`}D1\text{'}) \vee (d > 4000))$
IC-8	insert(*emp*(*a*, *b*, *c*, *d*))	21. $(\forall x \forall y \forall z)(\neg dept(b, x, y, z) \vee (d \leq z))$
		22. $(\exists t \exists v \exists w)(emp(t, b, v, w) \wedge (w \geq d))$
	insert(*dept*(*a*, *b*, *c*, *d*))	23. $(\forall t \forall v \forall w)(\neg emp(t, a, v, w) \vee (w \leq d))$
IC-9	insert(*proj*(*a*, *b*, *P*1))	24. $(\exists z)(proj(z, b, P2))$
		25. $(\exists z)(proj(z, b, P1))$
	delete(*proj*(*a*, *b*, *P*2))	26. $(\forall x)(\neg proj(x, b, P1))$
		27. $(\exists z)(proj(z, b, P2) \wedge (z \neq a))$

a constraint or simplified form. This is measured by analysing the number of sites that might be involved in validating the constraint.

Figures 2-4 present the performance of the checking mechanisms with respect to the amount of data transferred across the network for the update operations (a), (b) and (c) discussed above when various combinations of integrity tests are selected, as follows:

- *IC*: The initial integrity constraints are checked to identify the existence of incon-

*Table 4. Estimation of the amount of data transferred (*T*) and the number of sites involved (σ)*

Update operation, *UO*	Integrity constraint, *IC*	The amount of data transferred, T	The number of sites, σ	Integrity test, *IT*	The amount of data transferred, T	The number of sites, σ	Type of integrity test
(a)	*IC*-1	0	1	1	0	1	Local complete test
	IC-2	0	1	2	0	1	Local complete test
				3	δ*proj*	2	Global support necessary test
				4	δ*dept*	2	Global support necessary test
	IC-4	δ*dept*	2	8	δ*dept*	2	Global complete test
				9	0	1	Local sufficient test
				10	δ*proj*	2	Global support sufficient test
	IC-8	δ*dept*	2	21	δ*dept*	2	Global complete test
				22	0	1	Local sufficient test
(b)	*IC*-1	δ*emp*	1	1	0	1	Local complete test
	IC-2	δ*emp*	2	2	δ*emp*	2	Global complete test
				3	δ*proj*	2	Global support necessary test
				4	0	1	Local support necessary test
	IC-4	δ*emp*	2	8	0	1	Local complete test
				9	δ*emp*	2	Global sufficient test
				10	δ*proj*	2	Global support sufficient test
	IC-8	δ*emp*	2	21	0	1	Local complete test
				22	δ*emp*	2	Global sufficient test
(c)	*IC*-1	δ*emp*	1	1	0	1	Local complete test
	IC-2	δ*emp*	2	2	δ*emp*	2	Global complete test
				3	0	1	Local support necessary test
				4	δ*dept*	2	Global support necessary test
	IC-4	δ*emp* + δ*dept*	3	8	δ*dept*	2	Global complete test
				9	δ*emp*	2	Global sufficient test
				10	0	1	Local support sufficient test
	IC-8	δ*emp* + δ*dept*	3	21	δ*dept*	2	Global complete test
				22	δ*emp*	2	Global sufficient test

sistency in the distributed databases. This method is termed the brute force checking.

- *CT*: Only the complete tests are selected.
- *CT/ST*: Either the complete test or the sufficient test is selected depending on which test will minimize the amount of data transferred and the number of sites involved, for a given integrity constraint. This approach is as proposed by previous works (Gupta, 1994; Ibrahim, 2006; Ibrahim, 2002; Ibrahim, Gray & Fiddian, 2001; Madiraju & Sunderraman, 2004; Mazumdar, 1993; Qian, 1989; Simon & Valduriez, 1986).
- *CT/ST/SupT*: Either the compete test, the sufficient test or the support test is selected, depending on which test will minimize the amount of data transferred and the number of sites involved, for a given integrity constraint. This is what we proposed.

Figure 2 shows that the amount of data transferred across the network for update operation (a) is reduced for all the constraints (*IC*-1, *IC*-2, *IC*-4 and *IC*-8) even though only the complete test is selected and it is further reduced when both the complete and sufficient tests are considered. For this case, no data is being transferred, i.e. the integrity tests selected can be evaluated at the target site and the number of sites involved is 1. Note also, this case represents the assumption made by the previous researches where the update operation is submitted at the site where the relation is located. Hence, for the update operation (a), integrity tests 1, 2, 9 and 22 should be selected.

Figure 3 shows that the amount of data transferred across the network for update operation (b) is reduced for most of the constraints (*IC*-1, *IC*-4 and *IC*-8) when the complete test is considered. No further improvement can be seen even when both complete and sufficient tests are considered. The amount of data transferred for *IC*-2 is the same for the initial constraint, complete and sufficient tests. But this is further reduced when the support test is considered. For this case, no data is being transferred, i.e. the integrity tests selected can be evaluated at the target site and the number of sites involved is 1. Hence, for the update operation (b), integrity tests 1, 4, 8 and 21 should be selected.

Figure 4 shows that the amount of data transferred across the network for update operation (c) is reduced for most of the constraints (*IC*-1, *IC*-4 and *IC*-8) when the complete tests are considered. The amount of data transferred is still the same even when both the complete and sufficient tests are being considered. But this is further reduced when the support test is considered. For this case,

Figure 2. The amount of data transferred for update operation (a)

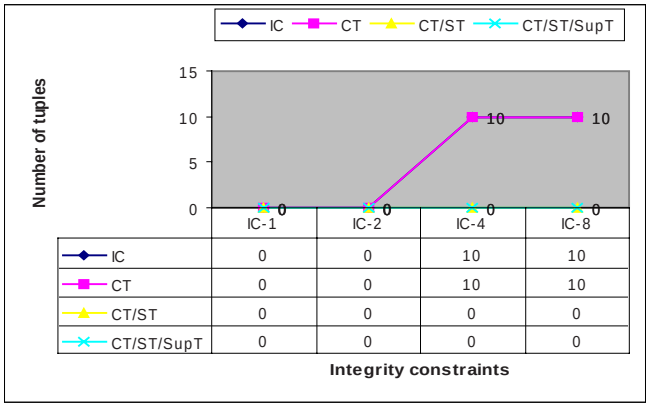
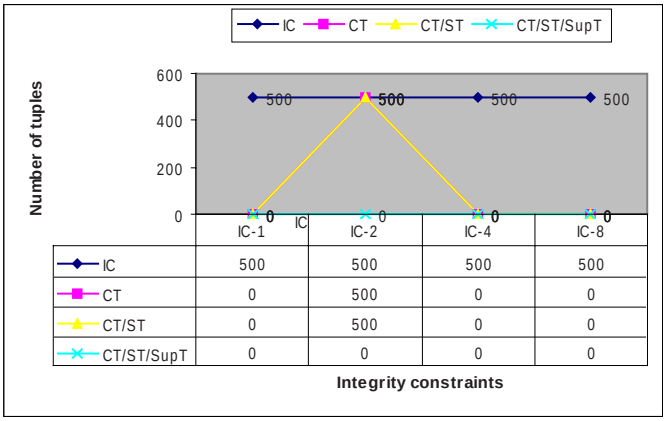
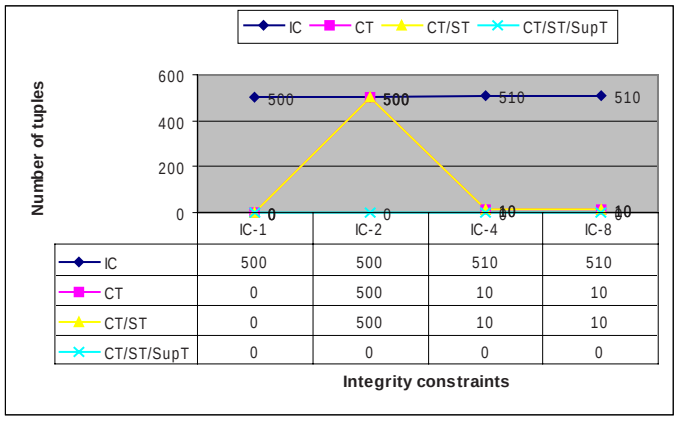


Figure 3. The amount of data transferred for update operation (b)



	IC-1	IC-2	IC-4	IC-8
IC	500	500	500	500
CT	0	500	0	0
CT/ST	0	500	0	0
CT/ST/SupT	0	0	0	0

Figure 4. The amount of data transferred for update operation (c)



	IC-1	IC-2	IC-4	IC-8
IC	500	500	510	510
CT	0	500	10	10
CT/ST	0	500	10	10
CT/ST/SupT	0	0	0	0

no data is being transferred, i.e. the integrity tests selected can be evaluated at the target site for *IC*-1, *IC*-2 and *IC*-4, and the number of sites involved is 1. Hence, for the update operation (c), integrity tests 1, 3, 10 and 21 should be selected.

From the analysis, it is clear that local checking can be achieved by not only selecting sufficient test as suggested by previous works (Gupta, 1994; Ibrahim, 2006; Ibrahim, 2002; Ibrahim, Gray & Fiddian, 2001; Madiraju & Sunderraman, 2004; Mazumdar, 1993; Qian, 1989; Simon & Valduriez, 1986) but also by evaluating the sufficient, complete or support tests depending on the information that is available at the target site. We have also showed that necessary, complete, sufficient and support tests can be local or global tests depending on where the update operation is submitted and the location of the relation(s) specified in the integrity tests (refer to Table 4). Most importantly, we have proved that in some cases, support tests can benefit the distributed database, as shown by update operation (b) with test 4 (*IC*-2) and update operation (c) with tests 3 (*IC*-2) and 10 (*IC*-4), where local constraint check-

ing can be achieved. This simple analysis shows that applying different type of integrity tests will give different impacts to the performance of the constraint checking. Integrating these various types of integrity tests during constraint checking and not concentrating on certain types of integrity tests (as suggested by previous works) can enhance the performance of the constraint mechanisms. Thus, developing an approach that can minimize the number of sites involved and the amount of data transferred across the network during the process of checking the integrity of the distributed databases is important.

FUTURE TRENDS

Although the topic of checking integrity constraints has been explored since the middle of 1970s (Eswaran & Chamberlin; 1975), it is still one of the main topics discussed in today's conferences and workshops (e.g. the International Conference on Database and Expert Systems Applications – the Second International Workshop on Logical Aspects and Applications of Integrity Constraints (LAAIC'06), Krakow (Poland), 8 September 2006). The issues as discussed in the previous section are still debated. In the future it is predicted that new technologies especially those employed in the Artificial Intelligent field such as agents, data mining and fuzzy set will be adopted in deriving and checking constraints, especially in mobile databases.

Mobile agents which are autonomous active programs that can move both data and functionality (code) to multiple places in a distributed system are now getting popular as a technique for checking databases' constraints. It is believed that mobile agents can be used to speed up the process of checking integrity constraints (Mazumdar & Chrysanthis, 2004; Madiraju & Sunderraman, 2004).

Data mining, on the other hand, can be used to identify the pattern that occurs in a database to derive the relationships between the data. Based on these relationships, integrity constraints can be generated automatically. This can reduce the burden of the user in specifying the correct and complete set of business rules. While, fuzzy set is adopted to identify the range of possible **consistent** values in which consistency of the database is flexible and not as rigid as the traditional approach.

CONCLUSION

In a distributed database, the cost of accessing remote data for verifying the consistency of the database is the most critical factor that influences the performance of the system (Ibrahim, Gray & Fiddian, 2001). In this paper, we have showed and proved through a simple analysis that selecting the suitable type of test can benefit the distributed database system, in which the amount of data transferred across the network is significantly reduced and the number of sites involved is always 1.

REFERENCES

Alwan, A. A., Ibrahim, H., & Udzir, N. I. (2007). Local integrity checking using local information in a distributed database. *Proceedings of the 1st Aalborg University IEEE Student Paper Contest 2007 (AISPC'07).* Aalborg, Denmark.

Eswaran, K. P., & Chamberlin, D. D. (1975). Functional specifications of a subsystem for database integrity. *Proceedings of the 1st International Conference on Very Large Data Bases (VLDB 1), 1*(1), 48-68.

Feras, A. H. H. (2005). Integrity constraints maintenance for parallel databases. Ph.D. Thesis, Universiti Putra Malaysia, Malaysia.

Grefen, P. W. P. J. (1993). Combining theory and practice in integrity control: A declarative approach to the specification of a transaction modification subsystem. *Proceedings of the 19th International Conference on Very Large Data Bases*, Dublin, (pp. 581-591).

Gupta, A. (1994). *Partial information based integrity constraint checking.* Ph.D. Thesis, Stanford University.

Henschen, L. J., McCune, W. W., & Naqvi, S. A. (1984). Compiling constraint-checking programs from first-order formulas. *Advances in Database Theory, 2*, 145-169, Plenum Press.

Ibrahim, H. (2006). Checking integrity constraints – How it differs in centralized, distributed and parallel databases. *Proceedings of the Second International Workshop on Logical Aspects and Applications of Integrity Constraints,* Krakow, Poland, (pp. 563-568).

Ibrahim, H. (2002). A strategy for semantic integrity checking in distributed databases. *Proceedings of the Ninth International Conference on Parallel and Distributed Systems.* Republic of China, IEEE Computer Society.

Ibrahim, H., Gray, W. A., & Fiddian, N. J. (2001). Optimizing fragment constraints – A performance evaluation. *International Journal of Intelligent Systems – Verification and Validation Issues in Databases, Knowledge-Based Systems, and Ontologies, 16*(3), 285-306. John Wiley & Sons Inc.

Madiraju, P., & Sunderraman, R. (2004). A mobile agent approach for global database constraint checking. *Proceedings of the ACM Symposium on Applied Computing,* Nicosia, Cyprus, (pp. 679-683).

Martinenghi, D. (2005). Advanced techniques for efficient data integrity checking. Ph.D. Thesis, Roskilde University.

Mazumdar, S. (1993). Optimizing distributed integrity constraints. *Proceedings of the 3rd International Symposium on Database Systems for Advanced Applications,* Taejon, *4*, 327-334.

Mazumdar, S., & Chrysanthis, P. K. (2004). Localization of integrity constraints in mobile databases and specification in PRO-MOTION. *Proceedings of the Mobile Networks and Applications,* (pp. 481-490).

McCarroll, N. J. (1995). *Semantic integrity enforcement in parallel database machines.* Ph.D. Thesis, University of Sheffield.

McCune, W. W., & Henschen, L. J. (1989). Maintaining state constraints in relational databases: A proof theoretic basis. *Journal of the Association for Computing Machinery, 36*(1), 46-68.

Nicolas, J. M. (1982). Logic for improving integrity checking in relational databases. *Acta Informatica, 18*(3), 227-253.

Qian, X. (1990). Synthesizing database transaction. *Proceedings of the 16th International Conference on Very Large Data Bases,* Brisbane, (pp. 552-565).

Qian, X. (1989). Distribution design of integrity constraints. *Proceedings of the 2nd International Conference on Expert Database Systems*, USA, 205-226.

Simon, E. and Valduriez, P. (1986). Integrity control in distributed database systems. *Proceedings of the 19th Hawaii International Conference on System Sciences*, Hawaii, (pp. 622-632).

KEY TERMS

Consistent Database State: The state of the database in which all constraints in the set of integrity constraints are satisfied.

Constraint Checking: The process of ensuring that the integrity constraints are satisfied by the database after it has been updated.

Distributed DBMS: The actual database and DBMS software distributed over many sites, connected by a computer network.

Integrity Constraints: A formal representation of a property that a database is required to satisfy at any time in order to faithfully describe the real world represented by the database.

Integrity Control: Deals with the prevention of semantic errors made by the users due to their carelessness or lack of knowledge.

Integrity Tests: Integrity checks that verify whether the database is satisfying its constraints or not.

Local Constraint Checking: The process of ensuring that the integrity constraints are satisfied by the database after it has been updated is performed at the site where the update is being executed.

Chapter XXXVIII
Inconsistency-Tolerant Integrity Checking

Hendrik Decker
Instituto Technológico de Informática, Spain
Ciudad Politécnica de la Innovación, Spain

Davide Martinenghi
Politecnico di Milano, Italy

INTRODUCTION

Integrity checking has been a perennial topic in almost all database conferences, journals, and research labs. The importance of the issue is testified by very large amounts of research activities and publications. They are motivated by the fact that integrity checking is practically unfeasible for significant amounts of stored data without a dedicated approach to optimize the process. Basic early approaches have been extended to deductive, object-relational, XML- (extensible markup language) based, distributed, and other kinds of advanced database technology. However, the fundamental ideas that are already present in the seminal paper (Nicolas, 1982) have not changed much.

The basic principle is that, in most cases, a so-called simplification, that is, a simplified form of the set of integrity constraints imposed on the database, can be obtained from a given update (or just an update schema) and the current state of the database (or just the database schema). Thus, integrity, which is supposed to be an invariant of all possible database states, is checked upon each update request, which in turn is authorized only if the check of the simplification yields that integrity is not violated. Here, *simplified* essentially means *more efficiently evaluated* at update time. A general overview of the field of simplified integrity checking is provided in Martinenghi, Christiansen, and Decker (2006).

A common point of view by which the need for integrity checking is justified can be characterized as follows. Whenever a database contains erroneous, unwanted, or faulty information, that is, data that violate integrity, answers to queries cannot be trusted. Hence, simplification methods for integrity checking usually address this issue in a very drastic way: In order to avoid possibly wrong answers that are due to integrity violation, incorrect stored data that cause inconsistency need to be completely prevented. However, this drastic

attitude is most often unrealistic: The total absence of unwanted, incorrect, or unexpected data is definitely an exception in virtually all real-world scenarios. Still, it is desirable to preserve the good data in the database while preventing more bad ones from sneaking in and, thus, further diminish the trustworthiness of answers to queries.

The intolerant attitude of the simplification approach of integrity checking toward data that violate integrity is reflected in Nicolas (1982) and virtually all publications on the same subject that came after it. They all postulate the categorical premise of total integrity satisfaction, that is, that each constraint must be satisfied in the old database state, given when an update is requested but not yet executed. Otherwise, correctness of simplification is not guaranteed.

As opposed to the attention granted to integrity checking in academia, support for the declarative specification and efficient evaluation of semantic integrity in practical systems has always been relatively scant, apart from standard constructs such as constraints on column values, or primary and foreign keys in relational database tables. Various reasons have been identified for this lack of practical attention. Among them, the logically abstract presentation of many of the known simplification methods is often mentioned. Here, we focus on another issue of integrity checking that we think is even more responsible for a severe mismatch between theory and practice: Hardly any database ever is in a perfectly consistent state with regard to its intended semantics. Clearly, this contradicts the fundamental premise that the database must always satisfy integrity. Thus, due to the intolerance of classical logic with respect to inconsistency, integrity checking is very often not considered an issue of practical feasibility or even relevance.

Based on recent research results, we are going to argue that inconsistency is far less harmful for database integrity than as suggested by commonly established results. We substantiate our claim by showing that, informally speaking, the consistent part of a possibly inconsistent database can be preserved across updates. More precisely, we show that, if the simplified form of an integrity theory is satisfied, then each instance of each constraint that has been satisfied in the old state continues to be satisfied in the new updated state, even if the old database is not fully consistent. Therefore, such an approach can rightfully be called inconsistency tolerant. Yet, we are also going to see that the use of inconsistency-tolerant integrity checking methods prevents an increase of inconsistency and may even help to decrease it.

BACKGROUND

Throughout, we refer to the relational framework of deductive databases, that is, relational databases, with possibly recursive view definitions described in clause form (Abiteboul, Hull, & Vianu, 1995). Thus, a database consists of a set of facts and a set of rules, that is, tuples and view definitions, respectively, in the terminology of the relational model.

An integrity constraint (or, shortly, constraint) expresses a semantic invariant, in other words, a condition that is supposed to hold in each state of the database. In general, it can be expressed by any closed first-order logic formula in the language of the database on which it is imposed. Usually, without loss of generality, constraints are either represented in prenex normal form (i.e., with all quantifiers moved leftmost and all negation symbols moved innermost) or as denials (i.e., datalog clauses with empty heads). Such a denial expresses that, if its condition is satisfied, then integrity is violated. An integrity theory is a finite set of constraints.

We limit ourselves to databases with a unique standard model. For a closed formula W and a database D, we write $D \models W$ (resp., $D \not\models W$) to indicate that W evaluates to true (resp., false) in the standard model of D. For a set of formulas Γ, we write $D \models \Gamma$ (resp., $D \not\models \Gamma$) to indicate that, for each (resp., some) formula W in Γ, we have $D \models W$ (resp., $D \not\models W$). For a constraint W and an integrity theory Γ, it is also usual to say that D satisfies (resp., violates) W and Γ, respectively.

Also, D is said to be consistent with an integrity theory Γ if and only if $D \models \Gamma$. Moreover, let & denote logical conjunction.

Database states are determined and changed by atomically executed updates. An update U is a mapping from the space of databases as determined by a fixed but sufficiently large language into itself. In the deductive case, each update can be realized by an atomic transaction consisting of soundly sequentializable insertions and deletions of clauses. Conveniently, for a database D, let D^U denote the new database state obtained by applying U on the old state D.

Given a set Γ of constraints, a database D consistent with Γ, and an update U, the integrity checking problem asks whether $D^U \models \Gamma$ holds, that is, whether the new database satisfies the constraints. However, evaluating Γ in D^U may be prohibitively expensive. So, a reformulation of the problem is called for that attempts to simplify the evaluation by taking advantage of the incrementality of updates. As already mentioned, all such reformulations have been made under the assumption that the old state is consistent.

Two kinds of such reformulations have been often discussed in the literature, subsequently called posttest and pretest. Both determine an alternative integrity theory, the evaluation of which is supposed to be simpler than to evaluate the original integrity theory Γ, while yielding equivalent results. Therefore, post- and pretests often are also referred to, generically, as simplifications. Posttests are to be evaluated in the new state, and pretests in the old state.

Often, an orthogonal distinction is made between simplifications that are obtained with or without taking the database into account, or only its view definitions but not the extensions of its base predicates. That is, at least sometimes, some methods only take Γ and U and the view definitions of D into account but not the extension of D, or only Γ and U, or even only Γ. In this exposition, we refrain from such distinctions since inconsistency tolerance is independent thereof. In fact, it is also independent of using either pre- or posttests. Nevertheless, we expatiate this distinction because pretests are less common but arguably more efficient whenever applicable (c.f. Christiansen & Martinenghi, 2006).

Informally speaking, the process of integrity checking involving a posttest consists of executing the update, checking the posttest, and, if it fails to hold, correcting the database by performing a repair action, that is, a rollback and optionally a modification of the update that does not violate integrity. Approaches of this kind are described, for example, in Nicolas (1982), Decker (1987), Lloyd, Sonenberg, and Topor (1987), Sadri and Kowalski (1988), and Leuschel and de Schreye (1998). A more abstract formal definition follows.

Definition (Posttest)

Let D be a database, Γ an integrity theory, and U an update. An integrity theory Δ is a posttest of Γ for D and U (or simply a posttest of Γ for U if D is understood or irrelevant) if, whenever $D \models \Gamma$, the following holds:

$D^U \models \Gamma$ iff $D^U \models \Delta$.

Box 1.

$\Delta = \{$ $\leftarrow sub(c,b)$, % b did not submit c
$\leftarrow rev(c,a)$, % a does not review c
$\leftarrow pub(P,b)$ & $pub(P,a)$, % b is not coauthor of a
$\leftarrow sub(c,A)$ & $pub(P,b)$ & $pub(P,A)$ % b is not coauthor of an author of c
$\leftarrow rev(c,R)$ & $pub(P,R)$ & $pub(P,a)$ % c is not reviewed by a coauthor of a $\}$.

Clearly, Γ itself is a posttest of Γ for any database and any update. But, as indicated, one is interested in producing a posttest that is actually simpler to evaluate than the original constraints. Traditionally, that is achieved by exploiting the fact that the old state D is consistent with Γ, thus avoiding redundant checks of cases that are already known to satisfy integrity.

As already mentioned, the second kind of approach to incremental integrity checking by reformulating a given integrity theory Γ is to determine a simplified integrity theory Δ to be evaluated in the old state. In other words, pretests allow one to predict without actually executing the update whether the new, updated state will be consistent with the constraints. Hence, not only the consistency of the old state D with Γ is exploited, but also the fact that inconsistent states can be prevented without executing the update, and, thus, without ever having to undo a violated new state. Informally speaking, the integrity checking process involving a pretest is to therefore check whether the pretest holds and, if it does, execute the update. Examples of pretest-based approaches are (Christiansen & Martinenghi, 2006; Qian, 1988). A more abstract formal definition follows.

Definition (Pretest)

Let D be a database, Γ an integrity theory, and U an update. An integrity theory Δ is a pretest of Γ for D and U (or simply a pretest of Γ for U if D is understood or irrelevant) if, whenever $D \models \Gamma$, the following holds:

$D^U \models \Gamma$ iff $D \models \Delta$.

For convenience, to say, for a simplification Δ, that the evaluation of Δ yields *satisfied* (or *violated*) means that $D^U \models \Delta$ (or, resp., $D^U \not\models \Delta$) if Δ is a posttest and $D \models \Delta$ (or, resp., $D \not\models \Delta$) if Δ is a pretest. We conclude this section with an example of a simplification.

Example 1

Consider a database with the relations *rev*(*S*,*R*) (submission *S* assigned to reviewer *R*), *sub*(*S*,*A*) (submission *S* authored by *A*), and *pub*(*P*,*A*) (publication *P* authored by *A*). Assume a policy establishing that no one can review a paper by himself or herself or by a (possibly former) co-author. This is expressed by the following set Γ of denial constraints.

$\Gamma = \{\leftarrow rev(S,R)\ \&\ sub(S,R),\ \leftarrow rev(S,R)\ \&\ sub(S,A)\ \&\ pub(P,R)\ \&\ pub(P,A)\}$

Let U be the insertion of the facts *sub*(*c*,*a*) and *rev*(*c*,*b*) into the database, where *a*, *b*, and *c* are constants. A simplification of Γ for U (with comments behind simplified instances) is shown in Box 1.

The elements of this test are arguably easier to evaluate than Γ as they greatly reduce the space of tuples to be considered by virtue of the instantiation of variables with constants.

MAIN THRUST: INCONSISTENCY-TOLERANT INTEGRITY CHECKING

In order to define inconsistency tolerance for simplifications, we need to introduce the notion of a case of a constraint. It formalizes the thought that integrity typically is violated not as a whole, but only by cases. That way, it is possible to deal with consistent (satisfied) cases of constraints in isolation from inconsistent (violated) cases. To clarify this, we employ the notion of substitution, that is, a mapping from variables to terms. A substitution σ may also be written as $\{X_1, ..., X_n\}/\{t_1, ..., t_n\}$ to indicate that each variable X_i is orderly mapped to term t_n; the notation *range*(σ) refers to the set of variables X_i, *image*(σ) to the set of variables among the terms t_i. Whenever E is a term or formula and σ is a substitution, $E\sigma$ is the term or formula arising by applying the mapping σ to E and is called an instance of E.

A variable X occurring in a constraint W is called a global variable in W if it is universally quantified but not dominated by any existential quantifier (i.e., $\exists$ does not occur left of the $\forall$ quantifier of X) in the prenex normal form of W. Further, let *glob*(W) denote the set of global variables in W.

In fact, global variables serve very well to modularize a constraint W into cases because for a global variable X in W, the formula $\forall X\, W = W$ can be thought of as a (possibly infinite) conjunction of cases $W\{X/a\}$, $W\{X/b\}$, $W\{X/c\}$, and so on, that is, of instances of W where X is substituted by terms a, b, c, and so forth. Each such conjunct could as well be postulated and evaluated independently. It is easy to see that such a modularization is not possible for nonglobal variables.

Definition (Case)

Let W be an integrity constraint. Then, $W\sigma$ is called a *case of* W if σ is a substitution such that *range*(σ) is a subset of *glob*(W), and *image*(σ) and *glob*(W) are disjoint.

Clearly, each variable in a constraint W represented as a denial is a global variable of W. Note that cases of constraints need not be ground, and that each constraint W as well as each variant of W is a case of W by means of the identity substitution or a renaming of variables.

Now, for a given update U, the basic technical idea of simplification, which goes back to Nicolas (1982) and has been adopted by many integrity checking methods, is to focus on those cases of constraints that are affected by U and ignore all other cases. Those other cases remain unchecked because their integrity status is assured not to change from satisfied to violated by executing the update. The main conceptual difference between traditional inconsistency-intolerant and inconsistency-tolerant integrity checking essentially is that the latter does not require that the integrity status of constraints before the update be satisfied. Everything else may remain the same, that is, if the simplification obtained by focusing on affected cases evaluates to satisfied, then an inconsistency-tolerant method assures that the update will not alter the integrity status of any unconsidered case from satisfied to violated. In fact, for some cases, it may even change from violated to satisfied, as we shall argue later on, but it will indeed never worsen.

Based on the notion of cases, we are now going to define inconsistency tolerance. For convenience, we first define it for individual pre- and posttests, and then for methods that use such tests.

Definition (Inconsistency Tolerance of Pre- and Posttests)

Let D be a database, Γ an integrity theory, and U an update. A pretest (resp., posttest) Δ of Γ for D and U is inconsistency tolerant if, for each constraint γ in Γ and each case φ of γ such that $D \models \varphi$, the following holds:

$D^U \models \varphi$ if $D \models \Delta$ (resp., $D^U \models \varphi$ if $D^U \models \Delta$).

In order to characterize any given simplification method M with respect to inconsistency tolerance, let, for a database D, an integrity theory Γ, and an update U, $simp_M(D,\Gamma,U)$ denote the simplification returned by M on input D, Γ, and U. Recall again that it is left open whether M actually takes any of D or even U into account.

Definition (Inconsistency Tolerance of Simplification Methods)

A simplification method M is inconsistency tolerant if, for each database D, each integrity theory Γ, and each update U, $simp_M(D,\Gamma,U)$ is an inconsistency-tolerant (pre- or post-) test of Γ for D and U.

Example 2

To continue Example 1, suppose that the facts *rev*(d,e) and *sub*(d,e) are in D. Clearly, $D \not\models \Gamma$. However, U is not going to introduce new violations of any constraint in Γ as long as the test Δ obtained as above succeeds. Informally, the

method that returned Δ tolerates inconsistency in D and can be used to guarantee that all the cases of Γ that were satisfied in D will still be satisfied in D^U.

We first note that inconsistency tolerance is not gratuitous in general; not all simplification methods are inconsistency-tolerant.

Example 3

Consider $D = \{p(a)\}$, $\Gamma = \{\leftarrow p(X)\}$, and let U be the insertion of $p(b)$. Now, let M be an integrity checking method that produces $\Delta = \exists X\ (p(X)\ \&\ X \neq b)$ as a posttest of Γ for U. Indeed, Δ is a posttest of Γ for U since it correctly evaluates to false in D^U whenever Γ holds in D since the updated state is anyhow inconsistent. Similarly, we can argue that $\Omega = \exists X\ p(X)$ is a pretest of Γ for U, which we assume to be produced by some method M'. However, if we consider the case $\varphi = \leftarrow p(b)$, we have $D \models \varphi$ and $D^U \not\models \varphi$, although $D^U \models \Delta$ and $D \models \Omega$, so none of Δ, Ω, M, M' are inconsistency tolerant.

As for concrete integrity checking methods, we have verified in Decker and Martinenghi (2006, 2006) that the basic approaches in Nicolas (1982), Decker (1987), Lloyd et al. (1987), and Sadri and Kowalski (1988) are indeed inconsistency tolerant.

As opposed to that, the method in Gupta et al. (1994) fails to be inconsistency tolerant as shown in Decker and Martinenghi (2006). In that paper, it has also been observed that the method of Christiansen and Martinenghi (2006) is guaranteed to be inconsistency tolerant if applied to each constraint at a time, but not to nonunitary sets of constraints. In general, the lack of inconsistency tolerance of a method may be due to its capacity to deduce inconsistency caused by possible implications and dependencies between constraints, as in Gupta et al. (1994) and Christiansen and Martinenghi (2006). This tie of inconsistency tolerance to a treatment of constraints as mutually isolated requirements of integrity is not surprising since the notion of inconsistency tolerance as defined above is based on individual cases of individual constraints.

Repeating all technical details of the proofs of inconsistency tolerance of methods as cited above would be beyond the space limitations of this exposition. However, all cited proofs recur on a lemma, which provides a sufficient condition that entails inconsistency tolerance, as stated below.

Lemma (a sufficient condition for inconsistency tolerance): A simplification method $\boldsymbol{M}$ is inconsistency tolerant if, for each database D, each integrity theory Γ, each case φ' of any constraint φ in Γ, and each update U, the evaluation of $simp_{\boldsymbol{M}}(D,\{\varphi'\},U)$ yields satisfied if the evaluation of $simp_{\boldsymbol{M}}(D,\{\varphi\},U)$ yields satisfied.

The positive results about inconsistency tolerance as cited above provide us with tools that can have a major practical impact, namely that the measure of inconsistency, in a sense that can be defined, for example, as in Grant and Hunter (2006), cannot increase as long as an inconsistency-tolerant approach is used. Or, more simply put, the number of violated cases of integrity constraints will not increase as long as an inconsistency tolerant method is used for checking integrity upon each update. It may even decrease, as argued subsequently.

For example, in a relational database, the percentage of the data that participate in inconsistencies will necessarily decrease in the new state if the update consists only of insertions, as is the case for Example 2 (the simplification of which would be produced by any of the cited inconsistency-tolerant methods); in general, preventing the introduction of new inconsistency will improve integrity. Take, for example, a federation of databases: Initially, a fair amount of inconsistency can be expected. For instance, a business that has taken over a former competitor possibly cannot wait to continue to operate with a new database resulting from federating the overtaken one with its own, despite inconsistencies. However, with an inconsistency tolerant integrity checking method, the amount of inconsistency will decrease over time.

FUTURE TRENDS

So far, support for data integrity in commercial DBMSs has been scant, mainly because it is considered not sufficiently productive and too expensive. In fact, the 100% consistency as required by all traditional approaches to integrity checking is rightfully considered an illusionary ideal in practice, and any implementation that works around practical complications that may arise with regard to integrity can indeed become prohibitively costly. The tremendous potential of inconsistency tolerance, for which we are going to argue again in the conclusion, can finally be expected to bear fruit in practical implementations.

In this article, we have defined inconsistency-tolerant integrity checking in declarative terms. This means that this property is independent of the implementation of the respective method to which it applies. However, thanks to the paraconsistent behavior of resolution procedures, logic-programming-based implementations of integrity checking approaches such as in Sadri and Kowalski (1988) can also be qualified as inconsistency tolerant in a procedural sense. This issue has been addressed in more detail in Decker (1998). We reckon that an exploitation of possible synergies resulting from combining declarative inconsistency tolerance with procedural counterparts holds lots of promise, both in theoretical and in practical terms.

Integrity checking often has been rephrased or embedded in terms of other problems, namely partial evaluation (Leuschel & de Schreye, 1998), view maintenance (Gupta & Mumick, 1999), knowledge assimilation (Decker, 1998), abduction (Kakas, Kowalski, & Toni, 1992; Denecker & Kakas, 2002), query containment (Christiansen & Martinenghi, 2006) and, of course, query answering. For these reasons, we believe that relevant future work should adapt and extend the notion of inconsistency tolerance to these contexts.

Since the seminal contribution of Arenas, Bertossi, and Chomicki (1999), many authors have studied the problem of providing consistent answers to queries posed to an inconsistent database. These techniques certainly add to inconsistency tolerance in databases, but cannot be directly used to detect inconsistencies for integrity checking purposes (i.e., by posing integrity constraints as queries). Yet, distance measures used in these contexts may be regarded as a starting point for further investigations about appropriate measures of inconsistency to be applied for tolerant integrity checking. On the other hand, the magnitude of the set of violated cases of constraints may also be considered as a viable measure of inconsistency. Apart from combining declarative with procedural inconsistency tolerance, as hinted to above, an investigation of looking for a convergence of the concepts of consistent query answering and inconsistency-tolerant integrity checking (which essentially boils down to the evaluation of simplified integrity constraints as queries) seems very promising.

Currently pursued and future work is going to deal with applications of the property here called inconsistency tolerance to applications that, instead of integrity, need to express other properties of data to be monitored across state changes of the database. To mention just a few, examples of such properties are replication consistency, and the uncertainty and trustworthiness of data. For instance, as shown in (Decker & Martinenghi, 2006), certain kinds of uncertainty can be modeled by logic expressions that have the same syntax as integrity constraints. Indeed, they can very conveniently be monitored by checks that, for efficiency, need simplifications similar or equal to those used for constraints. For such checks, inconsistency tolerance (in this case, rather uncertainty tolerance) is indispensable because 100% certainty of data simply cannot be expected for most (if not all) applications in that area. Both for data integrity and other applications, also other definitions of inconsistency tolerance, not necessarily based on the notion of case, should be looked for and balanced against the definition in here.

CONCLUSION

All correctness results in the literature about efficient ways to check integrity fundamentally rely on the requirement that integrity must be satisfied

in the old state, in other words, before the update is executed. To relax or even just question this requirement has been considered sacrilegious, essentially for two reasons. First, to abandon it was deemed to destroy the local incrementality of integrity checks upon updates. Second, even the least bit of compromised consistency used to be apprehended as to sacrifice consistency altogether, fearing the worst of the principle of explosion (also known as *ex falso quodlibet*) of classical logic. By the results reported on in this article, we could dispel both hunches: The incrementality and locality of simplified integrity checking remain fully in place, and the menace of a destructive explosion of meaningfulness in the presence of inconsistency has been dismantled. In short, to the best of our knowledge, the inconsistency tolerance of simplifications, as investigated in this article, has never been studied nor even defined beforehand. Thus, the vast potential of applying theoretical results about simplified integrity checking in practice, where inconsistency is the rule rather than the exception, has remained untapped up to now.

The conflict between the so far unquestioned theoretical requirement of consistency and the failure of databases in practice to comply with it has motivated us to investigate a more tolerant attitude with respect to inconsistency. The notion of tolerance proposed above greatly extends the applicability of existing integrity checking methods that have this property and yet require no additional effort. Due to space limitations, we have not given a detailed account of concrete inconsistency-tolerant methods. However, we have cited that many, though not all of them, are inconsistency tolerant and have been shown to be so due to satisfying a simple but nontrivial condition, for which we have given proof. We have observed that a continuous application of any correct inconsistency-tolerant integrity checking method will over time improve the quality of stored data in terms of their compliance with the given ICs without any additional cost. In other words, any negative effect related to data inconsistency can be localized and thus tamed, controlled, and possibly reduced. Although a continuous application of inconsistency-tolerant integrity checking over time is likely to contribute to such a reduction, the additional use of a repair method is orthogonal and not prevented. For that purpose, an extension of the notion of inconsistency tolerance to integrity-preserving view update methods (which implement a particular kind of repair actions) is called for.

REFERENCES

Abiteboul, S., Hull, R., & Vianu, V. (1995). *Foundations of databases.* Addison-Wesley.

Arenas, M., Bertossi, L., & Chomicki, J. (1999). Consistent query answers in inconsistent databases. *PODS '99* (pp. 68-79).

Christiansen, H., & Martinenghi, D. (2006). On simplification of database integrity constraints. *Fundamenta Informaticae, 71*(4), 371-417.

Decker, H. (1987). Integrity enforcement on deductive databases. In L. Kerschberg (Ed.), *Expert database system* (pp. 381-395). Benjamin Cummings.

Decker, H. (1998). Some notes on knowledge assimilation in deductive databases. In Freitag et al. (Eds.), *Lecture notes in computer science: Vol. 1472. Transactions and change in logic databases* (pp. 249-286). Springer.

Decker, H. (2003). Historical and computational aspects of paraconsistency in view of the logic foundation of databases. In Thalheim et al. (Eds.), *Lecture notes in computer science: Vol. 2582. Semantics in databases* (pp. 63-81). Springer.

Decker, H., & Martinenghi, D. (2006). Integrity checking for uncertain data. *Proceedings of 2nd Twente Data Management Workshop on Uncertainty in Databases* (pp. 41-48).

Decker, H., & Martinenghi, D. (2006). A relaxed approach to integrity and inconsistency in databases. *Proceedings of 13th LPAR* (LCNS 4246, pp. 287-301).

Denecker, M., & Kakas, A. C. (2002). Abduction in logic programming. *Computational Logic: Logic Programming and Beyond* (pp. 402-436).

Grant, J., & Hunter, A. (2006). Measuring inconsistency in knowledge bases. *Journal of Intelligent Information Systems, 27*, 159-184.

Gupta, A., & Mumick, I. S. (Eds.) (1999). *Materialized views: Techniques, implementations, and applications.* MIT Press.

Kakas, A., Kowalski, R., & Toni, F. (1992). Abductive logic programming. *Journal of Log. Comput., 2*(6), 719-770.

Leuschel, M., & de Schreye, D. (1998). Creating specialised integrity checks through partial evaluation of meta-interpreters. *JLP, 36*(2), 149-193.

Lloyd, J. W., Sonenberg, L., & Topor, R. W. (1987). Integrity constraint checking in stratified databases. *JLP, 4*(4), 331-343.

Martinenghi, D., Christiansen, H., & Decker, H. (2006). *Integrity checking and maintenance in relational and deductive databases—and beyond.* In Z. Ma (Ed.) Intelligent databases: Technologies and Applications (pp. 238-285). Hershey PA: IGI Global.

Nicolas, J.-M. (1982). Logic for improving integrity checking in relational data bases. *Acta Informatica, 18*, 227-253.

Qian, X. (1988). An effective method for integrity constraint simplification. *ICDE '88* (pp. 338-345).

Sadri, F., & Kowalski, R. (1988). A theorem-proving approach to database integrity. In J. Minker (Ed.), *Foundations of deductive databases and logic programming* (pp. 313-362). Morgan Kaufmann.

KEY TERMS

Case: A case of a constraint *W* is another constraint obtained from *W* by substituting some (possibly all) of its global variables with constants or other variables not in *W*.

Inconsistency: Property of a database state that does not satisfy its associated integrity theory. Inconsistency is synonymous to integrity violation and is the contrary of integrity satisfaction.

Inconsistency Tolerance: Property of simplification methods that makes them usable also in the presence of inconsistency. Whenever an inconsistency-tolerant method returns that integrity is satisfied, the preservation of all consistent cases of the integrity theory in the updated state is guaranteed even if there were inconsistent cases before the update.

Integrity: In databases, integrity stands for semantic consistency, that is, the correctness of stored data with regard to their intended meaning, as expressed by integrity constraints. Integrity should not be confused with a namesake issue often associated with data security.

Integrity Constraint: A logical sentence expressing a statement about the semantics, that is, the intended meaning of the extensions of tables and views in the database. It is usually expressed either as closed formulas in prenex normal form or as denials. Both can be evaluated as queries for determining integrity satisfaction or violation.

Integrity Satisfaction: A given database state satisfies integrity if each integrity constraint in the database schema is satisfied. A constraint in prenex normal form is satisfied if, when posed as a query, it returns the answer *yes*. A constraint in denial form is satisfied if, when posed as a query, it returns the answer *no*.

Integrity Violation: A given database state violates integrity if at least one of the integrity constraints in the database schema is violated. A constraint in prenex normal form is violated if, when posed as a query, it returns the answer *no*. A constraint in denial form is violated if, when posed as a query, it returns the answer *yes*.

Posttest: A posttest of an integrity theory Γ (for an update and, possibly, a database) is an integrity theory, easier to evaluate than Γ, that evaluates in the new state exactly as Γ.

Pretest: A pretest of an integrity theory Γ (for a given update and, possibly, a database) is an integrity theory, easier to evaluate than Γ, that evaluates in the old state exactly as Γ does in the new state.

Simplification Method: A procedure taking as input an integrity theory, an update, and possibly a database state, and returning as output a pretest or posttest thereof.

Update: An update is a mapping from the space of databases into itself. Its input is the old state, while its output is the new state of the database. Many simplification methods use the difference between old and new state for making integrity checking more efficient.

Chapter XXXIX
Merging, Repairing, and Querying Inconsistent Databases

Luciano Caroprese
University of Calabria, Italy

Ester Zumpano
University of Calabria, Italy

INTRODUCTION

Data integration aims to provide a uniform integrated access to multiple heterogeneous information sources designed independently and having strictly related contents. However, the integrated view, constructed by integrating the information provided by the different data sources by means of a specified integration strategy could potentially contain inconsistent data; that is, it can violate some of the constraints defined on the data. In the presence of an inconsistent integrated database, in other words, a database that does not satisfy some integrity constraints, two possible solutions have been investigated in the literature (Agarwal, Keller, Wiederhold, & Saraswat, 1995; Bry, 1997; Calì, Calvanese, De Giacomo, & Lenzerini, 2002; Dung, 1996; Grant & Subrahmanian, 1995; S. Greco & Zumpano, 2000; Lin & Mendelzon, 1999): repairing the database or computing consistent answers over the inconsistent database. Intuitively, a repair of the database consists of deleting or inserting a minimal number of tuples so that the resulting database is consistent, whereas the computation of the consistent answer consists of selecting the set of certain tuples (i.e., those belonging to all repaired databases) and the set of uncertain tuples (i.e., those belonging to a proper subset of repaired databases).

Example 1. Consider the database consisting of the relation *Employee(Name, Age, Salary)* where the attribute *Name* is a key for the relation, and suppose we have the integrated database DB = {Employee(Mary, 28, 20), Employee(Mary, 31, 30), Employee(Peter, 47, 50)}. DB is inconsistent and there are two possible repaired databases each obtained by deleting one of the two tuples whose value of the attribute *Name* is Mary. The answer to the query asking for the age of Peter is constituted by the set of certain tuples {<47>}, whereas the answer to the query asking for the age of Mary produces the set of uncertain values {<28>, <31>}.

This work proposes a framework for merging, repairing, and querying inconsistent databases. To this aim the problem of the satisfaction of integrity constraints in the presence of null values is investigated and a new semantics for constraints satisfaction, inspired by the approach presented in Bravo and Bertossi (2006), is proposed. The present work focuses on the inconsistencies of a database instance with respect to particular types of integrity constraints implemented and maintained in a commercial DBMS (database management system) such as primary keys, general functional dependencies, and foreign-key constraints.

The framework for merging, repairing, and querying inconsistent databases with functional dependencies restricted to primary-key constraints and foreign-key constraints has been implemented in a system prototype, called RAINBOW, developed at the University of Calabria.

PRELIMINARIES

Before presenting the problems related to the merging, repairing, and querying of inconsistent databases, let us introduce some basic definitions and notations. For additional material see Abiteboul, Hull, and Vianu (1994) and Ullman (1989).

A relational schema DS is a pair DS = $<R_S, IC>$ where Rs is a set of relational symbols and IC is a set of integrity constraints; that is, it is an assertion that has to be satisfied by a generic database instance. Given a database schema DS = $<R_S, IC>$ and a database instance DB over R_S, we say that DB is consistent if DB |= IC, in other words, if all integrity constraints in IC are satisfied by DB; otherwise, it is inconsistent.

A relational query (or simply a query) over R_S is a function from the database to a relation. In the following we assume queries over $<R_S, IC>$ are conjunctive queries. We will denote with Dom the database domain, that is, the set of values an attribute can assume, consisting of a possibly infinite set of constants, and assume $\perp \in$ Dom, where $\perp$ denotes the null value.

DATABASE MERGING

Once the logical conflicts owing to the schema heterogeneity have been resolved, conflicts may arise during the integration process among data provided by different sources. In particular, the same real-world object may correspond to many tuples that may have the same value for the key attributes but different values for some nonkey attribute.

The database integration problem consists of the merging of *n* databases $DB_1 = \{R_{1,1}, .., R_{1,n1}\}$, …, $DB_k = \{R_{k,1}, .., R_{k,nk}\}$. In the following we assume that relations corresponding to the same concept and furnished by different sources are homogenized with respect to a common ontology so that attributes denoting the same property have the same name (Yan & Ozsu, 1999). We say that two homogenized relations R and S, associated with the same concept, are overlapping if they have the same value for the key attributes. Given a set of overlapping relations, an important feature of the integration process is related to the way conflicting tuples are combined. Before performing the database integration, the relations to be merged must be first reconciled so that they have the same schema.

Definition 1. Given a set of overlapping relations $\{S_1, \ldots, S_n\}$, a reconciled relation R is $attr(R) = \bigcup_{1}^{n} attr(S_i) \cup \{Src\}$, and contains all tuples $t \in S_i$, $1 \leq i \leq n$. All attributes belonging to $attr(R) - attr(S_i)$ are fixed to $\perp$; R[Src] = i, where i is the unique index of the source database.

Example 2. Consider the following two overlapping relations S_1 and S_2.

K	Title	Author
1	Moon	Greg
2	Money	Jones
3	Sky	Jones

S_1

K	Title	Author	Year
3	Flower	Smith	1965
4	Sea	Taylor	1971
7	Sun	Steven	1980

S_2

The reconciled relation R is the following.

K	Title	Author	Year
1	Moon	Greg	⊥
2	Money	Jones	⊥
3	Sky	Jones	⊥
3	Flower	Smith	1965
4	Sea	Taylor	1971
7	Sun	Steven	1980

Given a database DB consisting of a set of reconciled relations $\{R_1, \ldots, R_n\}$, the integrated database DB^I consists of a set of *n* integrated relations $\{R^I_1, \ldots, R^I_n\}$, where each R^I ($j \in [1..n]$) is obtained by applying an integration operator, denoted as ♦, to the reconciled relation R_j, that is, $R^I_j = \blacklozenge(R_j)$.

In order to perform the database integration task, several integration operators have been proposed in the literature. We recall here the match join operator (Yan & Ozsu, 1999), the merging-by-majority operator (Lin & Mendelzon, 1999), the merge operator, and the prioritized merge operator (G. Greco, Greco, & Zumpano, 2001).

Before presenting, in an informal way, these operators, we introduce some preliminary definitions.

Definition 2. Given two relations R and S such that attr(R) = attr(S), and two tuples $t_1 \in R$ and $t_2 \in S$, we say that t_1 is less informative than t_2 ($t_1 << t_2$) if for each attribute A in attr(R), $t_1[A] = t_2[A]$ or $t_1[A] = \perp$. Moreover, we say that $R << S$ if $\forall t_1 \in R, \exists\, t_2 \in S$ s.t. $t_1 << t_2$.

In the following, given a set of overlapping relations $\{S_1, \ldots, S_n\}$ and the corresponding reconciled relation R, we will denote as $R|_i$ the set of tuples in R obtained by extending tuples in S_i with ⊥ value for each attribute A not belonging to attr(S_i).

In the rest of this section, we will informally describe some integration operators proposed in the literature. We suppose the database source DB_i is preferred by the user over each database DB_j, with $j > i$.

The integration strategy performed by different integration operators, as proposed in the literature, will be described by evaluating the integrated relation, R^I, obtained by applying them to the reconciled relation *R* in Example 2. Moreover, we suppose the functional dependency *Title* → *Author* is defined over *R*; that is, the attribute *Author* is functionally dependent on the attribute *Title*. We recall that inconsistencies taken into account by these integration operators only derive from primary-keys violations.

The match join operator, proposed in Yan and Ozsu (1999), manufactures tuples in the integrated relation by performing the outer join of the *ValSet* of each attribute, where the *ValSet* of an attribute A is the projections of the reconciled relation on {K,A}.

Example 3. The integrated relation R^I is the following.

K	Title	Author	Year
1	Moon	Greg	⊥
2	Money	Jones	⊥
3	Sky	Jones	1965
3	Sky	Smith	1965
3	Flower	Jones	1965
3	Flower	Smith	1965
4	Sea	Taylor	1971
7	Sun	Steven	1980

As shown by this example, since the match join operator mixes values coming from different tuples with the same key in all possible ways, it may generate a relation that is not anymore consistent with the set of integrity constraints.

The merging-by-majority operator, proposed in Lin and Mendelzon (1999), tries to remove conflicts taking into account the majority view of the databases; in other words, it maintains the (not-null) value that is present in the majority of the databases. Thus, the operator constructs an integrated relation containing generalized tuples, that is, tuples where each attribute value is a simple value if the information respects the

majority criteria, or a set if the operator does not resolve the conflict.

Example 4. The merging-by-majority operator is not able to solve the conflict present in *R* between the two tuples t_1 and t_2 having $key(t_1) = key(t_2) = 3$. Thus, R^I will contain also the generalized tuple *{*3*, {Sky, Flowers}, {Jones, Smith},* 1965*}*.

The prioritized merge operator, introduced in G. Greco et al. (2001), is an operator that, if conflicting tuples are detected, gives preference to those belonging to the source on which the user expressed preference: A tuple coming from a preferred relation is always maintained, and if it has some null values, it is completed with not-null values provided by less preferred relations.

Example 5. The integrated relation R^I is the following.

K	Title	Author	Year
1	Moon	Greg	⊥
2	Money	Jones	⊥
3	Sky	Jones	⊥
4	Sea	Taylor	1971
7	Sun	Steven	1980

It can be easily shown that the complexity of constructing the merged database by means of an integration strategy is polynomial time.

REPAIRING INCONSISTENT DATABASES IN THE PRESENCE OF NULL VALUES

In this section, we provide details on the satisfaction of functional dependencies and foreign-key constraints in the presence of null values and show how to handle a violation for these kinds of constraints. The motivation for considering only these limited forms of constraints relies on the fact they can be defined and maintained in commercial DBMSs and are the types of constraints we manage in the system prototype for integrating, repairing, and querying inconsistent databases. The interested reader can refer to Caroprese and Zumpano (2006) for the proposal of a general constraints satisfaction in the presence of null values.

Definition 3. (Functional Dependency Satisfaction) Given a functional dependency fd of the form

$$\forall\ (X, Y,Z)[p(X,Y\ ),\ p(X\ ,Z) \supset Y = Z],$$

a database DB satisfies fd if for each $p(x, y) \in$ DB and, $p(x, z) \in$ DB then $y = z$ or $x = \perp$. If a violation of fd occurs, that is, there exists $p(x, y) \in$ DB and $p(x, z) \in$ DB with $x \neq \perp$ and $y \neq z$, then the constraint can be satisfied by deleting either p(x, y) or p(x, z).

Definition 4. (Foreign Key Satisfaction) Given a foreign key constraint fk of the form

$$\forall\ (X, Y)[p(X,Y\ ) \supset \exists\ Zq(X, Z)],$$

where X, Y, Z are lists of variables and Y, Z may be empty lists, then a database DB satisfies fk if for each $p(x, y) \in$ DB there exists $q(x, z) \in$ DB or $x = \perp$. If a violation of fk occurs, that is, there exists $p(x, y) \in$ DB DB with $x \neq \perp$ and there does not exist $q(x, z) \in$ DB, then the constraint can be satisfied by either deleting the tuple p(x, y) or inserting a tuple $q(x,\perp)$.

Before formally introducing the notion of repair in the presence of null values, some preliminaries are provided. An update atom is in the form +a(X) or -a(X). A ground atom +a(t) states that a(t) will be inserted into the database, whereas a ground atom -a(t) states that a(t) will be deleted from the database. Given a set U of ground update atoms, we define the sets $U^+ = \{a(t) \mid + a(t) \in U\}$, $U^- = \{a(t) \mid - a(t) \in U\}$. We say that U is consistent if it does not contain two update atoms +a(t) and -a(t) (i.e., if $U^+ \cap U^- = \varnothing$). Given a database DB and a consistent set of update atoms U, we denote as U(DB) the updated database $DB \cup U^+ \cup U^-$. In the following we will use IC to denote a set

of constraints including functional dependencies and foreign-key constraints.

Definition 5. Given a database DB and a set of integrity constraints IC, a repair for {DB,IC} is a consistent set R of update atoms such that:

- *R(DB)* |= *IC*,
- there is no consistent set U of update atoms such that U ⊂ R and U(DB) |= IC, and–
- there is no update atom +a(x) ∈ R s.t. There exists +a(x') with +a(x') << +a(x) when R' = R ∪ {+a(x')} – {+a(x)}, R'(DB)|= IC.

The third condition in the previous definition ensures that for each database *DB* and set of integrity constraints *IC* there is a finite number of repairs. Observe that if ⊥ ∈ *Dom*, the notion of repair here provided coincides with the one given in (Greco et al., 2001).

Example 6. Consider the following set of integrity constraints *IC:*

- ∀ (X, Y,Z)[p(X,Y), p(X ,Z) ⊃ Y = Z]
- ∀ (X, Y,Z)[q(X,Y), q(X ,Z) ⊃ Y = Z]
- ∀ (X, Y)[p(X,Y) ⊃ ∃ Zq(X, Z)]

The database DB = {p(a, ⊥), p(a, b)}. DB is inconsistent with respect to IC and the repairs for {DB, IC} are R_1 = {-p(a, b)} and R_2 = {-p(a, ⊥), +q(b, ⊥)}. Observe that each set U = {-p(a, ⊥), +q(b, X)} of update atoms with X, a constant different from ⊥ is not a repair as it does not satisfy the third condition in Definition 5.

QUERYING INCONSISTENT DATABASES IN THE PRESENCE OF NULL VALUES

In the rest of the section we investigate the problem of querying an inconsistent database by considering the computation of consistent answers. The set of tuples present in the database, that is, those implied by the constraints or originally present may be either true, false, or undefined.

Definition 6. Given a database schema DS = <R_S, IC> and a database DB over R_S, an atom A is true (resp. false) with respect to <DB,IC> if A belongs to all repaired databases (resp. there is no repaired database containing A). The set of atoms that are neither true nor false are undefined.

Thus, true atoms appear in all repaired databases (Arenas, Bertossi, & Chomicki, 1999), whereas undefined atoms appear in a proper subset of repaired databases.

Definition 7. Given a database schema DS = <R_S,IC>, a database DB over R_S, and a query Q, the consistent answer of the query Q on the database DB, denoted as Q(DB,IC), consists of three sets, denoted $Q(DB,IC)^+$, $Q(DB,IC)^-$, and $Q(DB,IC)^u$, containing, respectively, the sets of tuples that are true (i.e., belonging to Q(DB') for all repaired databases DB'), false (i.e., not belonging to Q(DB') for all repaired databases DB'), and undefined (i.e., set of tuples that are neither true nor false).

Example 7. Consider the set of constraints in Example 6 and the database DB = {p(a, ⊥), p(a,b), p(c,d), q(d,e)}. There are two repairs R_1 and R_2 that coincide with those reported in Example 6 and produce respectively the repaired databases DB_1 = {p(a, ⊥), p(c,d), q(d, e)} and DB_2 = {p(a, b), p(c,d), q(d, e), q(b, ⊥)}. Given the query Q_1, p(X, Y), the set of true tuples is {<d,e>}, whereas the set of undefined tuples is {<a, ⊥>, <a, b>}. For the query Q_2, s(X;Z) ← p(X, Y), q(Y,Z), the set of true tuples is {<d, e>}, whereas the set of undefined tuples is {<a, ⊥>}.

CONCLUSION AND FUTURE TRENDS

In this work, a framework for merging, repairing, and querying inconsistent databases with functional dependencies restricted to primary-key constraints and foreign-key constraints has been presented. The approach consists of two main

steps: the merging of the source databases by means of a specific integration and the repairing of the possible inconsistent integrated database, which may be inconsistent with respect to functional dependencies and foreign-keys constraints. The architecture of a system prototype, called RAINBOW, developed at the University of Calabria, implementing the proposed approach, has been also presented. We are currently extending this work by specifying more flexible semantics that select as true information that supported by all repaired databases, that is, the set of atoms that maximizes the information shared by the repaired databases.

REFERENCES

Abiteboul, S., Hull, R., & Vianu, V. (1994). *Foundations of databases.* Addison-Wesley.

Agarwal, S., Keller, A. M., Wiederhold, G., & Saraswat, K. (1995). *Flexible relation: An approach for integrating data from multiple, possibly inconsistent databases.* ICDE.

Arenas, M., Bertossi, L., & Chomicki, J. (1999). Consistent query answers in inconsistent databases. *Proceedings of PODS 1999* (pp. 68-79).

Baral, C., Kraus, S., & Minker, J. (1991). Combining multiple knowledge bases. *IEEE-TKDE, 3*(2), 208-220.

Bravo, L., & Bertossi, L. (2006). Semantically correct query answers in the presence of null values. *Proceedings of EDBT WS on Inconsistency and Incompleteness in Databases (IIDB)* (pp. 336-357).

Bry, F. (1997). Query answering in information system with functional des. *IICIS* (pp. 113-130).

Calì, A., Calvanese, D., De Giacomo, G., & Lenzerini, M. (2002). Data integration under integrity constraints. *CAiSE* (pp. 262-279).

Caroprese & Zumpano. (2006). A framework for merging, repairing and querying inconsistent databases. *ADBIS* (pp. 383-398).

Dung, P. M. (1996). Integrating data from possibly inconsistent databases. *COOPIS* (pp. 58-65).

Grant, J., & Subrahmanian, V. S. (1995). Reasoning in inconsistent knowledge bases. *IEEE-TKDE, 7*(1), 177-189.

Greco, G., Greco, S., & Zumpano, E. (2001). A logic programming approach to the integration, repairing and querying of inconsistent databases. *ICLP* (pp. 348-364).

Greco, S., Pontieri, L., & Zumpano, E. (2001). Integrating and managing conflicting data. *Ershov Memorial Conference* (pp. 349-362).

Greco, S., & Zumpano, E. (2000). Querying inconsistent database. *LPAR* (pp. 308-325).

Lin, J. (1996a). Integration of weighted knowledge bases. *Artificial Intelligence, 83*(2), 363-378.

Lin, J. (1996b). A semantics for reasoning consistently in the presence of inconsistency. *AI, 86*(1), 75-95.

Lin, J., & Mendelzon, A. O. (1999). Knowledge base merging by majority. In R. Pareschi & B. Fronhoefer (Eds.), *Dynamic worlds.* Kluwer.

Subrahmanian, V. S. (1994). Amalgamating knowledge bases. *ACM-TODS, 19*(2), 291-331.

Ullman, J. D. (1989). *Principles of database and knowledge-base systems* (Vol. 1). Computer Science Press.

Yan, L. L., & Ozsu, M. T. (1999). Conflict tolerant queries in Aurora Coopis. (pp. 279-290).

KEY TERMS

Consistent Answer: A set of tuples, derived from the database, satisfying all integrity constraints.

Consistent Database: A database satisfying a set of integrity constraints.

Database Repair: Minimal set of insert and delete operations that makes the database consistent.

Data Integration: The activity of combining and matching information in different sources and resolving a variety of conflicts.

Foreign-Key Constraint: A foreign-key constraint (also called referential integrity constraint) on a column ensures that the value in that column is found in the primary key of another table.

Functional Dependency: A functional dependency is a constraint between two sets of attributes in a relation from a database. Given a relation R, a set of attributes X in R is said to functionally determine another attribute Y, also in R (written $X \rightarrow Y$) if and only if each X value is associated with at most one Y value.

Inconsistent Database: A database violating some integrity constraints.

Integration Operator: The operation of merging information by extracting coherent common information from several sources of data.

Chapter XL
The Challenges of Checking Integrity Constraints in Centralized, Distributed, and Parallel Databases

Hamidah Ibrahim
Universiti Putra Malaysia, Malaysia

INTRODUCTION

A vital problem that should be tackled in today's database system is guaranteeing *database consistency*. Many techniques and tools have been devised to fulfill this requirement in many interrelated research areas such as concurrency control, security control, reliability control, and integrity control (Eswaran & Chamberlin, 1975; Grefen, 1993). *Concurrency control* deals with the prevention of inconsistencies caused by concurrent access by multiple users or applications to a database. *Security control* deals with preventing users from accessing and modifying data in a database in unauthorized ways. *Reliability control* deals with the prevention of errors due to the malfunctioning of system hardware or software. *Integrity control* deals with the prevention of semantic errors made by users due to their carelessness or lack of knowledge. This chapter is concerned only with integrity control.

A database state is said to be consistent if the database satisfies a set of statements called *semantic integrity constraints* (or simply *constraints*). Integrity constraints stipulate those configurations of the data that are considered semantically correct. Any update operation (insert, delete, or modify) or transaction (sequence of updates) that

occurs must not result in a state that violates these constraints. Thus, a fundamental issue concerning integrity constraints is *constraint checking*; that is, the process of ensuring that the integrity constraints are satisfied by the database after it has been updated. Checking the consistency of a database state will generally involve the execution of *integrity tests* (query that returns the value true or false) on the database, which verify whether or not the database is satisfying its constraints.

Integrity checking is primarily implemented in one of the following ways, depending on who or what is responsible for ensuring the consistency of a database. The responsibility for constraint checking is either allocated to the users of the database or to a database management system component called *semantic integrity subsystem* (SIS). In the former approach, the users are responsible for specifying a set of integrity constraints that the update or transaction may violate, and they must include checks or integrity tests into update transactions to ensure that none of these constraints will be violated. In the alternative approach, a complete set of integrity constraints is identified once by the users and then applied to all updates or transactions of the database. Obviously, the first approach is less friendly and more error-prone because the efficiency and correctness of manually written integrity tests rely on the users' skills; and changes to constraint definitions require modification of all transactions, which include integrity control with respect to these definitions. However, integrity constraint checking is not fully supported by current database technology (Martinenghi, 2005).

The growing complexity of modern database applications plus the need to support multiple users has further increased the need for a powerful integrity subsystem to be incorporated into these systems. Therefore, a complete integrity subsystem is considered an important part of any modern DBMS. The crucial problem is the difficulty of devising an efficient algorithm for enforcing database integrity against updates. A naive approach is to perform the update and then check whether the integrity constraints are satisfied in the new database state. This method, termed *brute force checking*, is very expensive and impractical, and can lead to prohibitive processing costs because the evaluation of integrity constraints requires accessing large amounts of data that are not involved in the database update transition (Simon & Valduriez, 1987). Hence, improvements to this approach have been reported in many research papers.

Many factors need to be well thought out before an enforcement mechanism can be devised. These factors include the following:

- The type of environment considered (i.e., whether it is centralized, distributed, parallel, or even mobile. Different environments require different schemes of enforcing the integrity of the system due to different architectures, components, function of the components, and so forth.
- The type of integrity constraints considered. There are many classifications of integrity constraints ranging from simple to complex. The classifications of integrity constraints are based on some of their characteristics; for instance, scope space, data model, time scope, definiteness of the constraints (hard condition or fuzzy condition), computational complexity, or even properties of the constraints such as soundness and completeness. Most of the approaches proposed for checking constraints are limited to certain types of integrity constraints that are generally used and referred to in theory and practice, such as implicit, inherent, and explicit constraints of a data model, and state and transition constraints of a database application, which include value set (domain) constraints (e.g., the age of an employee must not be less than 20), key (uniqueness) constraints (e.g., every employee has a unique employee number), structural constraints (e.g., null value is not

allowed for a key attribute), relationship structural constraints (e.g., every employee is assigned to a department), superclass/subclass constraints (e.g., every manager is an employee), and general semantic integrity constraints (e.g., every employee earns less than his or her manager in the same department) (Date, 2003; Elmasri & Navathe, 2007; Ibrahim, 1998). Devising a general mechanism for checking all of the types of constraints is impossible since these constraints have different characteristics that require different treatments.

- The type of tests to be derived. Like integrity constraints, integrity tests can be classified into several categories, depending on the characteristics of the tests. Three types of integrity tests based on its properties are defined in McCune and Henschen (1989); namely, *sufficient tests*, *necessary tests*, and *complete tests*. An integrity test has the sufficiency property if, when the test is satisfied, this implies that the associated constraint is satisfied and thus the update operation is safe with respect to the constraint. An integrity test has the necessity property if, when the test is not satisfied, this implies that the associated constraint is violated, and thus the update operation is unsafe with respect to the constraint. An integrity test has the completeness property if the test has both the sufficiency and necessity properties.

In noncentralized databases, another two types of integrity tests are introduced; namely, *local test* and *global test*. These tests are based on region. The integrity tests that are evaluated to verify the consistency of a database within the local region are referred to as local tests. Since these tests can only span the local region, alternative tests are required (if the local tests failed); namely, those that are evaluated outside the boundary of the local region. These tests are referred to as global tests (Gupta, 1994; Ibrahim, 1998).

Another classification of tests is based on the method used; that is, either detection that allows an update operation to be executed on a database state and when an inconsistent result state is detected undo this update; or prevention that avoids undoing the updates by evaluating the tests before the database state transition caused by the update occurs. The tests are termed *posttest* and *pretest*, respectively (Ibrahim, 1998; Simon & Valduriez, 1987).

Studies show the performance of the database system can be improved if the suitable type of test is selected appropriate to the type of environment (Gupta, 1994; Ibrahim, 2002; Mazumdar, 1993) and believe that sufficient tests should be explored in distributed databases as they always span local regions (Alwan, Ibrahim & Udzir, 2007), while stating that in distributed databases, support tests can also benefit the system.

Table 1 presents the classifications of integrity tests, which are adopted from Ibrahim (1998). From the table, a *global post sufficient* test means that the evaluation of the test spans the remote regions (sites); the test is evaluated after the update is performed, and if the test is satisfied, this implies that the associated initial constraint is satisfied, and thus, the update operation is safe with respect to the constraint.

The problem of devising efficient enforcement mechanisms in a distributed environment is more crucial than in a centralized environment due to the following facts (Barbara & Garcia-Molina, 1992; Mazumdar, 1993; Qian, 1989; Simon & Valduriez, 1987):

- Integrity constraints may spread over several sites in distributed databases. A large amount of data may therefore need to be transferred around the network in order to determine the truth of such statements.
- Owing to the possibility of fragmentation of relations with the fragments stored at different locations, the integrity constraints must be transformed into constraints on the

Table 1. Classification of integrity tests

Integrity Test Based on Region	Integrity Test Based on Detection/Prevention Methods	Integrity Test Based on Its Properties
Global Test spans remote sites	Posttest evaluated after an update is performed	Sufficient Test
		Necessary Test
		Complete Test
	Pretest evaluated before an update is performed	Sufficient Test
		Necessary Test
		Complete Test
Local Test spans local sites	Posttest evaluated after an update is performed	Sufficient Test
		Necessary Test
		Complete Test
	Pretest evaluated before an update is performed	Sufficient Test
		Necessary Test
		Complete Test

fragments so they can be used straightforwardly for constructing efficient enforcement algorithms. Thus, there are usually more integrity constraints in distributed databases, which need to be maintained. In addition, replication of data imposes an additional constraint that the replicas must have equivalent values at all times.

- Frequent updates can lead to frequent executions of expensive violation testing operations.
- Allowing an update to execute with the intention of aborting it at commit time in the event of constraint violation is also inefficient since rollback and recovery must occur at all sites that participated in the update, which can be a very costly operation in distributed systems.

The brute force strategy of checking constraints is worse in the distributed context since the checking would typically require data transfer as well as computation, leading to complex algorithms to determine the most efficient approach.

In parallel databases, the database relations are either physically (static) or virtually (dynamic) partitioned into several regions (partitions). The problem of devising an efficient enforcement mechanism is more complicated if the parallel databases exist in a distributed environment compared to a centralized environment, as the mechanism has to not only cater to the previously mentioned facts but also adhere to the following facts (Ibrahim, 2006):

- The idea of parallel processing is to reduce the time taken for checking integrity constraints, which can be accomplished by executing several tests or subtests in parallel (at the same time) by different processors. Generating such tests requires complex algorithms.
- The integrity constraints must be transformed into constraints on the regions so they can be used straightforwardly.
- Partitioning the database relations into regions needs a good design strategy, as this can impact the performance of the enforce-

ment mechanism. The static partitioning strategy (McCarroll, 1995) partitions the database relations based on certain conditions, similar to the fragmentation strategy used in distributed databases. The constraint optimization techniques used in distributed databases can be applied to the set of integrity constraints. While the dynamic partitioning strategy partitions each relation into several partitions based on the number of available processors (Hanandeh, Ibrahim, Mamat & Johari, 2004). The comparisons between these strategies can be found in Hanandeh (2006).

- Selecting the best parallel processing strategy with regard to constraint checking depends on the application domain being considered and the partitioning strategy, as well as the contents of the database.

Throughout this chapter, the example *emp_dept* database is used, as given in Figure 1. The example database has been chosen since it has been used and cited in many previous research works.

ISSUES OF CHECKING INTEGRITY CONSTRAINTS

Many researchers have studied the problem of maintaining the consistency of a database, and not surprisingly, many approaches have been proposed. Whatever the environment is, the ultimate question of interest is *how to efficiently check integrity constraints.* This section deliberates on the foremost factors that influence the performance of constraint checking for the various types of environments; namely, centralized, distributed, and parallel.

For *centralized database*, researchers have suggested that efficient constraint checking can be achieved by reducing the number of integrity constraints that needs checking, reducing the amount of data that needs to be accessed, and avoiding undoing the updates. Other measurements that can be used to assess the efficiency of an enforcement mechanism include the complexity of the method embedded in the enforcement mechanism, the types of integrity constraints it can handle, and the various types of integrity tests that can be derived, as presented in the previous section. These general measurements can be used

Figure 1. The emp_dept intensional database

Schema:
emp(*eno, dno, ejob, esal*); *dept*(*dno, dname, mgrno, mgrsal*)
Integrity Constraints:
'The dno of every tuple in the *emp* relation exists in the *dept* relation'
IC-1*:* $(\forall t \forall u \forall v \forall w \exists x \exists y \exists z)(emp(t, u, v, w) \rightarrow dept(u, x, y, z))$
'Every employee must earn ≤ to the manager in the same department'
IC-2*:* $(\forall t \forall u \forall v \forall w \forall x \forall y \forall z)(emp(t, u, v, w) \Lambda\ dept(u, x, y, z) \rightarrow (w \leq z))$
Fragmentation Rules:
Case 1:
FR-1: $(\forall w \forall x \forall y \forall z)(emp_i(w, x, y, z) \rightarrow (x = Di))^+$
FR-2: $(\forall w \forall x \forall y \forall z)(dept_i(w, x, y, z) \rightarrow (w = Di))^+$
Case 2:
FR-3: $(\forall w \forall x \forall y \forall z)(empt_i(w, x, y, z) \rightarrow (x = Di))^+$
FR-4: $(\forall w \forall x \forall y \forall z)(dept_1(w, x, y, z) \rightarrow (z \geq 10000))$
FR-5: $(\forall w \forall x \forall y \forall z)(dept_2(w, x, y, z) \rightarrow (z < 10000))$
Allocation Rules: Assume that each fragment is allocated at different sites of the network.
$^+$ for $i = \{1, 2, \ldots, n\}$

not only to assess the efficiency of the enforcement mechanisms for centralized environment but also for noncentralized environment.

Reducing the number of integrity constraints that needs checking can be accomplished by exploiting the fact that the constraints are known to be satisfied prior to an update and only checking the subset of integrity constraints that may be violated by the current update or transaction. This is based on the following observation by Nicolas (1982). Given a valid database state, a new state is obtained when it is updated either by a single update operation or by a transaction. Depending on the operation leading to the new state, some integrity constraints remain necessarily satisfied in this new state, while others have to be evaluated to determine whether they are actually satisfied or not. Consequently, this revised strategy, which filters the initial set of integrity constraints to identify and select only those constraints that might be violated given an update operation, is more efficient than a basic strategy that checks all the constraints. This strategy, known as *incremental integrity checking* (Christiansen & Martinenghi, 2005; Gupta, 1994; Martinenghi, 2005; Plexousakis, 1993) or constraint filtering (Grefen, 1990, 1993), is the basis of most current approaches to integrity checking in databases. In Gupta (1994), this strategy is also referred to as a brute force strategy, because by default, it uses all the underlying relations. Example: Given the update operation, insert(*emp*(*a*, *b*, *c*, *d*)), both *IC*-1 and *IC*-2 need to be checked, whereas given the update operation, insert(*dept*(*a*, *b*, *c*, *d*)), none of the constraints *IC*-1 and *IC*-2 need to be checked. Here, the well-known update theorems can be applied (McCune & Henschen, 1989; Nicolas, 1982). The theorems state that given a constraint specified in prenex conjunctive normal form, the constraint is affected by an update only when a tuple is inserted into the extension of a literal occurring negatively in the constraint, or when a tuple is deleted from the extension of a literal occurring positively in the constraint. The previous theorems, which analyzed the syntactic structure of each constraint, are able to automatically identify the update operations that might violate a constraint. Analyzing the semantic structure of the constraints can further reduce the number of constraints to be checked. Example: If *IC*-2 is satisfied, then *IC*-1 is also satisfied. Thus, an approach that can discover the relationships in terms of the meaning (semantic) between constraints is indeed essential.

Another strategy is to simplify the constraint formulae so that *less data are accessed* in order to determine the truth of the constraint. With the assumption that the set of initial constraints, *IC*, is known to be satisfied in the state before an update, simplified forms of *IC* (e.g., *IC'*) are constructed such that *IC* is satisfied in the new state if and only if *IC'* is satisfied, and the evaluation cost of *IC'* is less than or equal to the evaluation cost of *IC*. This strategy is referred to as *constraint simplification*, and the simplified forms of these constraints are referred to as *integrity tests* (Ibrahim, Gray & Fiddian, 1996; McCarroll, 1995) or *constraint protectors.* This approach conforms to the admonition of Nicolas (1982) to concentrate on the problem of finding *good* constraints. Various simplification techniques have been proposed where integrity tests are derived from the syntactic structure of the constraints and the update operations (Bernstein & Blaustein, 1981; Blaustein, 1981; Gupta 1994; Henshen, McCune & Naqvi, 1984; Hsu & Imielinski 1985; McCarroll, 1995; McCune & Henschen, 1989; Nicolas, 1982; Qian, 1989; Simon & Valduriez, 1987). These techniques are referred to as *constraint simplification by update analysis*. An important property desired of an integrity test is that the test ought to be cheaper to execute than the initial constraint from which it is derived. Example: Checking the complete test $(\exists x \exists y \exists z)(dept(b, x, y, z))$ or the sufficient test $(\exists t \exists v \exists w)(emp(t, b, v, w))$ when insert(*emp*(*a*, *b*, *c*, *d*)) is submitted is said to be cheaper than checking its initial constraint, *IC*-1. Thus, it is important to ensure that such

integrity tests are as efficient as possible in order to reduce the performance overheads imposed by integrity enforcement. The issue addressed here is *how to derive an efficient set of tests to prove that an update operation will guarantee the semantic integrity of the database with respect to each and every constraint defined on the database.*

Furthermore, to *avoid undoing the updates*, these tests must be evaluated before the database state transition caused by the update occurs. Thus, preventive methods are favored over detective methods.

For *distributed database* environment, efficient constraint checking can be realized by not only considering the factors of centralized database but also focusing on the following components: *the amount of data that needs to be transferred across the network* and *the number of sites that are involved* in order to check a constraint. As a consequence, most of the strategies proposed for centralized systems are utilized, in particular, the incremental integrity checking strategy, which can reduce the number of constraints to be evaluated; the constraint simplification strategy, which can reduce the amount of data that needs to be accessed; and the pretest evaluation, which can avoid undoing the checking process.

In addition, new strategies appropriate to distributed environment have been proposed that aim to reduce the number of sites involved and thus reduce the amount of data transferred across the network (Barbara & Garcia-Molina, 1992; Gupta 1994; Ibrahim, 2002; Ibrahim, Gray & Fiddian, 1998; Mazumdar, 1993; Qian, 1989). These strategies try to avoid remote accesses to the update sites (target sites) and are invaluable in a situation where it is impossible to access data located at other sites in the distributed systems; for example, in situations where network failure is detected or a high-security database is involved. This means identifying *how to simplify the integrity constraints so that evaluation is more local and at best is entirely local to the sites involved in the update.*

As elaborated in the previous section, the constraints specified in terms of global relations should be transformed into constraints specified in terms of fragment relations so they can be used straightforwardly for constructing efficient enforcement algorithms. The knowledge about the data distribution and the application domain can be used to create algorithms, which derive using this knowledge, an efficient set of fragment constraints from the original constraints. Here, efficient means a set of fragment constraints that is semantically equivalent to the initial set, does not contain any semantic or syntactically redundant fragment constraints/constructs, eliminates any fragment constraints whose evaluation is proved to be true, and eliminates any fragment constraints that contradict already existing fragmentation rules. The derived constraints should be either more local (less distributed) or entirely local when compared to the initial set of constraints. Thus, the properties that one should look for in the derived fragment constraints are that they are *more efficient* and *more local* than the initial set of constraints (Ibrahim, 1998).

Complete tests, which are the tests used most often in centralized systems, are usually expensive in a distributed environment. However, sufficient tests are useful in a distributed environment (Mazumdar, 1993) since their checking space often only spans a single site and therefore can be performed at a reasonable cost as the number of sites involved and the amount of data transferred across the network are reduced. Example: Assuming each fragment is located at different sites of the network, then performing insert(*emp*(*a, b, c, d*)) will require access to the remote sites if the test chosen is the complete test $\mathrm{V}_{i=1}^{n}(\exists x \exists y \exists z)(dept_i(b, x, y, z))$ for the constraint *IC*-1 with Case 1. On the other hand, accessing the remote sites can be avoided if the sufficient test, $(\exists t \exists v \exists w)(emp_i(t, b, v, w))$, is selected.

Constraint simplification methods in distributed environment can be classified into two types of approaches. Both approaches are targeted at

deriving integrity tests, but they use different information. The first approach uses knowledge about the application domain and data fragmentation (Mazumdar, 1993; Qian 1989) and is referred to as *constraint simplification by reformulation*. The second approach analyzes the syntactic structure of the constraints and the update operations, as reported in Gupta (1994) and is referred to as *constraint simplification by update analysis*.

In parallel databases, efficiency is measured by analyzing the execution time taken to check the constraints. Thus, in parallel databases, localizing integrity checking is not as important as making sure each processor is given some processes and is not idle. Although the focus is more toward the time and workload given to the processors, incorporating the strategies utilized in the previous environments can significantly improve the performance of the constraint checking. Thus, the incremental integrity checking strategy and the pretest evaluation should be adopted, as they reduce the number of constraints to be evaluated and avoid the undoing process.

In this environment, two types of processing can be identified; namely, intraprocessing and interprocessing of the constraints. The former is where a single integrity test is broken into several subtests (OR-tests) to be checked concurrently, whereas the latter is where several integrity tests (AND-tests) are selected to be executed concurrently. Therefore, the issue that one should consider is *how to simplify the integrity constraints so that evaluation can be performed in parallel (concurrent) to reduce the total execution time of checking integrity constraints.* In parallel databases, if the processing is based on intraprocessing, then complete tests should be considered, as it is seldom the case where a sufficient test is generated as an OR-test. Example: Given the update operation insert(*emp*(*a, b, c, d*)), an example of an intraprocessing based on the *IC*s given in Figure 1 is when each of the *i*th test of the test $V_{i=1}^{n}(\exists x \exists y \exists z)(dept_i(b, x, y, z))$ is checked concurrently by the processor *i*, which accesses the *i*th region of the *dept* relation. An example of an interprocessing is when the tests $V_{i=1}^{n}(\exists x \exists y \exists z)(dept_i(b, x, y, z))$ and $V_{i=1}^{n}(\exists x \exists y \exists z)(dept_i(b, x, y, z) \wedge (d \leq z))$ are checked concurrently by different processors. Here, we assume the dynamic partitioning strategy has been adopted (Ibrahim, 2006).

Although the performance of constraint checking in distributed database systems and parallel database systems has been improved by adopting centralized strategies, with respect to the amount of data accessed or transferred, these strategies are still inefficient for distributed/parallel environments since most of the simplified forms are derived from the initial constraints as specified by the user. These derivations do not exploit knowledge about the database application, especially its data fragmentation and allocation, which can be used to (a) derive a set of simplified constraints that can be used straightforwardly for constructing efficient enforcement algorithms and (b) infer the information stored at various sites of the network and so minimize the support from remote sites required when checking the constraints (for distributed databases). Example: Given the update operation insert(*emp*(*a, D*1, *c, d*)) and *IC*-1, the tests derived are as follows: complete test $(\exists x \exists y \exists z)(dept(D1, x, y, z))$ and sufficient test $(\exists t \exists v \exists w)(emp(t, D1, v, w))$. For Case 1, it is known that the relations are being fragmented into several fragments; therefore, the tests are transformed into constraints specified in terms of fragments (i.e., $V_{i=1}^{n}(\exists x \exists y \exists z)(dept_i(D1, x, y, z))$ and $V_{i=1}^{n}(\exists t \exists v \exists w)(emp_i(t, D1, v, w))$, respectively). Referring to the fragmentation rules, these tests can be further optimized, and the test that needs to be checked is either $(\exists x \exists y \exists z)(dept_1(D1, x, y, z))$ or $(\exists t \exists v \exists w)(emp_1(t, D1, v, w))$. Assuming the worst case where each fragment is being located at different sites of the network, then checking $(\exists t \exists v \exists w)(emp_1(t, D1, v, w))$ will reduce the amount of data transferred as well as the number of sites involved. This test states that the *IC*-1 is satisfied if there exists at least an employee in the emp_1 fragment who is working in department, *D*1 (i.e.,

deducing the information in the *dept* relation). More examples can be found in Alwan, et al. (2007), Gupta (1994), and Ibrahim (2002).

Table 2 presents the integrity tests derived based on the example given in Figure 1 for the various types of databases along with the amount of data accessed, δ, the amount of data transferred across the network, τ, and the number of sites involved, σ (Ibrahim, 2006).

Table 3 summarizes the differences among centralized, distributed, and parallel databases with respect to the main parameters that should be considered during constraint checking.

FUTURE TRENDS

Although the topic of checking integrity constraints has been explored since the middle of the 1970s (Eswaran & Chamberlin, 1975), it is still one of the main topics discussed in today's conferences and workshops (e.g., the International Conference on Database and Expert Systems Applications—the Second International Workshop on Logical Aspects and Applications of Integrity Constraints (LAAIC'06), Krakow, Poland, 8 September 2006). The issues as discussed in the previous section are still debated. New technologies, especially those employed in the artificial

Table 2. The integrity tests, the amount of data accessed, the amount of data transferred, and the number of sites involved for the emp_dept database

Type of Database	Integrity Test for Update: insert(*emp*(*a*, *D*1, *c*, *d*))	δ	τ	σ	Remarks
Centralized	T^{IC-1}_{comp}:$(\exists x \exists y \exists z)(dept(D1, x, y, z))$	$\delta dept$	0	1	The parameter that can be reduced is the amount of data accessed.
	T^{IC-2}_{comp}:$(\exists x \exists y \exists z)(dept(D1, x, y, z) \wedge (d \leq z))$	$\delta dept$	0	1	
Distributed - Case 1	T^{IC-1}_{comp}:$(\exists x \exists y \exists z)(dept_1(D1, x, y, z))$	$\delta dept_1$	$\delta dept_1$	2	No data are being transferred across the network, and the number of sites involved is always one for sufficient test.
	T^{IC-1}_{suff}:$(\exists t \exists v \exists w)(emp_1(t, D1, v, w))$	δemp_1	0	1	
	T^{IC-2}_{comp}:$(\exists x \exists y \exists z)(dept_1(D1, x, y, z) \wedge (d \leq z))$	$\delta dept_1$	$\delta dept_1$	2	
	T^{IC-2}_{suff}:$(\exists t \exists v \exists w)(emp_1(t, D1, v, w) \wedge (w \geq d))$	δemp_1	0	1	
Distributed - Case 2	T^{IC-1}_{comp}:$V^2_{i=1}(\exists x \exists y \exists z)(dept_i(D1, x, y, z))$	$[\delta dept_i, \delta dept]$	$[\delta dept_i, \delta dept]$	3	
	T^{IC-1}_{suff}:$(\exists t \exists v \exists w)(emp_1(t, D1, v, w))$	δemp_1	0	1	
	T^{IC-2}_{comp}:$V^2_{i=1}(\exists x \exists y \exists z)(dept_i(D1, x, y, z) \wedge (d \leq z))$	$[\delta dept_i, \delta dept]$	$[\delta dept_i, \delta dept]$	3	
	T^{IC-2}_{suff}:$(\exists t \exists v \exists w)(emp_1(t, D1, v, w) \wedge (w \geq d))$	δemp_1	0	1	
Parallel - Interprocessing	T^{IC-1}_{comp}: $V^n_{i=1}(\exists x \exists y \exists z)(dept_i(D1, x, y, z))$ T^{IC-2}_{comp}: $V^n_{i=1}(\exists x \exists y \exists z)(dept_i(D1, x, y, z) \wedge (d \leq z))$ $T^{IC-1-2}_{comp-AND}$:$T^{IC-1}_{comp} \wedge T^{IC-2}_{comp}$	$2\delta dept$	0	1	The parameter that can be reduced is the time taken to check the tests.
Parallel - Intraprocessing	$T^{IC-1}_{comp-OR}$: $V^n_{i=1}(\exists x \exists y \exists z)(dept_i(D1, x, y, z))$ $T^{IC-2}_{comp-OR}$: $V^n_{i=1}(\exists x \exists y \exists z)(dept_i(D1, x, y, z) \wedge (d \leq z))$	$\delta dept$	0	1	

Note: δ – the amount of data accessed, τ - the amount of data transferred, σ - the number of sites; comp – complete test, suff – sufficient test, δR – the size of relation R, [δmin, δmax] – between the size min and max

Table 3. Constraints checking in centralized, distributed, and parallel databases

Type of Database	Parameters to be Considered	Strategies That Can Be Adopted
Centralized	Reduce the number of integrity constraints that needs checking	Syntactic analysis – incremental integrity checking
		Semantic analysis
	Avoid undoing the updates	Preventive methods
	Reduce the amount of data to be accessed	Syntactic analysis – constraint simplification by update analysis (complete test)
		Semantic analysis – constraint reformulation
Distributed	Reduce the number of integrity constraints that needs checking	Syntactic analysis – incremental integrity checking
		Semantic analysis
	Avoid undoing the updates	Preventive methods
	Reduce the amount of data to be accessed	Syntactic analysis – constraint simplification by update analysis (sufficient test) Semantic analysis – constraint reformulation
	Reduce the amount of data transferred across the network	
	Reduce the number of sites	
Parallel	Reduce the number of integrity constraints that needs checking	Syntactic analysis – incremental integrity checking
		Semantic analysis
	Reduce the amount of data to be accessed	Syntactic analysis – constraint simplification by update analysis (complete and sufficient tests) Semantic analysis – constraint reformulation
	Reduce the total execution time	

intelligent field such as agents, data mining, and fuzzy set, are now being investigated to examine their potential in enhancing the performance of deriving and checking constraints.

Mobile agents, which are autonomous active programs that can move both data and functionality (code) to multiple places in a distributed system, are now becoming popular as a technique for checking databases' constraints. It is believed that mobile agents can be used to speed up the process of checking integrity constraints. Data mining, on the other hand, can be used to identify the pattern that occurs in a database to derive the relationships among the data. Based on these relationships, integrity constraints can be generated automatically. This can reduce the burden of the user in specifying the correct and complete set of business rules, while fuzzy set is adopted to identify the range of possible consistent values in which consistency of the database is flexible and not as rigid as the traditional approach.

Interestingly, these technologies are explored not only in distributed databases but also in mobile databases. Checking and maintaining database integrity in a mobile environment require different mechanisms from those used in centralized and distributed databases due to the following facts: (i) characteristics of the mobile environment, which include high communication latency, intermittent wireless connectivity, limited battery life, and changing client location (Elmasri & Navathe, 2007); (ii) data is unevenly distributed among the base stations and mobile units; the consistency constraints compound the problem of cache management, and caches attempt to provide the most frequently accessed and up-

dated data to mobile units that process their own transactions and may be disconnected over long periods (Elmasri & Navathe, 2007); (iii) integrity constraints may spread over several base stations in the system; a large amount of data may therefore need to be transferred around the network in order to determine the truth of such statements (Ibrahim, 2006); (iv) frequent updates can lead to frequent executions of expensive violation testing operations (Ibrahim, 2006); and (v) allowing an update to execute with the intention of aborting it at commit time in the event of constraint violation is also inefficient since rollback and recovery must occur at all stations that participated in the update, which can be a very costly operation in mobile systems (Ibrahim, 2006).

Mazumdar and Chrysanthis (2004) adopt the localization technique of distributed database in mobile databases in which a distributed constraint is reformulated into local constraints. Only two types of constraints are considered; namely, equality and inequality constraints (set-based constraints).

Madiraju and Sunderraman (2004) propose a framework of a mobile agent-based approach for checking global constraints. The approach generates subconstraint checks on multiple sites and sends multiple remote agents for checking the subconstraints. They claimed that mobile agents could speed up the process of checking constraints since checking the consistency of remote databases can be executed in parallel by the mobile agents.

Ibrahim (2007) proposes an agent-based semantic integrity subsystem framework, AbSIS, for mobile databases. Three types of agents have been identified; namely, *Local Agent*, *Remote Agent*, and *Repair Agent*. These agents are responsible to ensure that the databases in the mobile environment are always consistent.

The works reported here are still in their early stages, as no concrete findings (results) have been published. We believe the capabilities and characteristics of artificial intelligence technologies, especially in deducing (inferring) new information based on the current or incomplete information, handling incomplete data, and fuzzy conditions, will be explored greatly in the future and can significantly improve the performance of the database systems, especially when dealing with the processes of maintaining the consistency and checking the integrity of the database, regardless of the environments.

CONCLUSION

An important aim of a database system is to guarantee database consistency, which means that data contained in a database are both accurate and valid. There are many ways that inaccurate data may occur in a database. Several factors and issues have been highlighted with regard to checking integrity constraints in centralized, distributed, and parallel databases. These factors and issues are the challenges in devising an efficient enforcement mechanism.

REFERENCES

Alwan, A.A., Ibrahim, H., & Udzir, N.I. (2007). *Local integrity checking using local information in a distributed database*. Proceedings of the 1st Aalborg University IEEE Student Paper Contest 2007 (AISPC'07), Denmark.

Barbara, D., & Garcia-Molina, H. (1992). *The demarcation protocol: A technique for maintaining linear arithmetic constraints in distributed database systems*. Proceedings of the Conference on Extending Database Technology, LNCS 580, Austria, 373–388.

Bernstein, P.A., & Blaustein, B.T. (1981). *A simplification algorithm for integrity assertions and concrete views*. Proceedings of the 5th International Computer Software and Applications Conference, 90–99.

Blaustein, B.T. (1981). Enforcing database assertions: Techniques and applications [doctoral thesis]. Harvard University.

Christiansen, H., & Martinenghi, D. (2005). *Incremental integrity checking: Limitations and possibilities.* Proceedings of the 12th International Conference on Logic for Programming and Artificial Intelligence and Reasoning, Jamaica, 712–727.

Date, C.J. (2003). *An introduction to database systems.* Pearson Addison Wesley.

Elmasri, R., & Navathe, S.B. (2007). *Fundamentals of database systems.* Pearson Addison Wesley.

Eswaran, K.P., & Chamberlin, D.D. (1975). *Functional specifications of a subsystem for data base integrity.* Proceedings of the 1st International Conference on Very Large Data Bases (VLDB 1), 48–68.

Grefen, P.W.P.J. (1990). *Design considerations for integrity constraint handling in PRISMA/DB1.* Prisma Project Document P508. The Netherlands.

Grefen, P.W.P.J. (1993). *Combining theory and practice in integrity control: A declarative approach to the specification of a transaction modification subsystem*. Proceedings of the 19th International Conference on Very Large Data Bases, Ireland, 581–591.

Gupta, A. (1994). *Partial information based integrity constraint checking* [doctoral thesis]. Stanford University.

Hanandeh, F.A.H. (2006). *Integrity constraints maintenance for parallel databases* [doctoral thesis]. Malaysia: Universiti Putra Malaysia.

Hanandeh, F.A.H., Ibrahim, H., Mamat, A., & Johari, R. (2004). Virtual rule partitioning method for maintaining database integrity. *The International Arab Journal of Information Technology, 1*(1), 103–108.

Henschen, L.J., McCune, W.W., & Naqvi, S.A. (1984). Compiling constraint-checking programs from first-order formulas. *Advances in Database Theory, 2,* 145–169.

Hsu, A., & Imielinski, T. (1985). *Integrity checking for multiple updates.* Proceedings of the 1985 ACM SIGMOD International Conference on the Management of Data, 152–168.

Ibrahim, H. (1998). *Semantic integrity constraints enforcement for a distributed database* [doctoral thesis]. UK: University of Cardiff.

Ibrahim, H. (2002). *A strategy for semantic integrity checking in distributed databases.* Proceedings of the Ninth International Conference on Parallel and Distributed Systems, IEEE Computer Society, China.

Ibrahim, H. (2006). *Checking integrity constraints—How it differs in centralized, distributed and parallel databases.* Proceedings of the 17th International Conference on Database and Expert Systems Applications—The Second International Workshop on Logical Aspects and Applications of Integrity Constraints (LAAIC'06), Poland, 563–568.

Ibrahim, H. (2007). *AbSIS—An agent-based semantic integrity subsystem for managing mobile databases' constraints.* Proceedings of the 2007 World Congress in Computer Science, Computer Engineering, and Applied Computing—the 2007 International Conference on Information and Knowledge Engineering (IKE'07), Las Vegas, Nevada.

Ibrahim, H., Gray, W.A., & Fiddian, N.J. (1996). *The development of a semantic integrity constraint subsystem for a distributed database (SICSDD).* Proceedings of the 14th British National Conference on Databases, UK, 74–91.

Ibrahim, H., Gray, W.A., & Fiddian, N.J. (1998). *SICSDD—A semantic integrity constraint subsystem for a distributed database*. Proceedings

of the 1998 International Conference on Parallel and Distributed Processing Techniques and Applications, 1575–1582.

Madiraju, P., & Sunderraman, R. (2004). *A mobile agent approach for global database constraint checking.* Proceedings of the ACM Symposium on Computing (SAC '04), Cyprus.

Martinenghi, D. (2005). *Advanced techniques for efficient data integrity checking* [doctoral thesis]. Roskilde University.

Mazumdar, S. (1993). *Optimizing distributed integrity constraints.* Proceedings of the 3rd International Symposium on Database Systems for Advanced Applications, Korea, 327–334.

Mazumdar, S., & Chrysanthis, P.K. (2004). *Localization of integrity constraints in mobile databases and specification in PRO-MOTION.* Proceedings of the Mobile Networks and Applications, 481–490.

McCarroll, N.F. (1995). *Semantic integrity enforcement in parallel database machines* [doctoral thesis]. UK: University of Sheffield.

McCune, W.W., & Henschen, L.J. (1989). Maintaining state constraints in relational databases: A proof theoretic basis. *Journal of the Association for Computing Machinery, 36*(1), 46–68.

Nicolas, J.M. (1982). Logic for improving integrity checking in relational data bases. *Acta Informatica, 18*(3), 227–253.

Plexousakis, D. (1993). *Integrity constraint and rule maintenance in temporal deductive knowledge bases.* Proceedings of the 19th International Conference on Very Large Data Bases, Ireland, 146–157.

Qian, X. (1989). *Distribution design of integrity constraints.* Proceedings of the 2nd International Conference on Expert Database Systems, 205–226.

Simon, E., & Valduriez, P. (1987). Design and analysis of a relational integrity subsystem. MCC Technical Report DB-015-87.

KEY TERMS

Centralized DBMS: All the DBMS functionality, application program execution, and user interface processing were carried out on one machine.

Constraint Checking: The process of ensuring the integrity constraints are satisfied by the database after it has been updated.

Distributed DBMS: The actual database and DBMS software distributed over many sites, connected by a computer network.

Integrity Constraints: A formal representation of a property that a database is required to satisfy at any time in order to faithfully describe the real world represented by the database.

Integrity Control: Deals with the prevention of semantic errors made by users due to their carelessness or lack of knowledge.

Integrity Tests: Integrity checks that verify whether or not the database is satisfying its constraints.

Parallel Database Management Systems: Database management systems developed using shared memory (tightly coupled) architecture or shared disk (loosely coupled) architecture.

Chapter XLI
Data Quality Assessment

Juliusz L. Kulikowski
Institute of Biocybernetics and Biomedical Engineering PAS, Warsaw, Poland

INTRODUCTION

For many years the fact that for a high information processing systems' effectiveness high quality of data is not less important than high systems' technological performance was not widely understood and accepted. The way to understanding the complexity of data quality notion was also long, as it will be shown below. However, a progress in modern information processing systems development is not possible without improvement of data quality assessment and control methods. Data quality is closely connected both with *data form* and *value of information* carried by the data. High-quality data can be understood as data having an appropriate form and containing valuable information. Therefore, at least two aspects of data are reflected in this notion: 1st - technical facility of data processing, and 2nd - usefulness of information supplied by the data in education, science, decision making, etc.

BACKGROUND

In the early years of information theory development a difference between the *quantity* and the *value* of information was noticed; however, originally little attention was paid to the information value problem. R. Hartley interpreting information value as its psychological aspect stated that it is desirable to eliminate any additional psychological factors and to establish an information measure based on purely physical terms only (Klir 2006, pp. 27-29). C.E. Shannon and W. Weaver created a mathematical communication theory based on statistical concepts, fully neglecting the information value aspects (Klir ,2006, p.68). In most of later works concerning information theory backgrounds attention was focused on extension of the *uncertainty* concept rather than on this of information value. Nevertheless, L. Brillouin tried to establish a relationship between the quantity and the value of information stating that for an

information user the relative information value is smaller than or equal to the absolute information, i.e. to its quantity (Brillouin,1956, Chapt. 20.6). M.M. Bongard (Bongard, 1960) and A.A. Kharkevitsch (Kharkevitsch, 1960) have proposed to combine the information value concept with the one of a statistical decision risk. This concept has also been developed by R.L. Stratonovitsch (Stratonovitsch, 1975, Chapts. 9, 10). This approach leads to an economic point of view on information value as profits earned due to information using (Beynon-Davies, 1998, Chapt. 34.5). Such approach to information value assessment is limited to the cases in which economic profits can be quantitatively evaluated. In physical and technical measurements data accuracy (described by a mean-square error or by a confidence interval length) is used as the main data quality descriptor. In medical diagnosis data actuality, relevance and credibility as well as their influence on diagnostic sensitivity and specificity play relatively higher role than data accuracy (Wulff, 1981). This indicates that, in general, no universal set of data quality descriptors exists; they rather should be chosen according to the application area specificity. In the last years data quality became one of the main problems posed by the world wide web (WWW) development (Baeza-Yates & B. Ribeiro-Neto, 1999, Chapt. 13.2). The focus in the domain of finding information in the WWW increasingly shifts from merely locating relevant information to differentiating high-quality from low-quality information (Oberweis & Perc, 2000, pp. 14-15). In the recommendations for databases of the Committee for Data in Science and Technology (CODATA) several different quality types of data are distinguished: 1st primary (rough) data whose quality is subjected to individually or locally accepted rules or constraints, 2nd qualified data, broadly accessible and satisfying national or international (ISO) standards in the given application domain, 3rd recommended data – the highest quality broadly accessible data (like physical fundamental constants) that have passed a set of special data quality tests. In the last decades several technological tools for formal data incorrectness detection and rectifying have been proposed (Shankaranarayan & Ziad & Wang, 2003). In some countries the interests of information users are legally protected from distribution of certain types of incredible or misguided data. On the other hand, a governmental intervention into the activity of open-access databases is also limited by international legal acts protecting human rights to free distribution of information.

BASIC PROBLEMS OF DATA QUALITY ASSESSMENT

The idea that information value can be better characterized by a multi-component vector than by a scalar value arose in late 60-ths of the 20th century. In such case the following problems arise: 1st, what information (or – data, as its particular form) aspects should be taken into account for its value characterization, 2nd, what does it mean that a multi-aspect information value is "better" or "higher" than another one, 3rd, how simple data multi-aspect values can be extended on higher-level data structures. J.L. Kulikowski (Kulikowski, 1971) proposed to describe information value by an element of semi-ordered linear vector space (Kantorovitsch space). For a multi-aspect data quality evaluation data validity (actuality), relevance, credibility, accuracy, and operability as quality factors were originally proposed. In the ensuing years the list of proposed data quality factors by other authors has been extended up to almost two hundreds (Wang & Strong, 1996; Shanks & Darke, 1998; Pipino & Lee & al., 2002). Between arbitrarily chosen data quality factors hidden functional dependence may exist, as illustrated in Fig. 1.

Fig. 1a illustrates a hypothetical dependence between data validity v_{vd} and data operability v_{op}: improving data operability is a time-consuming operation reducing data validity. Fig 1b shows that

data validity reduction can also be caused by the necessity of data credibility (v_{cr}) proving. Fig 1c illustrates a data accuracy v_{acc} (e.g. inversion of a statistical measurement error) influencing on data relevance v_{rel} when high data accuracy for effective decision making is desired.

Any selection of a collection of data value factors leads to the problem of multi-aspect data quality instances comparison. For example, it may arise a problem of practical importance: what kind of data in a given situation should be preferred: the more fresh but less credible or outdated but very accurate ones. Comparison of multi-aspect data qualities is possible, in particular, when the following formal conditions are fulfilled: 1st - the quality factors are represented by non-negative real numbers, 2nd - if taken together, they form a vector in a linear semi-ordered vector space, 3rd - they are defined so that the multi-aspect quality vector is a non-decreasing function of its components (quality factors). For example, if t_0 denotes the time of data creation and t is a current time, then for $t_0 \le t$ neither the difference $t_0 - t$ (as not satisfying the 1st condition) nor $t - t_0$ (as not satisfying the 3rd one) can be used as data actuality measures. However, if T, such that $0 \le t_0 \le T$, denotes the maximum data validity time then for $t \le T$ a difference $\theta = T - t$ or a normalized difference $\Theta = \theta / T$, both satisfying the above-mentioned conditions, can be used as validity measures characterizing one of data quality aspects. For similar reasons, if Δ is a finite length of admissible data values and δ is the length of data confidence interval, where $0 < \delta < \Delta$, then $c = 1/\delta$ or $C = \Delta/\delta$, rather than δ, can be used as data accuracy measures.

Comparison of Multi-Aspect Data Qualities in a Linear Vector Space

For any pair of vectors ***Q'***, ***Q"*** describing multi-aspect qualities of some single data they can be compared according to the semi-ordering rules in the given linear vector space (Akilov & Kukate-ladze, 1978). Linearity of the vector space admits the operations of vectors adding, multiplying by real numbers and taking the differences of vectors. For comparison of vectors a, so called, *positive cone* K^+ in the vector space should be established, as shown in Fig. 2. It is assumed that ***Q'*** *has lower value than* ***Q"*** (***Q'*** $\prec$ ***Q"***) if their difference $\Delta \boldsymbol{Q} = \boldsymbol{Q"} - \boldsymbol{Q'}$ is a vector belonging to K^+. If neither ***Q'*** $\prec$ ***Q"*** nor ***Q"*** $\prec$ ***Q'*** then the vectors ***Q'***, ***Q"*** are called *mutually incomparable*; the last means that no quality vector in this pair with respect to the other one can be preferred. Fig. 2a shows a situation of ***Q'*** $\prec$ ***Q"***, while Fig. 2b a one of vectors' incomparability caused by a narrower positive cone (between the dotted lines) being assumed.

Figure 1. Functional dependence between selected data quality factors

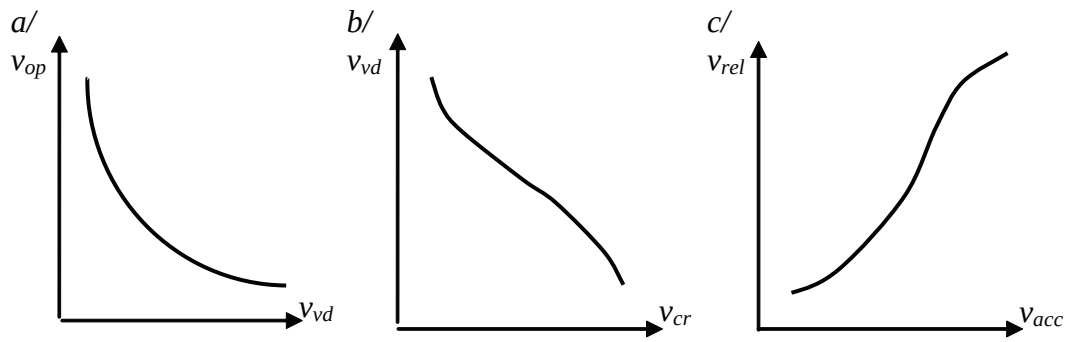
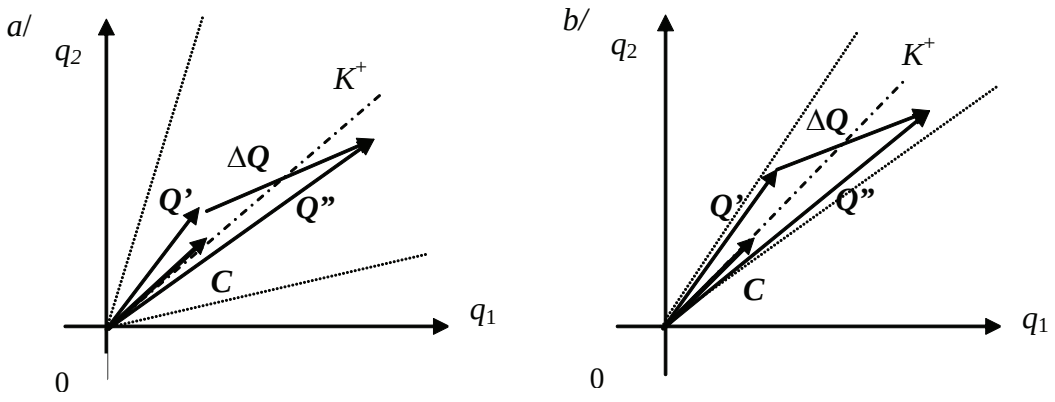


Figure 2. Principles of comparison of vectors describing multi-aspect data values



The example shows that the criteria of multi-aspect data qualities' comparability can be changed according to the user's requirements: they become more rigid when the positive cone K^+ established is narrower. In particular, if K^+ becomes as narrow as being close to a positive half-axis, then $\boldsymbol{Q'} \prec \boldsymbol{Q''}$ if all components of $\boldsymbol{Q'}$ are proportional to the corresponding ones of $\boldsymbol{Q''}$ with the coefficient of proportionality <1. On the other hand, if K^+ becomes as large as the positive sector of the vector space then $\boldsymbol{Q'} \prec \boldsymbol{Q''}$ means that all components of $\boldsymbol{Q'}$ are lower than the corresponding ones of $\boldsymbol{Q''}$; in practice the domination of the components of $\boldsymbol{Q''}$ over those of $\boldsymbol{Q'}$ may be not significant excepting one or several selected components. Between the above-mentioned two extremities there is a large variety of data quality vectors semi-ordering including those used in multi-aspect optimization theory. In particular, if $\boldsymbol{Q'} = [q_1', q_2', \ldots, q_k']$ and $\boldsymbol{Q''} = [q_1'', q_2'', \ldots, q_k'']$ are two vectors then their scalar product:

$$(\boldsymbol{Q'}, \boldsymbol{Q''}) = q_1' \cdot q_1'' + q_2' \cdot q_2'' + \ldots + q_k' \cdot q_k''$$

and the length (a norm) of a vector $\boldsymbol{Q}$ given by the expression:

$$| Q | = \sqrt{(Q,Q)}$$

can be defined. Then any fixed positive-components vector $\boldsymbol{C}$ of a norm $\|\boldsymbol{C}\| = 1$ indicates the direction of the K^+ cone axis. For any given data quality vector $\boldsymbol{Q}$ an angle $\angle(\boldsymbol{C}, \boldsymbol{Q})$ between the vectors $\boldsymbol{C}$ and $\boldsymbol{Q}$ can be calculated from its cosine given by:

$$\cos \angle(\boldsymbol{C}, \boldsymbol{Q}) = (\boldsymbol{C}, \boldsymbol{Q}) / \|\boldsymbol{Q}\|$$

The last expression can also be used to data quality comparison: for the given pair of quality vectors it is assumed that $\boldsymbol{Q'} \prec \boldsymbol{Q''}$ if and only if $\cos \angle (\boldsymbol{C}, \boldsymbol{Q'}) < \cos \angle (\boldsymbol{C}, \boldsymbol{Q''})$ (i.e. when $\boldsymbol{Q''}$ is closer to $\boldsymbol{C}$ than $\boldsymbol{Q'}$). This type of vectors comparison leads, in fact, to a comparison of weighted linear combinations of data quality factors.

Remarks on Composite Data Quality Assessment

The above-described single data quality assessment principles can be extended on composite data and on higher-order data structures. For example, if a structured data record has the form:

Identifier	*Q*1	*Q*2	*Q*3	.....	*Qn*

where *Q*1, *Q*2,...,*Qn* denote some single data whose current topicality, relevance, etc. have been

evaluated then it arises the problem of evaluation of the topicality, relevance, etc. of the record as a whole. In general, it is not a trivial problem, the higher-order data structures usually being composed of various types of simple data. The quality of a record consisting of a linearly ordered set of single data (say, a series of parameters describing the properties of a given physical object) can be defined as a non-decreasing vector function of the component data qualities (Kulikowski, 2002, p. 120). Several variants of such functions can be taken into account: the *minimum*, the *maximum*, the *weighted mean value* of the arguments, etc. For example, if ***Q'*** and ***Q"*** are multi-aspect qualities of two simple data and q_1', q_1" is a pair of their first quality factors, q_2', q_2" is a pair of their second quality factors, etc. then the quality of the pair [***Q'***, ***Q"***] can be defined as

$$\boldsymbol{Q} = [min(q_1', q_1''), min(q_2', q_2''),..., min(q_k', q_k'')].$$

The formula can easily be extended on any (n) number of simple data (components of data record). The *minimum* variant leads to a "careful" strategy of data using: a record satisfies user's expectations when all its components do so. On the other hand, a *maximum* variant follows from an "optimistic" strategy: a record as a whole satisfies user's expectations if they are satisfied by at least one of the record's components. Using other, less rigorous functions (like the *weighted mean value*) satisfying the general multi-aspect data quality assessment conditions makes possible adjustment of data quality requirements to users' needs. Additional vector components describing composite data quality factors, if necessary, can be introduced. They may characterize, for example, composite data completeness, irredundancy, etc. which are meaningless in simple data quality assessment. Other quality factors in higher-order data structures assessment can be used, respectively. Quality assessment of higher-level non-structured files (textual, visual, multi-media, etc. documents) can be based on meta-data characterizing files' source, form, scientific or artistic level, technical quality, expected users, etc. Effectiveness of such meta-data as tools used to information retrieval in data (files) repositories can be increased due to meta-data driven query languages designed for given application areas. In distributed databases increasing attention to data interoperability as to an important quality factor is paid. H. Sugawara (Sugawara, 2002) states that it may be a solution to provide an integrated view of heterogenous data sources distributed in many disciplines and also in distant places. For this purpose standardization of scientific terminology, data standards and metadata standards for document type definition are developed. Limited quality of data, in general, is not an obstacle in using them to decision making, because in last years various methods of decision making under uncertainty have been developed (Bubnicki, 2002). However, as some authors remark (Orr, 1998; Redman, 1998), in well-organized databases data quality should be carefully controlled.

FUTURE TRENDS

It can be expected that further progress in data quality assessment and control methods, as well as the corresponding technological tools in the nearest years will be achieved. New methods of automatic data quality control based on sophisticated mathematical and/or logical tools and on artificial intelligence approach will be applied. Advanced data mining techniques can be used to data credibility assessment. Some standard algorithms of this type will be included into commercial database management systems. Data quality assessment in the case when particular quality aspects can not be but with a dose of uncertainty evaluated remains still a problem which should be solved. Special attention will be paid to high data quality guarantee reaching in distributed multi-agent information processing systems.

Specialized organizations will act on specification and formulation of data quality requirements corresponding to their professional needs. As a consequence, high data quality requirements will also be included into international and/or national data processing standards. Due to a progress in investigation of legal, ethical, economical and/or organizational aspects of high-quality data acquirement, storage, and distribution the effectiveness of data- and/or knowledge-based information systems will be increased. A basic role in this area should be played by world-wide organizations: International Standard Organization (ISO), Committee for Data in Science and Technology (CODATA), International Federation of Information Processing (IFIP) , etc.

CONCLUSION

In most applications high data quality is not less important than easy access to data. Data quality is, in general, independent on data volume and closely related to the data users' needs. Despite the fact that no absolute and universal data quality measure exists, a variety of approaches to data quality definition by many authors has been proposed. However, the proposals of data quality assessment are useful only if the quality factors are measurable, the vectors of quality factors are comparable, and single data quality assessment methods can be extended on higher-order data structures. For this purpose non-trivial mathematical tools of data quality assessment should be used. They make possible choosing a data quality method the most suitable one to any given application domain and to the data users' requirements.

REFERENCES

Akilov, G. P., & Kukateladze, S. S. (1978). *Ordered Vector Spaces* (in Russian). Nauka, Novosibirsk.

Baeza-Yates, R., & Ribeiro-Neto, B. (1999). *Modern Information Retrieval*. Harlow, England: Addison-Wesley

Beynon-Davies, P. (1998). Information Systems Development. *An Introduction to Information Systems Engineering.* MacMillan Press Ltd.

Bongard, M. M. (1960). *On the Concept of "Useful Information"* (in Russian). Problemy kibernetiki, 4, Moscow.

Brillouin, L. (1956). *Science and Information Theory*. New York: Acad. Press Inc. Publishers.

Bubnicki, Z. (2002).*Uncertain Logics, Variables and Systems.* Berlin: Springer.

Kharkevitsch, A. A. (1960). *On Information Value* (in Russian). Problemy kibernetiki, 4, Moscow.

Klir, G. J. (2006). Uncertainty and Information. Foundations of Generalized Information Theory. Hoboken NJ: John Wiley & Sons Interscience.

Kulikowski, J. L. (1971). Notes on the Value of Information Transmitted through a Communication Channel. *2nd International Symposium on Information Theory, Cachkadzor, ASSR. Conf. Proceedings,* Akademiai Kiado, Budapest, (pp. 61-71).

Kulikowski, J. L. (2002). Multi-Aspect Evaluation of Data Quality. *In Scientific Databases. 18th International CODATA Conference.* Book of Abstracts, Montreal, Canada.

Oberweis, A., & Perc, P. (2000). Information Quality in the World Wide Web. *17th International CODATA Conference.* Book of Abstracts, Baveno, Italy.

Orr, K. (1998). Data Quality and Systems Theory. *Communications of the ACM, 41*(2), 66-71.

Pipino, L. L., Lee, Y. W., et al. (2002). Data Quality Assessment. *Communications of the ACM 45*(4), 211-218.

Redman, T. C. (1998). The Impact of Poor Data Quality on the Typical Enterprise. *Communications of the ACM, 41*(2), 79-82.

Shankaranarayan, G., Ziad, M., & Wang, R. Y. (2003). Managing Data Quality in Dynamic Decision Environment: An Information Product Approach. *Journal of Database Management, 14*(4).

Shanks, G., & Darke, P. (1998). A Framework for Understanding Data Quality. *Journal of Data Warehousing 3*(3), 46-51.

Stratonovitsch, R. L. (1975). *Teoria informacii* (in Russian). Sovetskoe Radio, Moscow.

Sugawara, H. (2002). Interoperability of Biological Data Resources. *18th International CODATA Conference*. Book of Abstracts, Montreal, Canada, (p. 9).

Wang, R. Y., & Strong, D. (1996). Beyond Accuracy: What Data Quality Means to Data Consumers. *Journal of Management Information Systems, 12*(4), 5-34.

Wulff, H. R. (1981). *Rationel klinik*. Munksgaard, Copenhagen.

KEY TERMS

Data Accuracy: An aspect of numerical (→) data quality connected with a standard statistical error between a real parameter value and the corresponding value given by the data. Data accuracy is inversely proportional to this error.

Data Actuality: → Data validity.

Data Completeness: Containing by a composite data all components necessary to full description of the states of a considered object or process.

Data Credibility: An aspect of (→) data quality: a level of certitude that the (→) data content corresponds to a real object or has been obtained using a proper acquisition method.

Data Irredundancy: The lack of data volume that by data recoding could be removed without information loss.

Data Legibility: An aspect of (→) data quality: a level of data content ability to be interpreted correctly due to the known and well-defined attributes, units, abbreviations, codes, formal terms, etc. used in the data record's expression.

Data Operability: An aspect of (→) data quality: a level of data record ability to be used directly, without additional processing: restructuring, conversion, etc.

Data Quality: A set of data properties (features, parameters, etc.) describing their ability to satisfy user's expectations or requirements concerning data using for information acquiring in a given area of interest, learning, decision making, etc.

Data Relevance: An aspect of (→) data quality: a level of consistency between the (→) data content and the area of interest of the user.

Data Validity: An aspect of (→) data quality consisting in its steadiness despite the natural process of data obsolescence increasing in time.

Chapter XLII
Measuring Data Quality in Context

G. Shankaranarayanan
Boston University School of Management, USA

Adir Even
Ben Gurion University of the Negev, Israel

INTRODUCTION

Maintaining data at a high quality is critical to organizational success. Firms, aware of the consequences of poor data quality, have adopted methodologies and policies for measuring, monitoring, and improving it (Redman, 1996; Eckerson, 2002). Today's quality measurements are typically driven by physical characteristics of the data (e.g., item counts, time tags, or failure rates) and assume an objective quality standard, disregarding the context in which the data is used. The alternative is to derive quality metrics from data content and evaluate them within specific usage contexts. The former approach is termed as *structure-based (or structural),* and the latter, *content-based* (Ballou and Pazer, 2003). In this chapter we propose a novel framework to assess data quality within specific usage contexts and link it to data utility (or utility of data) - a measure of the value contribution associated with data within specific usage contexts. Our utility-driven framework addresses the limitations of structural measurements and offers alternative measurements for evaluating completeness, validity, accuracy, and currency, as well as a single measure that aggregates these data quality dimensions.

BACKGROUND

Data quality is defined as fitness-for-use – the extent to which the data matches the data consumer's needs (Redman, 1996). However, in real-life set-

tings, a single definition of the data quality may fail to support data management needs (Strong et al, 1997, Lee and Strong, 2003). Kulikowski (1971) suggests that data quality should be measured as a multi-dimensional vector that reflects different aspects of quality. Wang and Strong (1996) show that data customers perceive quality as having multiple dimensions such as accuracy, completeness, and currency. Quality, along each dimension, is often measured as a number between *0* (poor) and *1* (perfect). Pipino et al. (2002) identify three archetypes for quality metrics that adhere to this scale: (a) ratio between the actually obtained and the expected values, (b) min/max value among aggregations and (c) weighted average between multiple factors. Different measurement methods have been proposed along these archetypes (e.g., Redman, 1996; Pipino et al., 2002). Such measurements can be stored as quality metadata (Shankaranarayanan and Even, 2004), presented by software tools (Wang, 1998; Shankaranarayanan and Cai, 2006), tied to visual representations of data processes (Shankaranarayanan et al., 2003), and used for process optimization (Ballou et al., 1998).

Some quality dimensions (e.g., accuracy) are viewed as impartial (Wang and Strong, 1996) - i.e., the perception of quality along these dimensions is based on the data itself, regardless of usage. Others are viewed as contextual quality dimensions and perception of quality depends on the usage context (e.g., relevance). Pipino et al. (2002), however, argue that the same dimension can be measured impartially and/or contextually, depending on the purpose the measurement serves. As both impartial assessment and contextual assessment contribute to the overall perception of data quality, it is important to address both. We posit that within a usage context, the business value of data resources is reflected more by the data content and less by physical characteristics. Hence, we suggest that content-based measurement of quality is more appropriate for contextual assessment. We use utility functions (Ahituv, 1980) to link impartial information characteristics (here, data contents and presence of defects) onto tangible values within specific usages. Utility mapping has been used to examine tradeoffs between quality dimensions and optimize their configuration (Ballou et al., 1998; Ballou and Pazer, 1995, 2003).

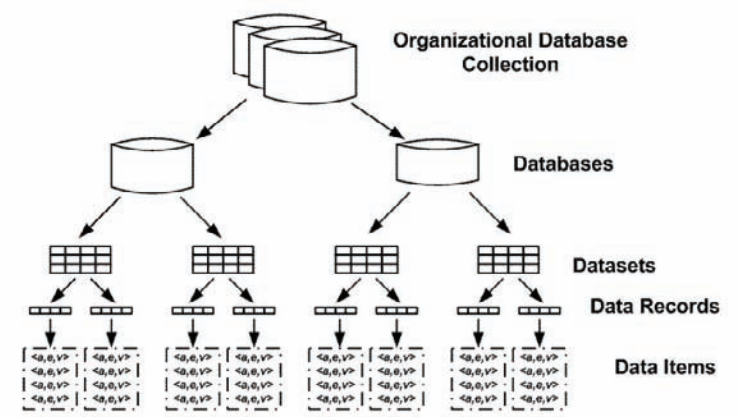
The quality measurements proposed here are based on the traditional data hierarchy (adapted from Redman, 1996). The foundation of this hierarchy (figure 1) is the data item. The data item is defined as a triplet <*a*,*e*,*v*> of a data value '*v*' selected from the value domain attached to attribute '*a*' of entity '*e*' that represents a physical or logical real-world object. The data record is a collection of data items that represent the attributes of an entity instance. A dataset is a collection of records that belong to the same entity class (e.g., a subset of records in a table), and a database is a collection of datasets with meaningful interrelationships. Organizations typically have a collection of databases. Certain measurements evaluated here are defined at different hierarchical levels. The annotation used to differentiate the same measurement between levels is described in the glossary at the end of the chapter.

The framework examines the tabular dataset, assuming *N* identically structured records (rows, indexed by *[n]*), and *M* attributes per record (columns, indexed by *[m]*). Data contents, the actual attribute values of record *[n]*, are denoted $f^E_{n,1}$ through $f^E_{n,M}$. Each attribute has a valid set of values (e.g., integer, real, alphanumeric, or a finite set) defined by its value domain. We next develop the concept of utility specifically for datasets. We then use it to develop content-based and contextual data quality measurements at different data-hierarchy levels.

UTILITY OF DATA

The utility of a data resource is a non negative measurement of its value contribution. In commercial

Figure 1. Data hierarchy



usages, data utility can be measured in monetary units. In other usage types (e.g., scientific research, medical, military) alternative measurements may capture utility more adequately (Ahituv, 1980). Regardless, our computations are indifferent to the utility-measurement units if used consistently. The models developed here account for multiple usages of data. The usages are assumed orthogonal – i.e., the utility gained by one usage is independent of others. In some real-life scenarios, dependency may exist between usages and the model should be revised accordingly.

The utility of a single dataset can be attributed along its records. First consider I known usages (indexed by *[i]*), where each has a maximum possible utility of $U^D_i \geq 0$. This maximum is obtained when the entire dataset is available and has no quality defects. The presence of defects might reduce the utility to some extent. The maximum utility U^D_i can be allocated between dataset records, based on the relative importance of record *[n]* to usage *[i]*, such that the sum of record utilities $\{U^R_{i,n}\}$ equals the maximum dataset utility:

$$U_i^D = \sum_{n=1}^{N} U_{i,n}^R \qquad (1)$$

Assuming orthogonal usages, the aggregated dataset utility U^D is the summation of the maximum utility for each usage, across all usages:

$$U^D = \sum_{i=1}^{I} U_i^D = \sum_{i=1}^{I} \sum_{n=1}^{N} U_{i,n}^R \qquad (2)$$

The aggregated utility U^R_n for record *[n]* is the sum of its utilities for each usage, and the aggregated dataset utility can be redefined correspondingly:

$$U_n^R = \sum_{i=1}^{I} U_{i,n}^R \qquad (3)$$

$$U^D = \sum_{i=1}^{I} \sum_{n=1}^{N} U_{i,n}^R = \sum_{n=1}^{N} U_n^R \qquad (4)$$

Utility allocation depends on usage context. A simple allocation, for example, may assign identical utility to each record (i.e., constant $U^R_{i,n} = U^D_i / N$). This "naïve" allocation rarely reflects real-world data usage where dataset records vary in their importance. It is assumed that utility allocation can be mapped from record contents. A context-mapping function u_i that maps attribute contents $(f^E_{n,1}..f^E_{n,M})$ of record *[n]* to the corresponding record utility is defined for each usage context *[i]*:

$$U^R_{i,n} = u_i\left(f^E_{n,1}, \ldots, f^E_{n,M}\right) \quad (5)$$

In some usages, u_i derives the utility from a single attribute (the overall dollar amount of a sale transaction). In many other cases the utility may be determined by multiple attributes – e.g., customer code, sale amounts, and purchase date for analyzing consumption behavior. The mapping function, hence, may have different shapes – monotonically increasing (e.g., higher utility with higher dollar amount), monotonically decreasing (e.g., lower utility with higher distance from a desired location), or non-monotonic (e.g., lower utility with higher absolute difference from the average). Utility mapping may also use a lookup table to map non-ordinal attribute contents to its corresponding utilities (e.g., assigning value to customers based on zip code).

Utility and Size

While utility is a content-based and contextual measure, an equivalent structural and context-independent measure is size *(S)*, defined as the count of data items. In a tabular dataset with each record having an identical set of attributes, the record size is the number of attributes $S^R_n = M$, and the dataset size is the total data item count:

$$S^D = \sum_{n=1}^{N} S^R_n = MN \quad (6)$$

Both utility and size may be used as scaling factors for developing quality measurements. However, as shown, size results in context-independent and structural measurements, while utility leads to equivalent contextual and content-based measurements.

UTILITY-DRIVEN MEASUREMENT

Table 1 summarizes the quality dimensions developed, describing the data characteristics that guide each definition, and comparing impartial and contextual interpretations.

We first develop metrics for data records and datasets, and later extend the definitions to address databases and database collections. The metrics conform to the following consistency principles: *(a) Interpretation Consistency:* Metrics should have a consistent semantic interpretation at all hierarchy levels. For example, "Completeness" must always measure the extent to which missing contents affect quality. *(b) Impartial-Contextual Consistency:* The measurements reflect context, but are derived from context-independent (impartial) characteristics. For context-free assessment, the calculation should fold back to the traditional (impartial) definitions. *(c) Representation Consistency:* The measure should be easy to interpret. The common *[0,1]* representation is adopted, with *1* representing perfection. For contextual assessment, the quality ratio reflects the effect of the extent of utility loss due to quality defects, on utility contribution. Even when the entire potential utility is obtainable (i.e., quality = *1*), defects may still exist without reducing utility. As measurements are defined as utility ratios, they are independent of the unit used to measure utility. *(d) Aggregation Consistency:* Quality measurement of a high level collection aggregates the measurements of each granular component within. The aggregation should result in a *[0,1]* score that cannot be higher than the highest quality level, or lower than the lowest, within the granular items. The weighted average operator guarantees aggregation consistency.

A General Framework for Contextual Quality Measurement

Quality is determined by the presence or absence of quality defects. A data item quality measure $q^E_{n,m}$ reflects the extent to which attribute *[m]* of record *[n]* suffers from a quality defect. It is a value between *0* (severe defects) and *1* (no defects). Equation (6) is now extended assuming that the

Table 1. Impartial versus contextual interpretation of data quality dimensions

Measurement	***Data Characteristics Observed***	***Impartial Interpretation***	***Contextual Interpretation***
Scale	*N* – The number of records *M* – Attributes per record *{f}* – Attribute contents	*S* – size, measured in item count	*U* – utility, the maximum potential value contribution
C – Completeness	The inclusion versus exclusion of data items	The extent to which items are excluded from the dataset	The extent to which excluded data items damage utility
V – Validity	Conformance of data items to corresponding value domain	The extent to which the data items included in the dataset are valid	The extent to which invalid data items damage utility
A - Accuracy	The correctness of data items, compared to a baseline	The extent to which the data items included in the dataset are correct	The extent to which incorrect data items damage utility
T - Currency	The age of data items – the time lag between last update and present time	The extent to which the data items in the dataset are recent	The extent to which outdated data items damage utility
Q – Consolidation	The consolidated effect of the above characteristics	The extent to which the dataset has quality defects	The extent to which quality defects damage utility

actual utility $U^{R*}_{i,n}$ of record *[n]* for usage *[i]* is affected not only by the contents $\{f^{E}_{m,n}\}$, but also by the presence of quality defects $\{q^{E}_{m,n}\}$:

$$U^{R*}_{i,n} = u_i\left(\left\{f^{E}_{n,m}\right\}_{m=1..M}, \left\{q^{E}_{n,m}\right\}_{m=1..M}\right) \qquad (7)$$

The utility obtainable with no quality defects defines the upper-bound on the utility of record *[n]* for usage *[i]*. With defects, the utility might be lower. The quality $Q^{R}_{i,n}$ of record *[n]* for usage context *[i]* is defined as a *[0,1]* ratio between the actual utility obtained $U^{R*}_{i,n}$ and the maximum potential utility $U^{R}_{i,n}$:

$$Q^{R}_{i,n} = U^{R*}_{i,n} \Big/ U^{R}_{i,n}$$
$$= \left(u_i\left(\left\{f^{E}_{n,m}\right\}_{m=1..M}, \left\{q^{E}_{n,m}\right\}_{m=1..M}\right)\right)$$
$$/\left(u_i\left(\left\{f^{E}_{n,m}\right\}_{m=1..M}, \left\{q^{E}_{n,m}=1\right\}_{m=1..M}\right)\right) \qquad (8)$$

Similarly, the aggregated record quality Q^{R}_{n} across all usages is the ratio between the actual utility and the maximum possible utility, both summarized across all usages. This is equivalent to a weighted average, where the non-negative weights $\{U^{R}_{i,n}/U^{R}_{n}\}$ sum up to *1*:

$$Q^{R}_{n} = \left(\sum_{i=1}^{I} U^{R*}_{i,n}\right) \Big/ \left(\sum_{n=1}^{I} U^{R}_{i,n}\right)$$
$$= \left(\sum_{i=1}^{I} U^{R}_{i,n} Q^{R}_{i,n}\right) \Big/ \left(\sum_{n=1}^{I} U^{R}_{i,n}\right) = \sum_{i=1}^{I}\left(U^{R}_{i,n} \Big/ U^{R}_{n}\right) Q^{R}_{i,n} \qquad (9)$$

The dataset quality (Q^{D}_{i}) for usage *[i]* is the ratio between the actual utility (given possible utility reductions due to defects) and the potential maximum utility:

$$Q_i^D = \left(\sum_{n=1}^{N} U_{i,n}^R Q_n^R\right) \Big/ \left(\sum_{n=1}^{N} U_{i,n}^R\right)$$

$$= \sum_{n=1}^{N} \left(U_{i,n}^R / U_i^D\right) Q_n^R \qquad (10)$$

The weight factors sum to *1* and, hence, the dataset quality $Q^D{}_i$ is always within *[0,1].* Similarly, the aggregated dataset quality Q^D across all usages is the ratio between the actual utility obtained and the maximum obtainable utility across all usages. This is equivalent to a utility-driven weighted average, where the weights $\{U^D{}_i/U^D\}$ are non-negative and sum to *1*:

$$Q^D = \sum_{i=1}^{I} U_i^D Q_i^D \Big/ \left(\sum_{n=1}^{I} U_i^D\right) = \sum_{i=1}^{I} \left(U_i^D / U^D\right) Q_i^D \qquad (11)$$

For a *context-independent* quality assessment, the size is used instead of utility. With size *S* defined as the item count, the record size is $S^R=M$. The record's context-independent utility is the average data item quality. Accordingly, the impartial dataset quality is:

$$Q_n^R = \frac{1}{M} \sum_{m=1}^{M} q_{n,m}^E \qquad (12)$$

$$Q^D = \left(\sum_{n=1}^{N} M Q_n^R\right) \Big/ \left(\sum_{n=1}^{N} M\right)$$

$$= \frac{1}{N} \sum_{n=1}^{N} Q_n^R = \frac{1}{MN} \sum_{n=1}^{N} \sum_{m=1}^{M} q_{n,m}^E \qquad (13)$$

With a binary measure (1–perfect, 0-imperfect), the results are equivalent to a ratio between the counts of perfect-items and total items and is consistent with structural measurement definitions.

Illustrative Example: Impartial vs. Contextual Quality Measurement

Consider the data sample in table 2. This dataset has customer data that firms typically collect.

The customer-life-time-value is commonly used to assess the potential purchase power of a customer. In this example, it reflects the utility of each customer record. Given that records (*B* and *E*) suffer quality defects and cannot be used, what is the quality of this dataset?

With structural measurement, with *2* out of *5* records bad, the ratio of good-quality records is *60%*. However, considering utility loss – these two records account for *1500* out of *2000* utility units. Hence, the corresponding quality is *25%*. Thus, content-based and contextual measurement can yield substantially different assessments of quality.

Extending General Definitions to Specific Quality Dimensions

New definitions for completeness, validity, accuracy, and currency, each reflecting a different type of quality defect, can be developed. The

Table 2. Illustrative example – Client data

ID	*Name*	*Address*	*Marital Status*	*Number of Children*	*…Other Attributes…*	*Customer Lifetime Value*
A	***John***	*…*	*…*	*…*	*…*	***100***
B	***Jennifer***	*…*	*…*	*…*	*…*	***800***
C	***William***	*…*	*…*	*…*	*…*	***200***
D	***Nancy***	*…*	*…*	*…*	*…*	***200***
E	***Gloria***	*…*	*…*	*…*	*…*	***700***

framework also permits consolidating quality measurements across dimensions by combining the effects of quality defects. Once quality measurements are defined for the data item, the definitions at higher levels of the hierarchy follow similar calculation schemes.

Completeness (C) measures the extent to which all anticipated data units in the collection are indeed included (Ballou and Pazer, 2003). The data item completeness $c^E_{n,m}$ is a binary indicator that is *1* if attribute *[m]* of record *[n]* is included in the database and available for use (i.e., retrievable and uncorrupted), and *0* otherwise. To reflect the extent to which the exclusion of a data item affects record utility, the utility mapping function (7) is redefined by replacing $q^E_{n,m}$ with $c^E_{n,m}$. The resulting completeness measurement (Eqs. *8-11*) reflects reduction of utility due to the exclusion of data items.

Validity (V) (a.k.a. integrity or domain consistency) measures the extent to which data items conform to their corresponding value domains.. Data item validity $v^E_{n,m}$ is a binary indicator that is *1* if attribute *[m]* of record *[n]* conforms to its value domain and *0* if not. The utility mapping function (7), redefined by replacing $q^E_{n,m}$ with $v^E_{n,m}$, reflects the extent to which invalid attributes affect utility of the data record. The resulting validity measurements (Eq. *8-11*) reflect reduction of utility due to invalid data items.

Accuracy (A) measures the extent to which data items conflict with a baseline that is perceived to be correct. Data item accuracy $a^E_{n,m}$ is the extent to which $f^E_{m,n}$, the content of attribute *[m]* in record *[n]*, is different from a correct baseline value $f^{E*}_{m,n}$ – a measure between *0* (very different) and 1 (perfectly identical). The binary measure is a possible approach - *1* if the content is perfectly accurate and *0* if not. A more refined approach uses a distance measure that assigns *1* if $f^E_{n,m}=f^{E*}_{n,m}$ and approaches *0* as the error margin increases. With these definitions, the utility mapping function (7) reflects the extent to which inaccurate contents affect utility of data records (by replacing $q^E_{n,m}$ with $a^E_{n,m}$). The resulting measurements (Eq. *8-11*) reflect utility reduction due to inaccuracies.

Currency (T) measures the degree to which the data is recent and up to date. Currency reflects the age $g^E_{n,m}$, the time lag between present time and the last update of the data item. To be consistent with the other measurements, data item currency, $t^E_{n,m}$, is defined by rescaling the age to a *[0,1]* range. A possible rescaling is, for example, an exponentially declining currency: $t^E_{n,m}=exp\{-\eta g^E_{n,m}\}$, where $\eta>0$ is an exponential decline factor. Applying this, the utility mapping function (7) now reflects the extent to which outdated attribute contents affect utility (by replacing $q^E_{n,m}$ with $t^E_{n,m}$). The resulting currency measurements (Eq. *8-11*) reflect reduction of utility due to outdated data items.

Consolidated Quality (Q) reflects the overall utility reduction as a result of the different quality defects. Using earlier definitions, the consolidated data item quality $q^E_{m,n}$ is defined as a product of the individual effects: $q^E_{m,n}=(c^E_{n,m})^\alpha(v^E_{n,m})^\beta(a^E_{n,m})^\gamma(t^E_{n,m})^\delta$, where $\alpha, \beta, \gamma,$ and δ are non-negative sensitivity factors. With this formulation, the consolidated data item quality is perfect $(q^E_{m,n}=1)$ if no quality defect is present (i.e., $c^E_{m,n}= v^E_{m,n}= a^E_{m,n}= t^E_{m,n}=1$). It degrades in the presence of quality defects, possibly to *0* if one or more of the components have a *0* value. The higher the sensitivity factor associated with a quality component (completeness, validity, accuracy, or currency), the steeper the overall quality decline with the decrease in the quality of that component. A consolidated measure can contribute to a rapid assessment of data quality with minimal cognitive effort by offering an overall view of quality. If the consolidated quality is nearly perfect (i.e., $Q^D \rightarrow 1$), data may be perceived as acceptable, and investigations of specific dimensions may be superfluous.

Data Quality Metrics at Higher-Levels of the Data Hierarchy

The contextual quality measurements can be extended to higher levels of the data hierarchy – databases and the database collections. Quality measurements at higher hierarchical levels can serve as managerial tools to assess the overall quality of the data resources. The methodology for the consolidated quality measurement *Q*, discussed in the previous section, is further developed here. A similar methodology can be applied to derive any specific quality dimension that is defined at the dataset level.

Consider a single database, with *J* datasets indexed by *[j]*. Each dataset may by used in *I(j)* different usage contexts (indexed by *[j ,i]*), and has an overall dataset utility of U^D_j. The maximum database utility U^B is the sum of maximum possible dataset utilities:

$$U^B = \sum_{j=1}^{J} U_j^D = \sum_{j=1}^{J} \sum_{i=1}^{I(j)} U_{j,i}^D \qquad (14)$$

The database utility U^{B*} is the sum of the actual dataset utilities, which are affected by the quality levels associated with each:

$$U^{B*} = \sum_{j=1}^{J} U_j^D Q_j^D \qquad (15)$$

The database quality Q^B is defined as the ratio between the actual utility obtained and the maximum utility obtainable.

$$Q^B = U^{B*} / U^B$$
$$= \sum_{j=1}^{J} U_j^D Q_j^D \Bigg/ \left(\sum_{j=1}^{J} U_j^D \right) = \sum_{j=1}^{J} \left(U_j^D / U^B \right) Q_j^D \qquad (16)$$

An equivalent impartial definition will rescale the database quality measure using size. The size of dataset *[j]* is defined as the number of data items in it $S^D_j = M_j N_j$ (M_j being the number of attributes and N_j the number of records in this dataset). The overall database size is the total size of the included datasets:

$$S^B = \sum_{j=1}^{J} S_j^D = \sum_{j=1}^{J} M_j N_j \qquad (17)$$

The impartial database quality Q^B is defined as a size-driven weighted average of dataset quality:

$$Q^B = \sum_{j=1}^{J} M_j N_j Q_j^D \Bigg/ \left(\sum_{j=1}^{J} M_j N_j \right)$$
$$= \sum_{j=1}^{J} \left(S_j^D / S^B \right) Q_j^D \qquad (18)$$

Consider a database collection with *K* databases (indexed by *[k]*), each with a maximum utility of U^B_k. The maximum utility of the database collection U^L is the sum of database utilities:

$$U^L = \sum_{k=1}^{K} U_k^B \qquad (19)$$

The actual utility of the database collection U^{L*} is the sum of the actual database utilities, which are affected by the quality levels of each associated database:

$$U^{L*} = \sum_{k=1}^{K} U_k^B Q_k^B \qquad (20)$$

The quality of the database collection Q^L is the ratio between the actual utility obtained and the maximum utility obtainable from it:

$$Q^L = U^{L*} / U^L = \sum_{k=1}^{K} U_k^B Q_k^B \Bigg/ \left(\sum_{k=1}^{K} U_k^B \right)$$
$$= \sum_{k=1}^{K} \left(U_k^B / U^L \right) Q_k^B \qquad (21)$$

An impartial equivalent to this measurement can be a size driven weighted average:

$$Q^L = \sum_{k=1}^{K} S_k^B Q_k^B \Bigg/ \left(\sum_{j=1}^{J} S_k^B \right) = \sum_{j=1}^{J} \left(S_k^B / S^L \right) Q_k^B \qquad (22)$$

CONCLUSION

Measurements using impartial and context-independent characteristics alone are of limited use to the data consumer as these do not permit gauging data quality within the context in which the data is used. A key contribution of this chapter is the development of the framework and metrics that support contextual assessment of data quality. The metrics link context-independent characteristics (e.g., data contents, exclusion/exclusion of items, or record age) with the context-dependent utility of the data. Contextual quality is conceptualized as the extent to which quality defects affect the utility of data within a particular usage context. Accordingly, the metrics, which use the utility mapping of impartial data characteristics as the baseline, reflect the extent to which utility is reduced. The utility-driven assessment is demonstrated by developing new definitions for completeness, validity, accuracy and currency. A similar methodology can help define measures for other quality dimensions and consolidate multiple measures to derive a single quality measure. In this study, we assume utility can be assessed and allocated and that usages and utilities are independent. Future work can reevaluate these assumptions, to account for interdependencies among utilities and usages.

Contextual assessment of data quality has important implications for business users. The methodology proposed grants the flexibility for adapting quality metrics to the specific contextual needs – e.g., by identifying important data attributes and selecting the appropriate functional form for mapping their contents onto utility. Consequently, business users working in a specific task/context can get numeric measurements that are closely related to the content that matters most for that context. Utility-driven computations can be integrated into databases, data management systems, and software for supporting data quality management. Such calculations can be treated as quality metadata and stored as part of the organizational metadata repository, or included as additional attributes in the dataset. While content-based calculations may produce numeric results that are different from structure-based calculations, they do not necessarily require different presentation styles. The outcome, a number within the range of *[0,1]*, adheres to common formats for representing quality and can be presented using previously developed end-user tools.

REFERENCES

Ahituv, N. A. (1980). Systematic approach towards assessing the value of information system. *MIS Quarterly, 4*(4), 61-75

Ballou, D. P., & Pazer, H. L. (1995). Designing information systems to optimize the accuracy-timeliness tradeoff. *Information Systems Research, 6*(1), 51-72

Ballou, D. P., & Pazer, H. L. (2003). Modeling completeness versus consistency tradeoffs in information decision systems. *IEEE Trans. on Knowledge and Data Eng., 15*(1), 240-243

Ballou, D. P., Wang, R., Pazer, H., & Tayi, G. K. (1998). Modeling Information Manufacturing Systems to Determine Information Product Quality. *Management Science, 44*(4), 462-484

Eckerson, W. W. (2002). *Data quality and the bottom line: Achieving business success through a commitment to high quality data. The Data Warehousing Institute Report Series, 101.* Chatsworth.

Kulikowski, J. L. (1972). Notes on the value of information transmitted through a communication channel. *Proceedings of the 2nd International Symposium on Information Theory (1971).* Cachkadsor: ASSR

Lee, W. Y., & Strong, D. M. (2003). Knowing-why about data processes and data quality. *Journal of Management Information Systems, 20*(3), 13-39.

Pipino, L. L., Lee, Y. W., & Wang, R. Y. (2002). Data quality assessment. *Communications of the ACM, 45*(4), 211-218.

Redman, T. C. (1996). Data quality for the information age. Boston, MA: Artech House.

Shankaranarayanan, G., Ziad, M., & Wang, R. Y. (2003). Managing data quality in dynamic decision making environments: An information product approach. *Journal of Database Management, 14*(4), 14-32.

Shankaranarayanan, G., & Even, A. (2004). Managing metadata in data warehouses: Pitfalls and possibilities. *Communications of the AIS, 14*, 247-274.

Shankaranarayanan, G., & Cai, J. (2006). Supporting data quality management in decision making. *Decision Support Systems, 42*, 302-317.

Strong, D. M., Lee, W. Y., & Wang, R. Y. (1997). Data quality in context. *Communications of the ACM, 40*(5), 103-110.

Wang, R. Y. & Strong, D. M. (1996). Beyond accuracy: What data quality means to data consumers. *Journal of Management Information Systems, 12*(4), 5-34.

Wang, R. Y. (1998). A product perspective on total quality management. *Communications of the ACM*, 41(2), 58-65.

KEY TERMS

Accuracy: A data quality dimension that reflects the confirmation of data items to a baseline that is perceived to be correct, and the extent to which conflicts with the correct baseline affects fitness to use. A baseline could be, for example, the real-world value that a data item reflects, a value in another dataset that was reliably validated, or a targeted calculation result.

Completeness: A data quality dimension that reflects the inclusion of all the anticipated data and the extent to which exclusion of certain items affects fitness to use.

Content-Based Data Quality Assessment: Perception and measurement of data quality which accounts for content - the actual values stored. Content-based assessment typically links content to a specific usage and does not assumes an absolute and objective quality standard.

Contextual Data Quality Assessment: Perception and measurement of data quality, which reflects its fitness for use within a specific usage context. Contextual assessment may be affected by usage characteristics such as the task, the organizational domain, the timing of usage, and/or the expertise of the individual user.

Currency: A data quality dimension that reflects the degree to which all data items are recent and up to date, and the extent to which non-recency of data items affects fitness to use. A base line could be, for example, the real-world value that a data item reflects, a value in another dataset that was reliably validated, or a targeted calculation result.

Data Utility: A measure of the business value attributed to data within specific usage contexts. Utility is typically, but not necessarily, measured in monetary units.

Impartial Data Quality Assessment: Perception and measurement of data quality, which is based on the data itself, regardless of how that data is used.

Structure-Based (or Structural) Data Quality Assessment: Perception and measurement of data quality which is driven by physical characteristics of the data such as item counts, time tags, or failure rates. Structure-based assessment typically assumes an absolute and objective quality standard.

Validity: A data quality dimension that reflects the confirmation of data items to their corresponding value domains, and the extent to which non-confirmation of certain items affects fitness to use. For example, a data item is invalid if it is defined to be integer but contains a non-integer value, linked to a finite set of possible values but contains a value not included in this set, or contains a NULL value where a NULL is not allowed.

Chapter XLIII
Geometric Quality in Geographic Information

José Francisco Zelasco
Universidad de Buenos Aires, Argentina

Gaspar Porta
Washburn University, USA

José Luís Fernandez Ausinaga
Universidad de Buenos Aires, Argentina

INTRODUCTION

Both this article, referred to as Article I, and another one, Article II, entitled "**Geometric Quality in Geographic Information IFSAR DEM Control**", published in this encyclopedia propose a method to evaluate the geometric integrity of Digital Elevation Models obtained by different techniques. Therefore, the theoretical aspect of the method to evaluate the geometric integrity and different stochastic hypotheses will be presented in both of them. Herein, we consider the classical topographic or aerial photogrammetry stereo images method (included ASTER or SPOT images) and we assume consistent stochastic hypotheses. In the Article II we consider Interferometry SAR (IFSAR) techniques and the stochastic hypotheses are specific according to the particular geometry involved in this technique.

A typical way to build surface numerical models or Digital Elevation Models (DEM) for Geographical Information Systems (GIS) is by processing the stereo images obtained from, for example, aerial photography or SPOT satellite data. These GIS can perform many computations involving their geographic databases. The quality control of a geographic database and in particular the topological and geometric integrity are, therefore, important topics (Guptill & Morrison, 1995; Harvey, 1997; Ubeda & Servigne, 1996; Laurini & Milleret-Raffort, 1993). The geometric quality control of the stored DEM is what we are concerned with here. «Quality» means the geometric accuracy measured in terms of the difference between a DEM and a reference DEM (R-DEM). We assume the R-DEM is a faithful model of the actual surface. Its point density may be greater than the DEM point density.

BACKGROUND

In the literature, several unsatisfactory solutions were proposed for the DEM control with respect to a reference. A critical problem in the error estimation (evaluated using the difference referred to in the previous paragraph) is to establish for each selected point of the DEM the corresponding homologous point in the R-DEM. Other kinds of problems and errors are related to the existence of aberrant points, systematization, etc. These problems were studied for horizontal errors in maps in (Abbas, 1994; Grussenmeyer, 1994; Hottier, 1996a). These authors found that the dissymmetry model-reference was the most important factor to determine homologous pairs.

Several solutions have been proposed for the 'punctual control method' (recognition algorithms, filtering methods, adjustment of histograms to theoretical laws) without obtaining completely satisfactory results (Dunn et al, 1990; Lester & Chrisman, 1991). Later, (Abbas, 1994; Grussenmeyer, 1994; Hottier, 1996a) present an alternative to the punctual control method: the 'linear control method' based on the dissymmetry of the Hausdorff distance.

Habib (1997) analyzes precision and accuracy in altimetry and mentions some of the proposals of the last decade for the elevation control of quality.

In the case of the DEM's, to assess the difference that gives rise to the error we wish to compute, we need to identify without ambiguity each point *M* in the DEM with its homologous point *P* in the R-DEM.

Two reasons make this task difficult:

1. Many points *M* are not identifiable, those situated on regular sides, which are indistinguishable from their neighbors. Potentially identifiable points are those located on sharp slope variations, and possibly those with zero slope (tops, bottoms and passes).
2. A point identifiable on the DEM is not necessarily identifiable in the R-DEM, because of a difference in scale ("generalization") or aberrant errors.

To find identifiable homologous pairs of points is difficult for an operator. Automating this is a very delicate process.

In the case of the geometric assessment of a DEM from an ASTER (Advanced Spaceborned Thermal Emission and Reflection Radiometer) (Hirano et al., 2002) profiles or benchmarks (particular points) are used for the elevation accuracy assessment. This elevation accuracy assessment refers to the vertical error of the DEM. However, using profiles the horizontal error is not taken into account and also there is a model-reference discrepancy. Moreover, the choice and number of the benchmarks might not be a good stochastic sample of points. On the other hand to estimate the elevation error (σ_z), by means of the commonly used (vertical distance) estimator, gives poor results with an irregular surface because it introduces a systematic error (Zelasco, 2002).

In light of the above difficulties, in (Zelasco & Ennis, 2000) an estimator for the variances of the vertical and horizontal errors were proposed. The values obtained for the estimator, according to the type of terrain, its unevenness, and the number of sample points in the DEM, were studied using simulations in (Zelasco et al., 2001; Zelasco, 2002). Throughout these studies, it is assumed that the errors in elevation and in the horizontal plane are independent Normal random variables - in agreement with Liapounov's theory (Hottier, 1996b). This proposed method is called the Perpendicular Distance Estimation Method (PDEM).

Let $e(M_k)=M_k - P_k$ where M_k, P_k are the k selected homologous pairs of points in the DEM and R-DEM respectively. Estimating $\sigma^2(e)$ (variance of the error as distance between *M* and *P*) without measuring each vector $e(M_k)$ completely, is the key advantage of the proposed PDEM. Only the projection of each $e(M_k)$ in particular directions

matters. Given an M_k it becomes unnecessary to find the homologous P_k in the R-DEM. How this is done is the subject of the following section.

For three-dimensional DEM's built with the usual techniques, such as photogrammetry or remote sensing, there are expected normal deviations in altimetry (σ_z) and in the horizontal plane (σ_p) which are 'a priori' known. They are generally different from each other due to the manner in which the values of the *z* coordinate (altimetry) and the *x, y* coordinates (the horizontal plane) are evaluated. The goal of this PDEM is to evaluate the actual errors ('a posteriori') of a given DEM.

THE QUALITY EVALUATION METHOD

Description of the PDEM

The gap between the DEM and the R-DEM is represented by a function that links each point of the DEM to the vector $e(M)=M-P$ in R^3 where *P* is the homologous point to *M* in the R-DEM. *M* and *P* represent the same relief element, however *P* is not vertically aligned with *M*.

The DEM accuracy is a function of the vectors $e(Mk)$ of a sample of $\{Mk\}$ points. This function is used in the construction of the covariance matrix $\sigma^2(e)$.

If the errors in the three directions are different (for instance in interferometry radar see the article II) the rank of $\sigma^2(e)$ is three. Its eigenvalues are the variances in the three main coordinates.

When the errors in the orthogonal directions in the plane are equal (hypothesis assumed in the case of stereo images techniques), we can consider that the rank of the covariance matrix $\sigma^2(e)$ is two. The eigenvalues are the variances σ_z^2, and $\sigma_x^2 = \sigma_y^2 = \sigma_{x,y}^2$.

As it can be seen the hypotheses assumed determine the way the method has to be used.

If the hypotheses are that the errors follows the normal law and are independent in the direction of the three main axes we have to use a rank three covariance matrix, if the hypotheses of the error in the horizontal plane are equal in all directions the rank covariance matrix can be reduced to two. In the article II we will analyze the case where the error is correlated in two directions.

Assuming the hypotheses that the error follows the normal law and are independent in the three main axes, in a rotated frame of reference, the new covariance matrix will no longer be diagonal. The new components are expressed in terms of the eigenvalues of σ^2. The following expression corresponds to the first component $\sigma_{x'}^2$ of the new covariance matrix:

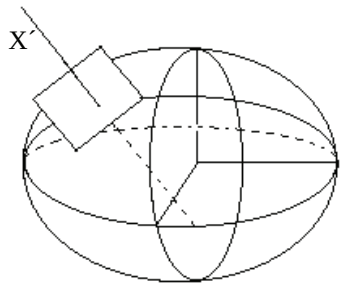
$$\sigma_{x'}^2 = a_{11}^2\sigma_x^2 + a_{12}^2\sigma_y^2 + a_{13}^2\sigma_z^2 \qquad (1)$$

where a_{1j} are the managing cosines of the unit vector in the x' coordinate of the rotated reference frame.

The value of the variance in the x' direction is equal to the square of the distance between the two planes normal to the x' direction: one tangent to the "distribution indicative ellipsoid" (whose parameters are $a=\sigma_x$; $b=\sigma_y$; $c=\sigma_z$) and the other passing through its center.

In Figure 1, the plane normal to the *x*' direction and tangent to the distribution indicative ellipsoid is drawn in the general case:

Figure 1. Distribution indicative ellipsoid and tangent plane Π



The algebraic form of equation [1] is used to obtain the variance in any direction ***d***, replacing the coefficients a_{ij} with the managing cosines ($cos\ \alpha$, $cos\ \beta$, $cos\ \gamma$) corresponding to this direction ***d***:

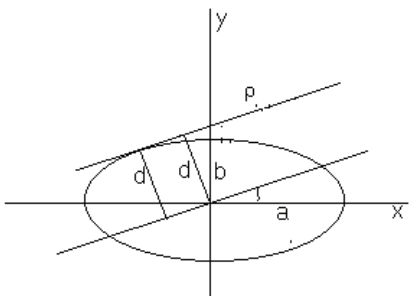
$$\sigma_d^2=\sigma_x^2 cos^2\alpha+\sigma_y^2 cos^2\beta+\sigma_z^2 cos^2\gamma \qquad (2)$$

σ_d^2 can be estimated with $\Sigma e_i^2/n$ $0<i<I$ where the e_i are the error components in the ***d*** direction.

We construct a mesh of triangles from the points of the R-DEM. We call this a model of the surface. Given a point *M* in the DEM, its projection onto the *x,y* plane is inside the projection of a unique triangle *T* from this model. We call this the corresponding triangle *T* to *M*. The e_i component in the direction $\boldsymbol{d}_i$ is the distance d_i, from each point to the plane of the corresponding triangle. The squares of the distances d_i, between each point of the model and the plane of the corresponding triangle, allow us to establish a least squares estimator for the assessment of (σ_x^2, σ_y^2, σ_z^2).

Recall that if the errors in the horizontal plane are independent of the direction the problem is reduced from three to two unknowns (figure 2). $\sigma_x=\sigma_y$ and the distribution indicative ellipsoid is a revolution ellipsoid. The maximum slope direction in each R-DEM triangle (equal to the angle made by the normal to the triangle and the *z* coordinate) will determine the managing cosines direction in two-dimensions. The position of the homologous point in the R-DEM does not need to be known. This is the strength of the PDEM. By having determined the corresponding triangle, we capture the required information without needing to find the particular homologous point of the R-DEM. Therefore it is only needed to determine the distance $\boldsymbol{d}_i$, which is the error component in the direction normal to the triangle.

Figure 2. Distribution indicative ellipse and tangent straight line ρ



From [2] and with:

$$\sigma_x^2=\sigma_y^2=\sigma_{x,y}^2 \qquad (3)$$

we get:

$$\sigma_d^2=\sigma_{x,y}^2(cos^2\alpha+cos^2\beta)+\sigma_z^2 cos^2\gamma \qquad (4)$$

and finally:

$$\sigma_d^2=\sigma_{x,y}^2 sin^2\gamma+\sigma_z^2 cos^2\gamma \qquad (5)$$

If we assume that we have only one value of the distance $\boldsymbol{d}_i$ for each direction $\boldsymbol{d}_i$, we may consider that $\boldsymbol{d}_i^2$ is an estimator of σ_d^2, so we get the solution with the i relations:

$$d_i^2=\sigma_{x,y}^2 sin^2\gamma+\sigma_z^2 cos^2\gamma$$

which is more precise than using σ_d^2 estimated by means $\Sigma e_i^2/n$ $0<i<I$. Additionally this avoids having to give a different weight to each equation.

How to Employ the PDEM

Recall that a R-DEM will be used as the control for a DEM, real or simulated. Also recall that a R-DEM, being a discrete set, allows constructing a mesh of *K* triangles defined by the points on the reference surface. Each triangle T_k, *(k=1,..,K)*, belongs to a plane which has a normal unitary vector n_k, *(k=1,..,K)*.

The mesh of *K* triangles of the R-DEM is constructed using, for instance, the Delaunay method in two dimensions.

In the case of a simulation we have to:

- Determine the mass center of each triangle.
- Adopt a standard deviation for each coordinate σ_x, σ_y and σ_z and add three random normal variables to the center mass coordinates to simulate the points of the DEM to be evaluated. With $\sigma_x=\sigma_y$ we evaluate $\sigma_{x,y}$ (horizontal plane standard deviation). In this case the distribution indicative ellipsoid is a revolution surface.

In the case of a real DEM we have to:

- Determine the triangle of the R-DEM containing the homologous point of each point of the DEM. Recall that the procedure is as follows: given a DEM point M_s, its homologous point P in the R-DEM will be in a plane define by the triangle T_s, if the projection of the point M_s on the (x, y) plane is inside the projection of the triangle T_s on the same plane. We establish a practical limit to avoid border problems given by the 'a priori' standard deviation σ_p (in the horizontal plane) which allows us to select the points of the DEM in the sample. If the horizontal distance between the projection of M_s and the limits of the projected triangle is at least three times greater than σ_p, the point is accepted. Otherwise, it is rejected. This is because σ_p is the standard deviation of a normal random variable.
- In both cases we have to:
- Calculate the distance from each point (point of the simulated or of the real DEM) to the plane of the corresponding triangle.
- Evaluate, using the PDEM, the standard deviation of the DEM in altimetry (σ_z) and in the horizontal plane ($\sigma_{x,y}$). In the case of the simulation their values should be similar to the ones already used.

Note 1: Knowledge of the γ (in [5]) angle makes it possible to select a R-DEM with more or less unevenness.

Note 2: For each one of the planes having the direction d given by the normal unitary vector n_k in question, the distance from the point M_j, *(j=1.. J<=I)* to the corresponding plane is a measure (in the direction ***d***) of the error e_d.

DISCUSSION

Evaluating the horizontal error is the most delicate task in the geometric quality control of a Geographical Database.

When the unevenness of the R-DEM surface is insufficient, there is not enough information to evaluate the horizontal error. The assessment of the horizontal error will not be reasonably precise. With enough unevenness, the horizontal error can be comfortably estimated. The elevation error estimation is better than the horizontal one. The quality of the horizontal error assessment will depend on the unevenness of the reference surface as well as on the dimension of the sample (Zelasco, 2002).

With simulations, the error precision values can be calculated, but with a real DEM, the error precision is unknown (it can only be estimated). Applying the PDEM, two values are obtained: the elevation error and the horizontal error of a DEM. With these estimated values (in elevation and in the horizontal plane) simulations may be performed to evaluate the standard deviation of the estimated errors. These standard deviations determine whether to keep or to reject the estimated errors - particularly in the case of the horizontal one.

FUTURE TRENDS

Quite interesting would be to consider the effects of different error values for the *x* and the *y* coordinates. In the article II we study how to modify the proposed method taking into account the correlation between the error in elevation and the error in a particular direction of the horizontal plane. So the PDEM is extended to the case of the Interferometry SAR.

CONCLUSION

Estimating the elevation error (σ_z), by means of the commonly used vertical distance estimator is affected by the bias introduced by slanted surfaces. Using the PDEM the elevation error evaluation will always be very good. The horizontal error evaluation $(\sigma_{x,y})$ will be satisfactory when the roughness and the number of points of the sample are suitable.

REFERENCES

Abbas, I. (1994). *Base de données vectorielles et erreur cartographique: problèmes posés par le contrôle ponctuel; une méthode alternative fondée sur la distance de Hausdorff*: le contrôle linéaire. Thèse, Université Paris 7.

Abbas, I., Grussenmeyer, P., & Hottier, P. (1994). *Base de Données géolocalisées: estimation de l'erreur cartographique: une méthode fondé sur la distance Hausdorf: le contrôle linéaire.* Remarques sur le contrôle ponctuel. Notes de cours, ENSG IGN, France.

Dunn, R., Harrison, A. R., & White, J. C. (1990). Positional accuracy and measurement error in digital databases of land use: an empirical study. *International Journal of Geographic Information Systems, 4*(4), 385- 398.

Grussenmeyer, P., Hottier, P., & Abbas, I. (1994). Le contrôle topographique d'une carte ou d'une base de données constituées par voie photogrammétrique. *Journal XYZ 59,* Association Française de Topographie, (pp. 39-45).

Guptill, S. C., & Morrison, J. L. (Eds.) (1995) Elements of spatial data quality. Oxford: Elsevier.

Habib, M. (1997). *Etude, par simulation, de la précision altimétrique et planimétrique d'un MNT obtenu par corrélation d'images spatiales (capteur à balayage) - Précision d'un MNT régulier -Erreur de rendu d'un semis régulier* - Description d'une structure de MNT régulier. Thèse, Univerisité Paris 7.

Harvey, F. (1997). Quality Needs More Than Standards. M. Goodchild & R. Jeansoulin (Eds.), *Data Quality in Geographic Information: From Error to Uncertainty* (pp. 37-42). Paris: Hermès..

Hirano, A., Welch, R., & Lang, H. (2003). Mapping from ASTER stereo image data: DEM validation and accuracy assessment. *ISPRS Journal of Photogrammetry and Remote Sensing,* 57, 356-370.

Hottier, P. (1996). Qualité géométrique de la planimétrie. Contrôle ponctuel et contrôle linéaire. Dossier: La notion de précision dans le GIS. *Journal Géomètre 6*, juin, Ordre des Géomètres-Experts Français, (pp. 34-42).

Hottier, P. (l996). *La méthode du contrôle linéaire en planimétrie - Propositions pour une typologie des détails anormaux*. Rapport de recherche, ENSG, IGN, France.

Hottier, P. (1996). *Précis de statistiques.* ENSG, IGN, France.

Lester, M., & Chrisman, N. (1991). Not all slivers are skinny: a comparison of two methods for detecting positional error in categorical maps. *GIS/LIS, 2,* 648-656.

Laurini, R., & Milleret-Raffort, F. (1993). *Les bases de données en géomatique*. Editions HERMES, Paris.

Ubeda, T., & Servigne, S. (1996). Geometric and Topological Consistency of Spatial Data. *Proceedings of the 1st International Conference on Geocomputation*, (pp. 830-842, vol. 1). Leeds, United Kingdom, September 17-19.

Zelasco, J. F., & Ennis, K. (2000). *Solución de un problema en la estimación de variancias en un caso especial en el que no se pueden aplicar estimadores habituales.* XXVIII Coloquio Argentino de estadística, Posadas, Misiones, Argentina.

Zelasco, J. F., Ennis, K., & Blangino, E. (2001). Quality Control in Altimetry and Planimetry for 3D Digital Models. *Procedures 8th European Congress for Stereology and Image Analysis.* Bordeaux, France.

Zelasco, J. F. (2002). Contrôle de qualité des modèles numériques des bases de données géographiques. *Journal XYZ 90,* Association Française de Topographie, 50-55.

KEY TERMS

Accuracy: The degree of conformity of a measured or calculated quantity to its actual (true) value. Accuracy is the degree of veracity.

Covariance Matrix: the square n x n of which the entries are the pair wise correlations of the variables of a random vector of length n; the (i, j) th entry is the correlation between the ith and the jth variables.

DEM Geometric Quality: The geometric precision measured in terms of the difference between a digital elevation model (DEM) and a reference DEM (R-DEM).

Digital Elevation Model (DEM): Set of points in a 3 dimensional coordinate system modeling a real object surface.

Geographic Information System (GIS): Tool for processing localized information. A GIS will model and locate the spatial data of a real phenomenon.

Homologous Points: The point in the DEM and the point in the R- DEM modeling the same point in the real surface. Their distance is the error of the DEM point, assuming the points in the R-DEM are without error.

Photogrammetry: Technique permitting the set up of the coordinates of points (DEM) of an object surface by processing stereo images of the object surface.

Precision: The degree of reproducibility or repeatability. The measure to which, further measurements or calculations show the same or similar results. In many cases precision can be characterized in terms of the standard deviation of the measurements, sometimes incorrectly called the measurement process's standard error. The results of calculations or a measurement can be accurate but not precise; precise but not accurate; neither; or both. A result is called valid if it is both accurate and precise.

Revolution Ellipsoid: The 3 dimensional solid obtained from revolving an ellipse about one of its axes.

Spatial Data: information related to a real system involving geometric (dimension, shape) and topological (relative position) aspects of it.

Stereo Images: two or more images of a real object's surface obtained from different points of view.

Variance (Estimation): The average of the sum of the squares of the differences of the values attained by a random variable from it's mean.

Chapter XLIV
Geometric Quality in Geographic Information IFSAR DEM Control

José Francisco Zelasco
Universidad de Buenos Aires, Argentina

Judith Donayo
Universidad de Buenos Aires, Argentina

Kevin Ennis
Universidad de Buenos Aires, Argentina

José Luís Fernandez Ausinaga
Universidad de Buenos Aires, Argentina

INTRODUCTION

Both this, article II and another one, article I, titled "Geometric Quality In Geographic Information" published in this encyclopedia propose the theoretical aspects of the method to evaluate the geometric integrity of Digital Elevation Models obtained by different techniques and therefore, different stochastic hypotheses are considered. In the article I we considerer the classical topographic or aerial photogrammetry stereo images method (included ASTER or SPOT images) and we assume consistent stochastic hypotheses. In this Article II we consider IFSAR techniques and the stochastic hypotheses are specific according to the particular geometry involved in this technique.

As it was established in the article I, studies performed on Digital Elevation Models (DEM), obtained by means of photogrammetry, SPOT satellite images or other methods show that the precision in the z coordinate is different from the horizontal precision. In the case of the Interferometry SAR (IFSAR), the precision in the azimuth axis may be different from the precision in the range-axis. Moreover, the error in elevation is correlated with the error in the range axis. The method employed in article I allows the evaluation of the DEM accuracy –vertical and horizontal- under

some conditions of topographic unevenness. A reference DEM is required. In recent (Zelasco et al, 2001; Zelasco, 2002a) works it has been shown, by simulation, that the vertical error estimation is good and the horizontal error estimation is reasonably good depending on the surface roughness. In this chapter we show how to employ the quality control method when a DEM is obtained by IFSAR technology taking into account the corresponding hypotheses.

Here we reformulate the Perpendicular Distance Estimation Method (PDEM) for the evaluation of IFSAR DEM accuracy. We obtain the accuracy in the azimuth-axis, range-axis and in elevation. These last two values may depend on the value of the range coordinate and the H parameter (overage antenna height). The method is viable depending on the surface feature conditions. In (Zelasco et al, 2007) the results of simulation are shown.

We develop an application to use the method with real or simulated DEM's. Finally, we propose a way to determine the evaluation of the results precision, so that, applying the method to a particular IFSAR DEM, a user knows the precision of the obtained results.

As it is see in article I, a typical way to build Digital Elevation Models (DEM) for Geographical Information Systems (GIS) is by processing aerial or satellite data. As we said the geometric quality control of the stored DEM is what we are concerned with here. « Quality » means the geometric precision measured in terms of the difference between a DEM and a reference DEM (R-DEM). Also in this case (IFSAR DEM) we assume the R-DEM is a faithful model of the actual surface. Its point density may be greater than the DEM point density.

We wish to assess the discrepancy that gives rise to the error. Normally, we need to identify without ambiguity each point *M* in the DEM with its homologous point *P* in the R-DEM.

Two reasons that can be read in the article I, make this task difficult.

To find identifiable homologous pairs of points is difficult for an operator. Automating this is a very delicate process.

We don't need to know the position of the homologous points with the proposed PDEM.

Let $e(M_k) = M_k - P_k$ where M_k, P_k are the k selected homologous pairs of points in the DEM and R-DEM respectively. Estimating $\sigma^2(e)$ (variance of the error as distance between *M* and *P*) without measuring each vector $e(M_k)$ completely, is the key advantage of the proposed PDEM. Only the projection of each $e(M_k)$ in particular directions matters. Given an M_k it becomes unnecessary to find the homologous P_k in the R-DEM. How this is done is the subject of the article I.

The goal of this PDEM is to evaluate the actual errors ('a posteriori') of a given DEM.

For the description of the PDEM see article I.

IFSAR CORRELATION PROBLEM: METHOD REFORMULATION

Now we outline the way in which the general estimator for variances in the three-dimensional case is applied to the IFSAR-derived DEM.

Figure 1 shows the geometrical problem and the correlation between the range-axis *y* and the *z*-axis. The covariance matrix is diagonal (no correlation) when a rotation around the azimuth-axis *x* is done. The rotation makes the *z*'-axis in the direction of the beam. This hypothesis is justified by the formulas presented in (Dupont, 1997; Kervyn, 2001;, Massonnet et al, 1993; Massonnet et al, 1996; Massonnet et al, 1995; Rabus et al, 2003; Touting et al, 2000) where the coordinates in the range-axis and the elevation value are function of the distance radar-target and the radar-target direction.

The *x*-axis (the azimuth-axis) is parallel to the antenna trajectory A_1. The *y*'-axis is normal to the *z*'-axis and belongs to the plane defined by the *z*'-axis and the vertical direction. With this rota-

tion the three directions will not have correlated random variables. In the case of an IFSAR-derived DEM, the measured errors in the *z*'-axis and in the *y*'-axis (after the rotation) are the ones caused by the measurement error of the antenna-target distance and the antenna-target direction. The rotation angle θ is the angle between the vertical direction and the beam direction.

B is the base (distance between A_1 and A_2), *H* is the antenna A_1 height and *R* is the distance from the antenna to the target *C*. The *z* value is given by:

$z = H - R \cos \theta$: *H* and *R* are known but $\cos \theta$ is unknown.

But:

$\cos \theta = \cos \varepsilon \cos inus (\theta - \varepsilon) - \sin \varepsilon \sin us (\theta - \varepsilon)$

And,

$\cos (\theta - \varepsilon) = (1 - \sin^2(\theta - \varepsilon))1/2$

In the $A_1 A_2 C$ made by the two antennas A_1 et A_2 and the target *C*; $R + r$ is the distance between A_2 and *C*, and α the angle in A_1.

Then:

$(R+r)^2 = B^2 + R^2 + 2BR \cos \alpha$

Where *R*, *B*, *r* (*r* is function of the difference of phase and the wave length) are known and

$\alpha = 90 + \theta - \varepsilon$

With:

$\cos \alpha = \sin (\theta - \varepsilon)$,

The height *z* of the target is equal to:

$z = H - R \cos \theta$

and,

$y = R \sin \theta$

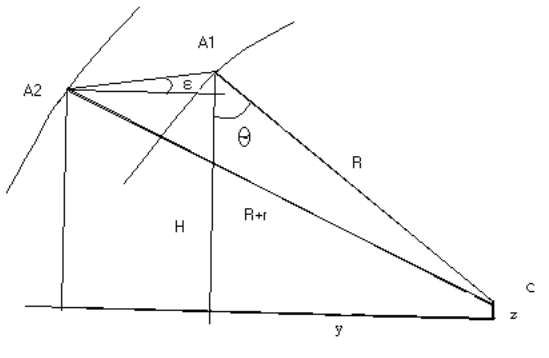
That is *y* and *z* are correlated

Different hypotheses may be considered in relation to the distance and direction errors. The distance error (in the *z*' direction) and the direction error (*y*' normal to *z*' direction and in the *yz* plane) may be functions of the measured distance. For the simulations we will consider arbitrary values, but these errors may be linear functions of the distance. Other functions of the distance can be assumed to model these errors. We may guess independence of the distance for one or for both errors. The error evaluation in each particular real case will give the elements to fix these functions. Indeed, a way to establish these functions could be to evaluate these errors for far and for near range points. With this information, functions may be derived.

METHOD DESCRIPTION

The method we describe below is similar to the one described in article I in which we have just included a few modifications and added some complements

Figure 1. IFSAR geometry



Recall (Zelasco, 2002a; Zelasco, 2004; Zelasco et al, 2005; Zelasco, 2005) that a R-DEM will be used as the control for a DEM, real or simulated. Also recall that a R-DEM, being a discrete set, allows constructing a mesh of *K* triangles defined by the points on the reference surface. Each triangle T_k, *(k=1,..,K),* belongs to a plane which has a normal unitary vector n_k, *(k=1,..,K)* .

The mesh of *K* triangles of the R-DEM is constructed using, for instance, the Delaunay method in two dimensions.

In the case of a simulation we have to:

- Determine the mass center of each triangle, and express their coordinates in the rotated system. The rotation angle will be a function of the coordinate value in the range-axis of the triangle mass center,
- Adopt a standard deviation for each coordinate σ_x, $\sigma_{y'}$ and $\sigma_{z'}$ and add three random normal variables to the center mass coordinates that are expressed in the new coordinate system. These points simulate the DEM to be evaluated. Note that $\sigma_{y'}$ and $\sigma_{z'}$ eventually have to take into account the distance between this simulated point and the antenna. This distance is easy to determine knowing the height of the antenna trajectory.

In the case of a real DEM we have to:

- Determine the triangle of the R-DEM containing the homologous point of each point of the DEM. The procedure is as follows: given a DEM point M_s , its homologous point *P* in the R-DEM will be in a plane define by the triangle T_s, if the projection of the point M_s on the *(x,y)* plane is inside the projection of the triangle T_s on the same plane. We establish a practical limit to avoid border problems given by the 'a priori' standard deviation $\sigma_{p'}$ (horizontal error in the x or the y' new coordinates direction) which allows us to select the points of the DEM in the sample. If the horizontal distance between the projection of M_s and the limits of the projected triangle is, v.g., at least three times greater than $\sigma_{p'}$, the point is accepted. Otherwise, it is rejected. This is because $\sigma_{p'}$ is the standard deviation of a normal random variable.
- Express the coordinates of the selected point in the rotated frame. The rotation angle will be a function of the range-axis DEM point value.

In both cases we have to:

- Express the normal unitary vector n_k the new coordinate system,
- Calculate the distance from each point (point of the simulated or of the real DEM) to the plane of the corresponding triangle,
- Evaluate, using the PDEM, the standard deviation of the DEM points: σ_z, $\sigma_{y'}$ and $\sigma_{z'}$. In the case of the simulation their values should be similar to the ones already used.

Note 1: Knowledge of the γ angle makes it possible to select a R-DEM with more or less unevenness.

Note 2: For each of the planes having the direction d given by the normal unitary vector n_k in question, the distance from the point M_j , (*j=1.. J<=I*) to the corresponding plane is a measure (in the direction *d*) of the error e_d,.

DISCUSSION

Selecting triangles normally making small angles with the *x* axis improves the error estimation in the *x* axis. Otherwise the error estimation in the mentioned axis may be not good (Zelasco, 2005).

In the axis *y´*, the results are satisfactory due to the rotation round the axis *x* corresponding to the angles of observation of the radar beams.

The axis z´ evaluation is excellent and it is not affected by the systematical error introduced by slanted surfaces as it happens in the commonly used vertical distance estimator (see Article I).

Nevertheless is important to consider (Dupont, 1997) when the points are vertically under the vector it is not possible to know their height, because the surface can not be observed from the radar. In these cases the corresponding DEM-R triangle has to be excluded, and the point measurements are aberrant.

CONCLUSION

To estimate the elevation error (σ_z), by means of the commonly used vertical distance estimator, when the surface is obtained by IFSAR is not reasonable because the correlation between *z* and *y* axis. Using the PDEM and taking in consideration the hypotheses admitted here, the IFSAR error may be estimated satisfactorily.

REFERENCES

Dupont, S. (1997). *Génération de modèles numériques de terrain par interférométrie ROS*. Thèse, Université de Nice- Sophia Antipolis.

Guptill, S. C., & Morrison, J. L. (1995). *Elements of spatial data quality.* Oxford: Elsevier.

Harvey, F. (1997). *Quality Needs More Than Standards. Data Quality in Geographic Information: From Error to Uncertainty.* M. Goodchild & R. Jeansoulin (Eds.). Paris: Hermès. (pp. 37-42).

Hirano, A., Welch, R., & Lang, H. (2003). Mapping from ASTER stereo image data: DEM validation and accuracy assessment. *ISPRS Journal of Photogrammetry & Remote Sensing,* 57, 356-370.

Hottier, P. (1996). *Précis de statistiques.* Notes de cours ENSG, IGN, France.

Kervyn, F. (2001). Modeling topography with SAR interferometry: Illustration of a favorable and less favorable environment. *Computer & Geosciences, 27,* 1039-1050.

Laurini, R., & Milleret-Raffort, F. (1993). *Les bases de données en géomatique*. Editions HERMES, Paris.

Massonnet, D., & Rabaute, T. (1993). Radar Interferometry: limits and potential. *IEEE Trans on Geosciences and Remote Sensing 31,* 455–464.

Massonnet, D., Vadon, H., & Rossi, M. (1996). Reduction of the need for phase unwrapping in radar interferometry. *IEEE Transaction on Geosciences and Remote Sensing, 34,* 489-497.

Massonnet, D., & Feigl, K. L. (1995). Radar interferometry and its applications to changes in the Earth surface. *Reviews of Geophysics, 36,* 441–500.

Rabus, B., Eineder, M., Roth, A., & Bamler, R. (2003). The shuttle radar topography mission – a new class of digital elevation models acquired by spaceborne radar. *Photogrammetry & Remote Sensing, 57,* 241-262.

Touting, T., & Gray, L. (2000). State - of - the - art of elevation extraction from satellite SAR data. *Photogrammetry and Remote Sensing, 55,* 13-33.

Ubeda, T., & Servigne, S. (1996). Geometric and Topological Consistency of Spatial Data. *Proceedings of the 1st International Conference on Geocomputation*, (pp. 830-842, vol. 1), Leeds, United Kingdom, September 17-19.

Ubeda, T. (1997). *Contrôle de la qualité spatiale des bases de donnes géographiques: cohérence topologique et corrections d'erreurs.* Thèse, Laboratoire d'Ingénierie des Systèmes d'Information de L'INSA.France.

Zelasco, J. F., & Ennis, K. (2000) *Solución de un problema en la estimación de variancias en un*

caso especial en el que no se pueden aplicar estimadores habituales. XXVIII Coloquio Argentino de estadística, Posadas, Misiones, Argentina.

Zelasco, J. F., Ennis, K., & Blangino, E. (2001). Quality Control in Altimetry and Planimetry for 3D Digital Models. *8th European Congress for Stereology and Image Analysis.* Bordeaux, France.

Zelasco, J. F. (2002a). Contrôle de qualité des modèles numériques des bases de données géographiques. *Journal XYZ 90,* (pp. 50-55), Association Française de Topographie.

Zelasco, J. F. (2002). Geographical database integrity. International Conference on Computer Science, Software Engineering, Information Technology. *e-Business and Applications (CSITeA-02).* Foz do Iguazu, Brazil.

Zelasco, J. F., & Núñez, M. (2004). *Control de calidad de modelos numéricos de bases de datos en Sistemas de Información Geográficos*, Contribuciones a la geodesia aplicada, 1/2004, Instituto de Geodesia de la Facultad de Ingeniería de la Universidad de Buenos Aires.

Zelasco, J. F., Porta, G., & Fernández Ausinaga, J. L. (2004). Geographic Information Systems. *Digital Elevation Model: Geometric Quality Control Method,* chapitre du livre : Encyclopedia of Database Technologies and Applications, A publier par le Idea Group Inc. in spring 2005.

Zelasco, J. F. (2005). SAR Interferometry: DEM Quality Control Method. Assestement and Applications. *Congreso 6ª Semana Geomática Barcelona del 8 al 11 de febrero* 2005.

Zelasco, J. F., & Donayo, J. (2007). *Control de Calidad Geométrica de MNT obtenidos por IFSAR por medio del PDEM.* Contribuciones a la geodesia aplicada, in press, Instituto de Geodesia de la Facultad de Ingeniería de la Universidad de Buenos Aires.

KEY TERMS

Advanced Spaceborned Thermal Emission and Reflection Radiometer (ASTER): One of five remote sensory devices on board the Terra satellite launched into Earth orbit by NASA in 1999. The instrument has been collecting data since February 2000. ASTER provides high-resolution images of the Earth in 14 different bands of the electromagnetic spectrum, ranging from visible to thermal infrared light. The resolution of images ranges between 15 to 90 meter. ASTER data is used to create detailed maps of surface temperature of land, emissivity, reflectance, and elevation.

Correlated Random Variables: Probability theory and statistics, correlation, (often measured as a correlation coefficient) , indicates the strength and direction of a linear relationship between random variables. In general statistical usage, correlation refers to the departure of two variables from independence. In this broad sense there are several coefficients, measuring the degree of correlation, adapted to the nature of data

Delaunay Method: A set P of points in the plane is a triangulation DT(P) such that no point in P is inside the circum circle of any triangle in DT(P). Delaunay triangulations maximize the minimum angle of all the angles of the triangles in the triangulation; they tend to avoid «sliver» triangles. The triangulation was invented by Boris Delaunay in 1934.

Interferometric Synthetic Aperture Radar (IFSAR): A radar technique used in geodesy and remote sensing. This geodetic method uses two or more synthetic aperture radar (SAR) images to generate maps of surface deformation or digital elevation, using differences in the phase of the waves returning to the satellite, or aircraft. The technique can potentially measure centimetre-scale changes in deformation over timespans of days to years. It has applications for geophysical

monitoring of natural hazards, for example earthquakes, volcanoes and landlsides, and also in structural engineering, in particular monitoring of subsidence and structural stability.

Chapter XLV
Querying and Integrating P2P Deductive Databases

Luciano Caroprese
University of Calabria, Italy

Sergio Greco
University of Calabria, Italy

Ester Zumpano
University of Calabria, Italy

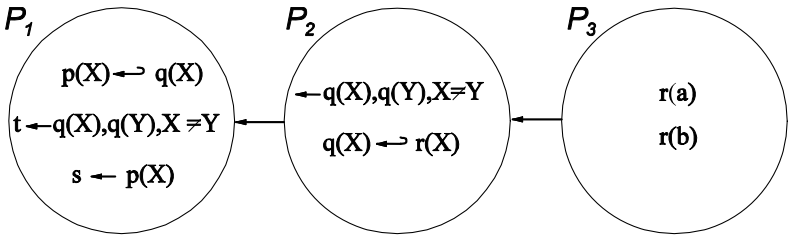
INTRODUCTION

Recently, there have been several proposals that consider the integration of information and the computation of queries in an open-ended network of distributed peers (Bernstein, Giunchiglia, Kementsietsidis, Mylopulos, Serafini, & Zaihrayen, 2002; Calvanese, De Giacomo, Lenzerini, & Rosati, 2004; Franconi, Kuper, Lopatenko, & Zaihrayeu, 2003) as well as the problem of schema mediation and query optimization in P2P (peer-to-peer) environments (Gribble, Halevy, Ives, Rodrig, & Suciu, 2001; Halevy, Ives, Suciu, & Tatarinov, 2003; Madhavan & Halevy, 2003; Tatarinov & Halevy, 2004).

Generally, peers can both provide or consume data and the only information a peer participating in a P2P system has is about neighbors, that is, information about the peers that are reachable and can provide data of interest. More specifically, each peer joining a P2P system exhibits a set of mapping rules, in other words, a set of semantic correspondences to a set of peers that are already part of the system (neighbors). Thus, in a P2P system, the entry of a new source, or peer, is extremely simple as it just requires the definition of the mapping rules. By using mapping rules as soon as it enters the system, a peer can participate and access all data available in its neighborhood, and through its neighborhood it becomes accessible to all the other peers in the system.

As stated before, the problem of integrating and querying databases in P2P environments has been investigated in Calvanese, De Giacomo, Lenzerini, et al. (2004) and Franconi et al. (2003). In both works, peers are modeled as autonomous agents that can export only data belonging to their knowledge, that is, data that are true in all possible

Figure 1. A P2P system



scenarios (models). In Calvanese, De Giacomo, Lenzerini, et al., new semantics for a P2P system, based on epistemic logic, is proposed.

The work also shows that the semantics is more suitable than traditional semantics based on FOL (first-order logic) and proposes a sound, complete, and terminating procedure that returns certain answers to a query submitted to a peer. In Franconi et al. (2003), a characterization of P2P database systems and a model-theoretic semantics dealing with inconsistent peers is proposed. The basic idea is that if a peer does not have models, all (ground) queries submitted to the peer are true (i.e., are true with respect to all models). Thus, if some databases are inconsistent, it does not mean that the entire system is inconsistent.

MOTIVATION

The motivation of this work stems from the observation that previously proposed approaches result in not being sound with respect to query answering when some peer is inconsistent.

Example 1. Consider the P2P system depicted in Figure 1 consisting of three peers P_1, P_2, and P_3 where

- P_3 contains two atoms, $r(a)$ and $r(b)$,
- P_2 imports data from P_3 using the (mapping) rule $q(X) \leftarrow r(X)$ (observe that a special syntax is used for mapping rules). Moreover, imported atoms must satisfy the constraint $\leftarrow q(X), q(Y), X \neq Y$ stating that the relation q may contain at most one tuple, and
- P_1 imports data from P_2 using the (mapping) rule $p(X) \leftarrow q(X)$. P_1 also contains the rules $s \leftarrow p(X)$ stating that s is true if the relation p contains at least one tuple, and $t \leftarrow p(X), p(Y), X \neq Y$, stating that t is true if the relation p contains at least two distinct tuples.

The intuition is that, with $r(a)$ and $r(b)$ being true in P_3, either $q(a)$ or $q(b)$ could be imported in P_2 (but not both, otherwise the integrity constraint is violated) and, consequently, only one tuple is imported in the relation p of the peer P_1. Note that whatever the derivation is in P_2, s is derived in P_1 while t is not derived. Therefore, the atoms s and t are, respectively, true and false in P_1. It is worth noting that the approaches above mentioned do not capture such a semantics. Indeed, the epistemic semantics proposed in Calvanese, De Giacomo, Lenzerini, et al. (2004) states that both the atoms $q(a)$ and $q(b)$ are imported in the peer P_2, which becomes inconsistent. In this case, the semantics assumes that the whole P2P system is inconsistent and every atom is true as it belongs to all minimal models. Consequently, t and s are true. The semantics proposed in Franconi et al. (2003) assumes that only P_2 is inconsistent as it has no model. Thus, as the atoms $q(a)$ and $q(b)$ are true in P_2 (they belong to all models of P_2), then the atoms $p(a)$ and $p(b)$ can be derived in P_1 and finally t and s are true.

The idea we propose in this work consists of importing in each peer maximal sets of atoms not violating integrity constraints.

BACKGROUND

We assume there are finite sets of predicate symbols, constants, and variables. A term is either

a constant or a variable. An atom is of the form $p(t_1,\ldots,t_n)$ where p is a predicate symbol and $t_1,\ldots, t_n$ are terms. A literal is either an atom A or its negation *not A*. A rule is of the form $H \leftarrow B$, where H is an atom (head of the rule) and B is a conjunction of literals (body of the rule). A program P is a finite set of rules. P is said to be positive if it is negation free. The definition of a predicate p consists of all rules having p in the head. A ground rule with empty body is a fact. A rule with empty head is a constraint. It is assumed that programs are safe; that is, variables appearing in the head or in negated body literals are range restricted as they appear in some positive body literal.

The ground instantiation of a program P, denoted by *ground*(P), is built by replacing variables with constants in all possible ways. An interpretation is a set of ground atoms. The truth value of ground atoms, literals, and rules with respect to an interpretation M is as follows: $val_M(A) = A \in M$, $val_M(not\ A) = not\ val_M(A)$, $val_M(L_1,\ldots,L_n) = min\{val_M(L_1),\ldots, val_M(L_n)\}$ and $val_M(A \leftarrow L_{1,\ldots,} L_n) = val_M(A) \geq val_M(L_{1,\ldots,} L_n)$, where A is an atom, and $L_{1,\ldots,} L_n$ are literals and true > false. An interpretation M is a model for a program P if all rules in *ground*(P) are true with respect to M. A model M is said to be minimal if there is no model N such that $N \subset M$. We denote the set of minimal models of a program P with $MM(P)$. Given an interpretation M and a predicate symbol g, $M[g]$ denotes the set of g-tuples in M. The semantics of a positive program P is given by its unique minimal model, which can be computed by applying the immediate consequence operator T_P until the fix point is reached ($T_P^{\infty}(\varnothing)$). The semantics of a program with negation P is given by the set of its stable models, denoted as $SM(P)$. An interpretation M is a stable model (or answer set) of P if M is the unique minimal model of the positive program P^M, where P^M is obtained from *ground*(P) by (Gelfond & Lifschitz, 1988)

- removing all rules r such that there exists a negative literal *not A* in the body of r and A is in M and
- removing all negative literals from the remaining rules.

It is well known that stable models are minimal models (i.e., $SM(P) \subseteq MM(P)$) and that for negation-free programs, minimal- and stable-model semantics coincide (i.e., $SM(P) = MM(P)$). A preference relation $\geq$ among atoms is defined as follows (Sakama & Inoue, 2000). Given two atoms e_1 and e_2, the statement $e_2 \geq e_1$ is a priority stating that for each a_2 instance of e_2 and for each a_1 instance of e_1, a_2 has higher priority than a_1. If $e_2 \geq e_1$ and *not* $e_1 \geq e_2$, we write $e_2 > e_1$. If $e_2 > e_1$, the sets of ground instantiations of e_1 and e_2 have empty intersection. The relation $\geq$ is transitive and reflexive. A prioritized logic program (PLP) is a pair (P, Φ) where P is a program and Φ is a set of priorities. Φ^* denotes the set of priorities that can be reflexively or transitively derived from Φ. Given a prioritized logic program (P, Φ), the relation $\blacktriangleright$ is defined over the stable models of P as follows. For any stable models M_1, M_2, and M_3 of P,

- $M_1 \blacktriangleright M_1$,
- $M_2 \blacktriangleright M_1$ if
  - $\exists\, e_2 \in M_2 - M_1$; $\exists\, e_1 \in M_1 - M_2$ such that $(e_2 \geq e_1) \in \Phi^*$ and
  - *not* $\exists\, e_3 \in M_1 - M_2$ such that $(e_3 > e_2) \in \Phi^*$
- if $M_2 \blacktriangleright M_1$ and $M_1 \blacktriangleright M_0$, then $M_2 \blacktriangleright M_0$.

If $M_2 \blacktriangleright M_1$ holds, then we say that M_2 is preferable to M_1 with respect to Φ. Moreover, we write $M_2 \rhd M_1$ if $M_2 \blacktriangleright M_1$ and *not* $M_1 \blacktriangleright M_2$. An interpretation M is a preferred stable model of (P, Φ) if M is a stable model of P and there is no stable model N such that $N \rhd M$. The set of preferred stable models of (P, Φ) will be denoted by $PSM(P, \Phi)$.

P2P SYSTEMS: SYNTAX AND FOL SEMANTICS

A (peer) predicate symbol is a pair $i : p$ where i is a peer identifier and p is a predicate symbol. A (peer) atom is of the form $i : A$ where A is a standard atom. A (peer) literal is of the form *not A* where A

is a peer atom. A (peer) rule is of the form $A \leftarrow A_1,...,A_n$, where A is a peer atom and $A_1,...,A_n$ are peer atoms or built-in atoms. A (peer) integrity constraint is of the form $\leftarrow L_1,...,L_m$ where $L_1,...,L_m$ are peer literals or built-in atoms. Whenever the peer is understood, the peer identifier can be omitted. The definition of a predicate $i : p$ consists of all rules having as head predicate symbol $i : p$. In the following, we assume that for each peer P_i there are three distinct sets of predicates called, base, derived, and mapping predicates. A base predicate is defined by ground facts; a derived predicate $i : p$ is defined by standard rules, that is, peer rules using in the body only predicates defined in the peer P_i. A mapping predicate $i : p$ is defined by mapping rules, in other words, peer rules using in the body only predicates defined in other peers. Without loss of generality, we assume that every mapping predicate is defined by only one rule of the form $i : p(X) \leftarrow j : q(X)$ with $j \neq i$. The definition of a mapping predicate $i : p$ consisting of n rules of the form $i : p(X_k) \leftarrow B_k$ with $1 \leq k \leq n$ can be rewritten into $2n$ rules of the form $i : p_k(X_k) \leftarrow B_k$ and $i : p(X) \leftarrow i : p_k(X)$, with $1 \leq k \leq n$.

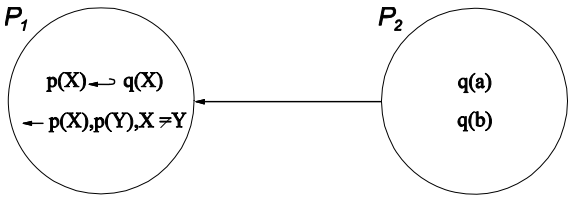
Definition 1. A peer P_i is a tuple $\langle D_i, LP_i, MP_i, IC_i \rangle$ where

- D_i is a (local) database consisting of a set of facts,
- LP_i is a set of standard rules,
- MP_i is a set of mapping rules, and
- IC_i is a set of constraints over predicates defined in D_i, LP_i, and MP_i.

A P2P system PS is a set of peers $\{P_1,...,P_n\}$.

Given a P2P system $PS = \{P_1,...,P_n\}$ where $P_i = \langle D_i, LP_i, MP_i, IC_i \rangle$, we denote as D, LP, MP, and IC respectively the global sets of ground facts, standard rules, mapping rules, and integrity constraints: $D = D_1 \cup ... \cup D_n$, $LP = LP_1 \cup ... \cup LP_n$, $MP = MP_1 \cup ... \cup MP_n$ and $IC = IC_1 \cup ... \cup IC_n$. With a little abuse of notation, we shall also denote with PS both the tuple $\langle D, LP, MP, IC \rangle$ and the set $D \cup LP \cup MP \cup IC$. Given a peer P_i, $MPred(P_i)$, $DPred(P_i)$, and $BPred(P_i)$ denote, respectively, the sets of mapping, derived, and base predicates defined in P_i. Analogously, $MPred(PS)$, $DPred(PS)$, and $BPred(PS)$ define the sets of mapping, derived, and base predicates in PS.

Figure 2. The system PS



FOL Semantics

The FOL semantics of a P2P system $PS = \{P_1,...,P_n\}$ is given by the minimal-model semantics of $PS = D \cup LP \cup MP \cup IC$. For a given P2P system PS, $MM(PS)$ denotes the set of minimal models of PS. As $D \cup LP \cup MP$ is a positive program, PS may admit zero or one minimal model. In particular, if $MM(D \cup LP \cup MP) = \{M\}$, then $MM(PS) = \{M\}$ if $M \models IC$; otherwise, $MM(PS) = \emptyset$. The problem with such a semantics is that local inconsistencies make the global system inconsistent.

WEAK-MODEL SEMANTICS

This section introduces a new semantics, called weak-model semantics, based on a new interpretation of mapping rules, which will now be denoted with a different syntax of the form $H \rightharpoonup B$. Intuitively, $H \rightharpoonup B$ means that if the body conjunction B is true in the source peer, the atom H could be imported in the target peer; that is, H is true in the target peer only if it does not imply (directly or indirectly) the violation of some constraints.

Preferred Weak Models

Informally, the idea is that for a ground mapping rule $A \rightharpoonup B$, the atom A could be inferred only if the

body B is true. Formally, given an interpretation M, a ground standard rule $C \leftarrow D$ and a ground mapping rule $A \hookleftarrow B$, $val_M(C \leftarrow D) = val_M(C) \geq val_M(D)$, whereas $val_M(A \hookleftarrow B) = val_M(A) \leq val_M(B)$.

Definition 2. Given a P2P system $PS = D \cup LP \cup MP \cup IC$, an interpretation M is a weak model for PS if $\{M\} = MM(St(PS^M))$, where PS^M is the program obtained from *ground*(PS) by removing all mapping rules whose head is false with respect to M.

We shall denote with $M[D]$ (resp. $M[LP]$, $M[MP]$) the set of ground atoms of M that are defined in D (resp. LP, MP).

Definition 3. Given two weak models M and N, we say that M is preferable to N, and we write $M \blacktriangleright N$, if $M[MP] \supseteq N[MP]$. Moreover, if $M \blacktriangleright N$ and not $N \blacktriangleright M$, we write $M \rhd N$. A weak model M is said to be preferred if there is no weak model N such that $N \rhd M$.

The set of weak models for a P2P PS system will be denoted by $WM(PS)$, whereas the set of preferred weak models will be denoted by $PWM(PS)$.

Proposition 1. For any P2P system PS, $\blacktriangleright$ defines a partial order on the set of weak models of PS.

The next theorem shows that P2P systems always admit preferred weak models.

Theorem 1. For every consistent P2P system PS, $PWM(PS) \neq \emptyset$.

Example 2. Consider the P2P system PS depicted in Figure 2.

P_2 contains the facts $q(a)$ and $q(b)$, whereas P_1 contains the mapping rule $p(X) \hookleftarrow q(X)$ and the constraint $\leftarrow p(X), p(Y), X \neq Y$. The weak models of the system are $M_0 = \{q(a), q(b)\}$, $M_1 = \{q(a), q(b), p(a)\}$, and $M_2 = \{q(a), q(b), p(b)\}$, whereas the preferred weak models are M_1 and M_2.

Prioritized Programs and Preferred Stable Models

We now present an alternative semantics based on the rewriting of mapping rules into prioritized rules (Brewka, Niemela, & Truszczynski, 2003; Sakama & Inoue, 2000). For the sake of notation, we consider rules of the form $A \oplus A' \leftarrow B$ whose meaning is that if B is true, then either A or A' must be true.[1]

Definition 4. Given a P2P system $PS = D \cup LP \cup MP \cup IC$ and a mapping rule $r = i : p(x) \hookleftarrow B$, then

- $Rew(r)$ denotes the pair $(i : p(x) \oplus i : p'(x) \leftarrow B; i : p(x) \geq i : p'(x))$, consisting of a disjunctive mapping rule and a priority statement,
- $Rew(MP) = (\{Rew(r)[1] \mid r \in MP\}; \{Rew(r)[2] \mid \in MP\})$ and
- $Rew(PS) = (D \cup LP \cup Rew(MP)[1] \cup IC ; Rew(MP)[2])$.

In the above definition, the atom $i : p(x)$ (resp. $i : p'(x)$) means that the fact $p(x)$ is imported (resp. not imported) in the peer P_i. For a given mapping rule r, $Rew(r)[1]$ (resp. $Rew(r)[2]$) denotes the first (resp. second) component of $Rew(r)$.

Example 3. Consider again the system analyzed in Example 2. The rewriting of the system is $Rew(PS) = (\{q(a), q(b), p(X) \oplus p'(X) \leftarrow q(X), \leftarrow p(X), p(Y); X \neq Y\}; \{p(X) \geq p'(X)\})$. $Rew(PS)[1]$ has three stable models: $M_0 = \{q(a), q(b), p'(a), p'(b)\}$, $M_1 = \{q(a), q(b), p(a), p'(b)\}$, and $M_2 = \{q(a), q(b), p'(a), p(b)\}$, but only M_1 and M_2 are preferred.

Given a P2P system PS and a preferred stable model M for $Rew(PS)$, we denote with $St(M)$ the subset of nonprimed atoms of M and we say that $St(M)$ is a preferred stable model of PS. We denote the set of preferred stable models of PS as $PSM(PS)$. The following theorem shows the equivalence of preferred stable models and preferred weak models.

Theorem 2. For every P2P system *PS*, *PSM*(*PS*) = *PWM*(*PS*).

For the system of the previous example, $PSM(PS)=\{\{q(a), q(b), p(a)\}, \{q(a), q(b), p(b)\}\}$.

QUERY ANSWERS AND COMPLEXITY

We now consider the computational complexity of calculating preferred weak models and answers to queries. As a P2P system may admit more than one preferred weak model, the answer to a query is given by considering brave or cautious reasoning (also known as possible and certain semantics). Given a P2P system $PS = \{P_1,...,P_n\}$ and a ground peer atom *A*, then *A* is true under:

- brave reasoning if $A \in \cup_{M \in PWM(PS)} M$, or
- cautious reasoning if $A \in \cap_{M \in PWM(PS)} M$.

The following lemma states that for every P2P system *PS*, an atom is true in some of its preferred weak models if and only if it is true in some of its weak models.

Lemma 1. $\cup_{M \in PWM(PS)} M = \cap_{M \in PWM(PS)} M$

The upper bound results can be immediately fixed by considering analogous results on stable-model semantics for prioritized logic programs. For (disjunction-free) prioritized programs, deciding whether an atom is true in some preferred model is Σ_2^P complete, whereas deciding whether an atom is true in every preferred model is Π_2^P complete.

Theorem 3. Let *PS* be a P2P system.

1. Deciding whether an interpretation M is a preferred weak model of *PS* is coNP complete
2. Deciding whether an atom A is true in some preferred weak model of *PS* is NP complete
3. Deciding whether an atom A is true in every preferred weak model of PS is in Π_2^P and coNP hard.

REFERENCES

Bernstein, P. A., Giunchiglia, F., Kementsietsidis, A., Mylopulos, J., Serafini, L., & Zaihrayen, I. (2002). Data management for peer-to-peer computing: A vision. *WebDB* (pp. 89-94).

Bertossi, L. & Bravo, L. (2004). Query answering in peer-to-peer data exchange systems. *EDBT Workshop* (pp. 476-485).

Brewka, G., & Eiter, T. (1999). Preferred answer sets for extended logic programs. *AI, 109*(1-2), 297-356.

Brewka, G., Niemela, I., & Truszczynski, M. (2003). Answer set optimization. *IJCAI* (pp. 867-872).

Calvanese, D., De Giacomo, G., Lembo, D., Lenzerini, M., & Rosati, R. (2004). Inconsistency tolerance in P2P data integration: An epistemic logic approach. *DBPL* (pp. 692-697).

Calvanese, D., De Giacomo, G., Lenzerini, M., & Rosati, R. (2004). Logical foundations of peer-to-peer data integration. *PODS* (pp. 241-251).

Delgrande, J. P., Schaub, T., & Tompits, H. (2003). A framework for compiling preferences in logic programs. *TPLP, 3*(2), 129-187.

Franconi, E., Kuper, G. M., Lopatenko, A., & Zaihrayeu, I. (2003). A robust logical and computational characterisation of peer to-peer database systems. *DBISP2* (pp. 64-76).

Gelfond, M., & Lifschitz, V. (1988). The stable model semantics for logic programming. *ICLP* (pp. 1070-1080).

Gribble, S., Halevy, A., Ives, Z., Rodrig, M., & Suciu, D. (2001). What can databases do for peer-to-peer? *WebDB* (pp. 31-36).

Halevy, A., Ives, Z., Suciu, D., & Tatarinov, I. (2003). Schema mediation in peer data management systems. *ICDT* (pp. 505-516).

Lenzerini, M. (2002). Data integration: A theoretical perspective. *PODS* (pp. 233-246).

Madhavan, J., & Halevy, A. Y. (2003). Composing mappings among data sources. *VLDB* (pp. 572-583).

Sakama, C. & Inoue, K. (2000). Prioritized logic programming and its application to commonsense reasoning. *AI, 123*(1-2), 185-222.

Tatarinov, I., & Halevy., A. (2004). Efficient query reformulation in peer data management systems. *SIGMOD* (pp. 539-550).

KEY TERMS

Consistent Database: A database satisfying a set of integrity constraints.

Inconsistent Database: A database violating some integrity constraints.

Integrity Constraints: Set of constraints that must be satisfied by database instances.

P2P System: A collection of autonomous nodes providing or requesting data.

Section V
Ontologies

Chapter XLVI
Using Semantic Web Tools for Ontologies Construction

Gian Piero
University of Paris IV / Sorbonne, France

INTRODUCTION

The current state of Web technology – the "first generation" or "syntactic" Web – gives rise to well-known, serious problems when trying to accomplish, in a non-trivial way, essential tasks like indexing, searching, extracting, maintaining, and generating information. These tasks would, in fact, require some sort of 'deep understanding' of the information dealt with: in a "syntactic" Web context, on the contrary, computers are only used as tools for posting and rendering information by brute force. Faced with this situation, Tim Berners-Lee first proposed a sort of "Semantic Web" where the access to information is based mainly on the processing of the *semantic properties* of this information: "… the Semantic Web is an extension of the current Web in which information is given well-defined *meaning* (emphasis added), better enabling computers and people to work in co-operation" (Berners-Lee *et al.*, 2001: 35). The Semantic Web's challenge consists of being able to access and retrieve information on the Web by "*understanding*" its proper semantic content (its meaning), and not simply by matching some keywords.

From a technical point of view, the Semantic Web vision is deeply rooted into an "ontological" approach, with some *proper characteristics* that differentiate it from the "classical" approach to the construction of ontologies based on a methodology of the frame" type (Chaudhri *et al.*, 1998) and on the use of tools in the "standard" Protégé style (Noy, Fergerson and Musen, 2000). We will describe these characteristics in the following Sections.

BACKGROUND INFORMATION

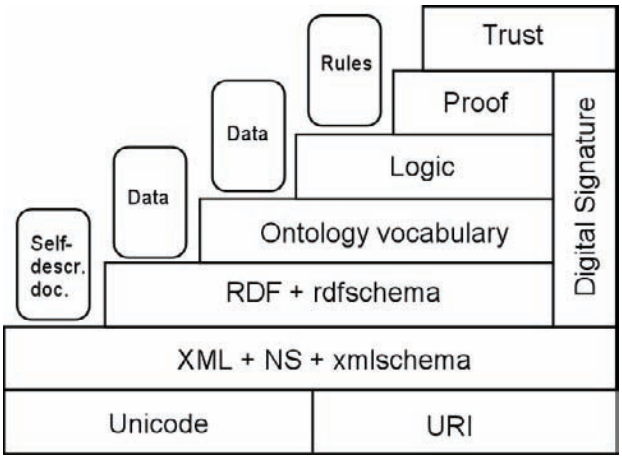
Berners Lee's Architectural Proposal for the Semantic Web

To support his vision, Berners-Lee has proposed in several papers an 'architecture' for the Semantic Web like that reproduced in Figure 1. Making abstraction now from all the discussions and criticisms that this proposal has brought up, see al so the 'Conclusion', what is relevant for the topic of this article is the central position that 'ontologies' occupy in the architecture. We can note immediately that a first, important difference with respect to the 'classical' approach evoked above is that ontologies are no more considered 'in isolation': they are now supported by lower-level tools like XML and RDF and must also implement and additional logic level.

In the embedded architecture of Figure 1, 'Unicode' and 'URI' make up the basis of the hierarchy. The Unicode Standard provides a unique numerical code for every character that can be found in documents produced according to any possible language, no matter what is the hardware and software used to deal with such documents. It is supported in many operating systems and all the modern browsers, and it enables a single software product or a single website to be targeted across multiple platforms, languages and countries without re-engineering. URI (Uniform Resource Identifier) represents a generalization of the well-known URL (Uniform Resource Locator), which is used to identify a 'Web resource' (e.g., a particular page) by denoting its primary access mechanism (essentially, its 'location' on the network). URI has been created to allow recording information about all those 'notions' that, unlike Web pages, do not have network locations or URLs, but that need to be referred to in an RDF statement. These notions include network-accessible things, such as an electronic document or an image, things that are not network-accessible, such as human beings, corporations, and bound books in a library, or abstract concepts like the concept of a 'creator'.

XML (eXtensible Markup Language), see (Bray *et al*., 2004), has been created to overcome some difficulties proper to HTML (Hypertext Markup Language), developed in 1989 by Tim Barners-Lee as a means for sharing information from any locations. An HTML file is a text file characterized by the presence of a small set of 'tags' – like <Head>, <Body>, <Input>, <Applet>, <Font> etc. – that instruct the Web browsers how to display a given Web page. HTML is, then, a

Figure 1. Semantic Web architecture according to Tim Berners-Lee



'presentation-oriented' markup tool. In spite of its evident utility, HTML suffers from a number of limitations, from its lack of efficiency in handling the complex client/server communication of today applications to (mainly) the impossibility of defining new tags to customize exactly the user's needs. XML is called 'extensible' because, at the difference of HTML, is not characterized by a fixed format, but it lets the user design its own customized markup languages (a specific DTD, Document Type Description, see below) for limitless different types of documents; XML is a 'content-oriented' markup tool.

Basically, the syntactic structure of XML is very simple. Its markup elements are normally identified by an opening and a closing tag, like <employees> and </employees>, and may contain other elements or text; the elements must be properly nested and every XML document must have exactly one root element. Markup elements can be characterized by adding attribute/value pairs inside the opening tag of the element, like <person name="Mary">; taking into account the nesting constraint, a very simple fragment of XML document could then be represented as: <employees> <person name="Mary"> <id>99276</id> </person> </employees>. To allow a computer interpreting correctly an XML fragment like this it is necessary, however, to specify the semantics of the markup elements and tags used to make up it; the simplest and 'traditional' way of doing this is to make use of a DTD. A DTD is a formal description in XML Declaration Syntax of a particular type of document: it begins with a <!DOCTYPE keyword and sets out what names are to be used for the different types of markup element, where they may occur, the elements' possible attributes, and how they all fit together. For example – see the example above – a DTD may specify that every person markup element must have a name attribute, and that it can have an offspring element called id whose content must be text. Before reading an XML document, the validating parsers and the application programs (editors, search engines, navigators, databases) read the corresponding DTD so that they can identify where every element type ought to come and how each relates to the other. There are many sorts of DTDs ready to be used in all kinds of areas (see, e.g., http://www.w3.org/QA/2002/04/valid-dtd-list.html#full), that can be downloaded and used freely: some of them are MathML, for mathematical expressions, SMIL, Sync Multimedia Integration Language, CML, Chemical Markup Language, OSD, Open Software Description, EDI, Electronic Data Interchange, PICS, Platform for Internet Content Selection, etc.

A more complete way of specifying the semantics of a set of XML markup elements is to make use of XML Schema (as mentioned in Figure 1 above). XML Schema, see (Thompson *et al.*, 2001; Biron and Malhotra, 2001), is an XML-based alternative to DTDs that, like these last, describes the structure of an XML documents, but that supplies a more complete grammar for specifying this structure allowing us, e.g., to define the cardinality and the order of the offspring elements, to define data types for elements and attributes, to define default and fixed values for these elements and attributes, etc. Moreover, XML Schemas supports namespaces, and are extensible for future additions.

RDF, Resource Description Framework

Moving up in the structure of Figure 1, we find now RDF (Resource Description Framework), an example of 'metadata' language (metadata = data about data) used to describe generic 'things' ('resources', according to the RDF jargon) on the Web. An RDF document is, basically, a list of statements under the form of triples having the classical format: <object, property, value>, where the elements of the triples can be URIs (Universal Resource Identifiers, see above), literals (mainly, free text) and variables. To follow a well-known RDF example – see, e.g., (Manola & Miller, 2004)

– let us suppose we want to represent a situation where someone named John Smith has created a particular Web page. We will then make use of the RDF triple: <http://www.example.org/index.html **(***object***),** creator **(***property***),** john_smith **(***value***)**>. Adding additional information about the situation, by stating, e.g., that the Web page was created October 20, 2004, and that the language in which the page is written is English, amounts to add two additional statements: <http://www.example.org/index.html **(***object***),** creation_date **(***property***),** October 20, 2004 **(***value***)> and** <http://www.example.org/index.html **(***object***),** language **(***property***),** English **(***value***)>. Note that** RDF uses a particular terminology for denoting the three elements of the triples, calling then 'subject', 'predicate' and 'object', respectively, the 'object', 'property' and 'value' elements of the triples; this decision is really questionable because it introduces an undue confusion with well defined and totally different linguistic categories.

RDF triples can be represented as directed labeled graphs, by denoting resources as *ovals*, properties (predicates) as *arrows*, and literal values like October 20, 2004 **or** English *within boxes*. Figure 2a represents then under graph form the original statement: "John Smith has created a Web page"; the addition of information about date and language gives rise to the graph of Figure 2b, given that groups of statements are represented by corresponding groups of nodes and arcs.

Note that, to simulate the true conditions of utilization of RDF, the properties creator, creation_date and language in Figure 2 have been replaced, respectively, by http://pur1.org/dc/elements/1.1/creator, http://www.example.org/terms/creation-date and http://pur1.org/dc/elements/1.1/language; in a similar way, the value john_smith has been replaced by http://www.example.org/staffid/85740. All these http://… terms are URIs that identify in an unambiguous way specific RDF entities; more exactly, they refer to the ontologies/metadata repositories/lists of reserved domain names where these entities are defined. For example, http://pur1.org/dc/… refers to the collection of metadata terms maintained by the Dublin Core Metadata Initiative, see (Dekkers & Weibel, 2003); in this collection of terms, e.g., http://pur1.org/dc/elements/1.1/creator is defined as: "An entity primarily responsible for making the content of the resource". The literal en (Unicode characters) is an international standard two-letter code for English, see http://purl.org/dc/elements/1.1/language; the example.org Internet domain name is reserved for documentation purposes.

From what expounded until now, RDF seems to be nothing more than a downgraded form, Internet-oriented, of Semantic Networks as they were used in the Artificial Intelligence domain at the beginning of the seventies. Its significance in a Semantic Web context becomes more evident when we examine the way of writing RDF statement into XML format – the so-called 'RDF/XML syntax', see (Beckett, 2004) – i.e., when RDF is seen as a sort of additional DTD of XML. Table 1 reproduces then the simple example of Figure 2b making use of the RDF/XML syntax.

The first line of the code, <?xml version="1.0"?> is the 'XML declaration', which states that what follows consists of XML, and which specifies the version used. In the second line we find an XML markup element that starts with the tag <rdf:RDF – this tag specifies that all the following XML code, until the </rdf:RDF> tag of the last line, is intended to represent RDF statements – and ends with the > symbol at the right limit of line 4. Within this markup element we find three 'XML attributes', see above, of the opening <rdf:RDF tag; all these attributes (xmlns attributes) have as values the declarations of the namespaces to be used within the RDF/XML code. An attribute like xmlns:rdf means that, according to the 'value' associated with this attribute (after the = symbol), all the terms/tags included in this RDF/XML content and prefixed with rdf: are part of the namespace identified with the URI: http://www.w3.org/1999/02/22-rdf-syntax-ns#;

Figure 2. RDF statements represented into graph format

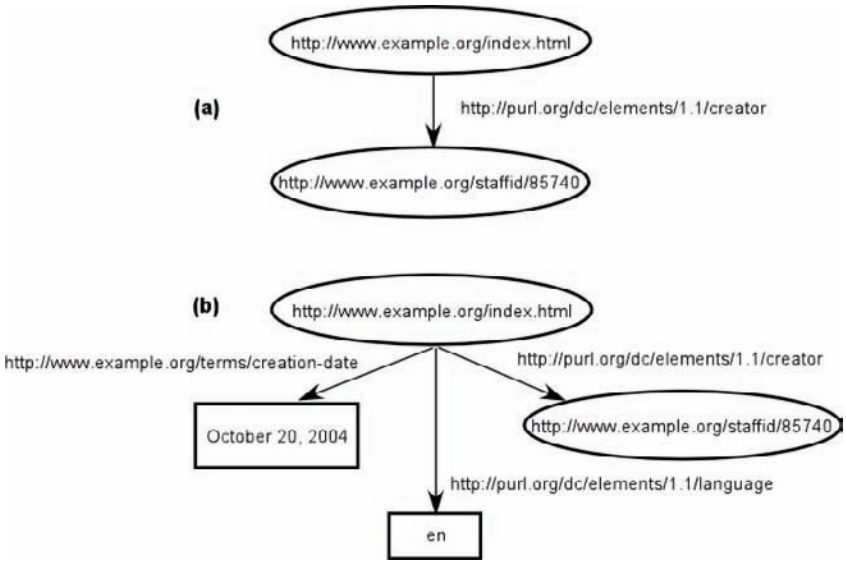


Table 1. The RDF/XML syntax

```
<?xml version="1.0"?>
 <rdf:RDF    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
             xmlns:dc="http://purl.org/dc/elements/1.1/"
                    xmlns:exterms="http://www.example.org/terms/">
     <rdf:Description rdf:about="http://www.example.org/index.html">
             <exterms:creation-date>October 20, 2004</exterms:creation-date>
             <dc:language>en</dc:language>
             <dc:creator rdf:resource=»http://www.example.org/staffid/85740»/>
 </rdf:Description>
 </rdf:RDF>
```

analogously for the xmlns:dc (Dublin Core terms) and xmlns:exterms (example terms) attributes.

After these preliminary, 'housekeeping' declarations, lines from 5 to 9 represent the core of the RDF/XML representation of the example. The rdf:Description start-tag of line 5 indicates that we are now introducing the 'description' of a resource; this resource, http://www.example.org/index.html, is identified as the value of the rdf:about attribute of the start-tag. The three following lines, 6-8, are examples of use of 'property element' constructions. In these lines, the tags are built up according to the XML Qname (Qname = Qualified name) convention, which allows shortening the writing of full RDF triples by introducing abbreviations for the URI references. A Qname tag contains, in fact, a 'prefix' that denotes a given namespace (e.g., exterms in line 6) followed, after a 'colon', by a 'local name' (creation-date, i.e., the name of the property); a

full URI reference is then created by appending the local name to the URI of the namespace identified by the first part of the Qname. For lines 6-8, the full URIs become then http://www.example.org/terms/creation-date, http://pur1.org/dc/elements/1.1/Language, and http://pur1.org/dc/elements/1.1/creator. Note that the values of the properties corresponding to literals (lines 6-7) are directly included within opening and closing Qname tags; for the property of line 8, which corresponds to a resource, the value corresponds to the value of the rdf:resource attribute of the dc:creator Qname tag. The description of the resource introduced in line 5 ends with the closing tag of line 9.

To conclude about RDF, we will note that RDF Schema (or RDFS), see (Brickley and Guha, 2004), is a very important complement to RDF that provides a mechanism for constructing *specialized RDF vocabularies* through the description of *application-specific classes and properties*: the RDFS facilities are provided as a specialized set of *predefined RDF resources with their own special meaning.* We can then use the constructs rdfs:Class to declare that a particular resource (denoted then by an URL) is a class *denoting a whole set of resources*; these resources are called the '*instances*' of the class. More precisely, the instances of a specific class are declared making use of the rdf:type property, i.e., when a resource has an rdf:type property whose value is some specific class, than we can consider that this resource is an instance of the specified class.

To describe the properties that can be associated with the classes, RDFS makes use of the RDF construct rdf:Property, to be interpreted now as the class of all the RDF properties: rdf:Property is then an instance of rdfs:Class. Some specific, useful properties – instances of rdf:Property – are defined in RDFS, like rdfs:domain and rdfs:range. For example, we could define the creator property saying that it has the resource document as domain – i.e., the resource document is necessarily associated with this property – and that it has the resource person as 'range' (rdfs:range) – the value of the creator property is always of the person type. Moreover, one of the main interests of RDFS consists in the possibility of setting up *hierarchies*, both *hierarchies of concepts* (i.e., ontologies) and *hierarchies of properties*. RDFS provides then the specific property rdfs:subClassOf, which allows us to build up hierarchies of classes by relating a subclass to its superclass. Analogously, we will make use of the constructs rdfs:subPropertyOf to declare that one property is a sub-property of another.

OWL, THE WEB ONTOLOGY LANGUAGE

The Main Structural Principles of the OWL Language

On February 10, 2004, the W3C published an official 'recommendation' concerning OWL, the Web Ontology Language, see (Bechhofer *et al.*, 2004); W3C – the World Wide Web Consortium, coordinated by MIT (USA), ERCIM, the European Research Consortium for Informatics and Mathematics, and the Keio University (Japan) – includes all the main bodies on earth interested in the developments of Internet and the Web. At the beginning of this document it is stated that: "The Web Ontology Language (OWL) is a semantic markup language for publishing and sharing ontologies on the World Wide Web. OWL is developed as a vocabulary extension of RDF (the Resource Description Framework) and is derived from the DAML+OIL Web Ontology Language … An OWL ontology is an RDF graph, which is in turn a set of RDF triples". The mention of DAML+OIL (McGuinness *et al.*, 2002) explains the strong logic orientation of OWL, given that OIL (Ontology Inference Layer), the 'European' component of DAML+OIL (DAML is the Darpa Agent Markup Language) was implemented in Description Logics (DLs) terms – DLs (Baader

et al., 2002) have been created to offer, among other things, a formal foundation for standard frame-based systems.

OWL consists of three subsets (three specific sub-languages) characterized by an increasing level of complexity and expressiveness, OWL Lite, OWL DL (DL stands for Description Logics) and OWL Full.

OWL Lite is the syntactically simplest sub-languages; it includes only a reduced sub-set of the OWL language constructors and has a lower formal complexity than the other OWL versions. It is meant mainly to allow i) the implementation of simple classification hierarchies; ii) the familiarization with the OWL approach; iii) the possibility of a quick migration path for existing thesauri/taxonomies and other conceptually simple hierarchies. It employs all the features already introduced by RDFS (see the end of the previous Section) making use of the same tags – like rdfs:subclassOf**,** rdfs:subPropertyOf**,** rdfs:domain**,** rdfs:range – with the same semantics. Note that rdfs:subclassOf is then *the fundamental constructor that is used to set up ontologies in OWL*. At already stated, it relates a more specific class (concept) to a more general one; if *X* is a sub-class of *Y*, then every instance of *X* is also an instance of *Y*. The relation rdfs:subclassOf is transitive: if *X* is a sub-class of *Y* and *Y* is a sub-class of *Z*, then *X* is a sub-class of *Z*. With respect to RDFS, OWL Lite includes some new features:

- Constructors for equality and inequality, i.e., owl:equivalentClass**,** owl:equivalentProperty**,** owl:sameAs (two individuals may be stated to be the same), owl:differentFrom**,** owl:AllDifferent**.**
- Constructors used to provide specific information about properties and their values, like owl:inverseOf – e.g., stating that the property hasChild is the inverse of the property hasParent, and stating that Mary is endowed with the property (hasParent Lucy), this allows an OWL reasoner to deduce that Lucy is endowed with the property (hasChild Mary) – owl:TransitiveProperty and owl:SymmetricProperty.
- Constructors used to impose constraints on the way properties can be used by the instances of a class (concept). They are owl:allValuesFrom and owl:someValuesFrom. For example, owl:allValuesFrom introduces a range restriction, imposing, e.g., that the property hasDaughter of the class Person is restricted to obtaining all its values (allValuesFrom) from the class Woman. This allows a reasoner to deduce that, if an individual Lucy is related by the property hasDaughter with the individual Mary, Mary must be an instance of the class Woman.
- Constructors, e.g., owl:minCardinality and owl:maxCardinality, used to introduce a *limited form of cardinality restrictions*, stated on the properties of a particular class, and to be interpreted as constraints on the cardinality of that property when used in the instances of that class. Note that, for algorithmic efficiency reasons, OWL Lite allows using only the integers 0 and 1 to express the cardinality constraints; this restriction is removed in OWL DL.
- OWL Lite includes also a (restricted form) of intersection constructor, owl:intersectionOf, allowing, e.g., to state that the class EmployedPerson is the intersectionOf the classes Person and EmployedThings.

OWL DL is much more expressive than OWL Lite and is totally based on Description Logics – this is denoted by the suffix DL. OWL DL makes use of the full set of the OWL constructors, but it introduces also some constraints on their use to give rise to systems that are '*complete*' (all the possible deductions are computable) and '*decidable*' (all the computations will be executed in finite time). Mainly, OWL DL implements what is called 'type separation', which means that a class (concept) cannot be also an individual or a property, and that a property cannot be also an

individual or a class. This restriction is removed in OWL Full. OWL DL adds to the OWL Lite list of constructors some new constructors like owl:oneOf (some classes may be described by enumeration of the individuals that make up the class), owl:hasValue (a property is required to have a given individual as value), owl:disjointWith (some classes can be described as disjoint from each other, see the classes Man and Woman), owl: unionOf, owl:complementOf, owl:intersactionOf (Boolean combinations of classes), etc.

OWL Full is the most expressive of the OWL sub-languages, and is should be used in situations where very high expressiveness is particularly important. It is similar to OWL DL but, in the OWL Full case, all the constraints have been suppressed – e.g., a class (concept) can be simultaneously treated as a collection of instances (individuals) and as an individual in itself. This can lead to the implementation of systems that are, at least partly, 'incomplete' and/or 'undecidable': this means that it is not possible to perform automated reasoning on OWL Full hierarchies. Currently, no complete implementation of OWL Full exists.

An Example of OWL Ontology

To give now an at least partial picture of the representation of an ontology in OWL format, we reproduce in Table 2 a small fragment of the OWL version of the 'wine' ontology, an ontology often used for exemplification's purposes in the Semantic Web milieus, see (McGuinness *et al.*, 2002; Smith *et al.* 2004). The code in this Table can be considered indifferently as Owl Lite, OWL DL or OWL Full. Note that, for simplicity's sake, we have not reproduced in Tables 2 and the following the 'housekeeping' declarations, see Table 1 above, that are necessary to identify all the XML namespaces associated with this ontology.

In the first line of Table 2, the class Wine is introduced making use of an rdf:ID attribute. At the difference of the rdf:about attribute used in Table 1 above, rdf:ID introduces as its value only a '*fragment identifier*' (here Wine) that represents an abbreviation of the complete reference to the URI of the resource being described. The full URI reference is formed by taking the base URI of the wine ontology, e.g., http://www.w3.org/TR/2004/REC-owl-guide-20040210/wine, and appending the character # (to indicate that what follows is a fragment identifier) and then Wine to it, giving then the absolute URI reference: http://www.w3.org/TR/2004/REC-owl-guide-20040210/wine#Wine. Note that the Wine class can now be referred to by using #Wine; e.g., rdf:resource="#Wine" is a well-formed OWL statement. As already stated, the fundamental 'ontological' constructor is rdfs:subClassOf; the

Table 2. A fragment of the OWL wine ontology

```
<owl:Class rdf:ID=»Wine»>
 <rdfs:subClassOf rdf:resource=»#PotableLiquid»/>
 <rdfs:subClassOf>
      <owl:Restriction>
               <owl:onProperty rdf:resource=»#madeFromGrape»/>
               <owl:minCardinality rdf:datatype=»&xsd;nonNegativeInteger»>1</owl:minCardinality>
      </owl:Restriction>
 </rdfs:subClassOf>
 ...
</owl:Class>
```

second line of the code of Table 2 allows then the insertion of the class Wine into the global ontology by asserting that it is a specialisation of the class PotableLiquid (liquid suitable for drinking) – which can be defined, in turn, as a specialisation of the class ConsumableThing.

The third line of the code warns that the class Wine is also a specialisation of a second class: this last is an 'anonymous' class, whose definition is included within the opening owl:Restriction markup element in line 4 and ends with the closing /owl:Restriction markup element in line 7. In OWL, in fact, a property restriction on a class is a special kind of class description, corresponding to the anonymous class including all the individuals that satisfy the given restriction. In line 5, the owl:onProperty constructor introduces then the name of the property, madeFromGrape, to associate with the class Wine; line 6 specifies that the cardinality of this property is 1. The insertion of this restriction in the definition of Wine states, globally, that every specific wine must also be characterized by at least one madeFromGrape relation. Note that i) the &xsd;nonNegativeInteger datatype used to introduce the literal 1 in the owl:minCardinality restriction of line 6 is part of the built-in XML Schema datatypes, see (Biron and Malhotra, 2001): their use is strongly recommended in an OWL context; ii) the value 1 complies with the OWL Lite restrictions.

Following (Smith *et al.* 2004: 11-13), we can now supply, in the upper part of Table 3, the definition of the Vintage class – vintage is a particular wine made in a specific year. The lower part of Table 3 shows how the property vintageOf ties a Vintage to a Wine; the rdfs:domain and rdfs:range features indicate, respectively, that the property vintageOf can only be associated with terms of the Vintage type (e.g., RomaneeConti1998), and that the values of this property can only be specific terms of the Wine hierarchy, e.g., RomaneeConti.

To conclude about OWL, we reproduce in Table 4 another fragment of the wine ontology that makes use of the specific constructors proper to the DL version of the language. The code fragment of Table 4 defines the class RedWine as the precise intersection (logical conjunction, 'and') of the class Wine and the set of things that are red in color (anonymous class). The presence of the attribute rdf:parseType="Collection" `is mandatory for this type of construction`. Note the use of the DL constructor owl:hasValue to impose the value Red on the property hasColor of the anonymous class.

Table 3. Vintage *class and* vintageOf *property*

```
<owl:Class rdf:ID="Vintage">
 <rdfs:subClassOf>
      <owl:Restriction>
                        <owl:onProperty rdf:resource="#vintageOf"/>
                        <owl:minCardinality rdf:datatype="&xsd;nonNegativeInteger">1</owl:minCardinality>
      </owl:Restriction>
 </rdfs:subClassOf>
</owl:Class>

<owl:ObjectProperty rdf:ID="vintageOf">
 <rdfs:domain rdf:resource="#Vintage" />
 <rdfs:range rdf:resource=»#Wine» />>
</owl:ObjectProperty>
```

Table 4. Use of OWL DL constructors in the context of the wine ontology

```
<owl:Class rdf:about=»#RedWine»>
      <owl:intersectionOf rdf:parseType=»Collection»>
      <owl:Class rdf:about=»#Wine» />
      <owl:Restriction>
      <owl:onProperty rdf:resource=»#hasColor» />
      <owl:hasValue rdf:resource=»#Red» />
      </owl:Restriction>
  </owl:intersectionOf>
</owl:Class>
```

FUTURE TRENDS

Rules – self-contained knowledge units that entail some form of reasoning – are explicitly mentioned in the upper level of the architecture of Figure 1 and they are, obviously, an essential prerequisite for a practical utilization of the RDF/OWL data structures. However, rules have not been included in the OWL standard, and they are still a hot topic of discussion in the Semantic Web and W3C milieus.

In RuleML, see (Boley *et al*., 2001), the inferential properties of Prolog/Datalog are associated with an XML/RDF-based rule format – for an introduction to Prolog/Datalog and to their logical support, Horn clauses, see, e.g., (Bertino *et al*. 2001, pp. 112-121, 170-207). The main categories of rules considered in RuleML are the 'derivation rules' (i.e., rules used to automatically defining derived concepts), 'integrity rules' (constraints on the state space), 'reaction rules' (for specifying the reactive behavior of a given system in response to specific events), 'production rules' (if-then rules according to the classical expert systems paradigm) and 'transformation rules' (used to implement translators between different versions of RuleML, and between RuleML and other rule languages like Jess).

TRIPLE, see (Sintek and Decker, 2002), is a rule language that also follows the logic programming paradigm; its core is based on Horn logic clauses, and it has been syntactically extended to support RDF primitives like namespaces, resources, and RDF triples – these last have given TRIPLE its name. Rules expressed in this core language can be compiled into Horn logic programs and then executed by Prolog inference engines like XSB.

SWRL is a recent proposal, see (Horrocks *et al*., 2004; 2005), based on a combination of OWL DL with the Datalog sub-language of RuleML. Concretely, the proposal extends the set of the OWL axioms to include Horn-like rules, enabling then these rules to be combined with an OWL knowledge base. The rules have the form of an implication between an antecedent (body) and a consequent (head); their meaning corresponds to say that, whenever the conditions specified in the antecedent hold, then the conditions specified in the consequent also hold. Both the antecedent and the consequent consist of zero or more atoms; atoms can be in the form of C(*x*), P(*x, y*), sameAs(*x, y*) or differentFrom(*x, y*), where C is an OWL description, P is an OWL property, and x, y are either variables, OWL individuals or OWL data values. An XML- and an RDF-based syntax for the rules have also been defined.

Another 'hot topic' in a Semantic Web context concerns Semantic Web Services. A 'normal' Web service can be defined as a Web site that does not

simply supply static information, but that allows also to execute automatically some 'actions' (services), like the sale of a product or the control of a physical device; an increasing number of Web Services are accessible on the Web, developed by independent operators or large companies such as Amazon and Google. To carry out their tasks, Web Services must provide interoperability among diverse applications, using platform- and language independent interfaces for a smooth integration of heterogeneous systems. This has led to a standardization of the Web Service descriptions, discovery and invocation, making use of XML-based standards like WSDL (Christensen *et al.*, 2001), a description protocol, and SOAP (Mitra, 2003), a messaging protocol. However these standards, in their present form, are characterized by a low level of semantic expressiveness: for example, WSDL can be used to describe the interface of the different services, and how these services are deployed via SOAP, but it is very limited in its ability to express what the overall competences of this service are. Semantic Web Services are then Web Services that can specify not only their interfaces, but also *describe in full their capabilities, and the prerequisites and consequences of their use.*

OWL-S (Semantic Markup for Web Services), see (Ankolekar *et al.*, 2002) – formerly DAML-S – is a specification, in the form of an OWL-based ontology, that describes different Semantic Web Services features. It should enable Web users and software agents to automatically discover, invoke, select, compose and monitor Web-based services. The ontology is structured into three main parts. The 'profile' component supplies a general description of a particular Web Service by specifying the input and output types, the preconditions and the effects. The 'process model' component describes how the Web Service works and the Web Service interaction protocol; each service is either an atomic process that can be executed directly or a combination of several processes. An example of atomic process can be a service that returns a postal code, or the longitude and latitude when supplied with an address. A complex service often requires some form of interaction with the user, who can make choices and provide information conditionally: an example can be that of a personal shopping agent, that can assist the user in finding and buying many different sorts of items, requiring, in case, credit card and mailing information. The 'grounding' component specifies how the atomic processes defined in the process model can be mapped into WSDL descriptions, able to directly call up the described (atomic) service.

The ODE SWS framework, see (Gómez-Pérez *et al.*, 2004), proposes both an ontology to describe Semantic Web Services and an environment to support their graphical development. A characteristic of this framework is the use of Problem Solving Methods to describe these Services at the 'knowledge level', i.e., independently of the language in which they will be actually expressed. The ODE SWS ontology reproduces the upper level concepts of OWL-S, with the exception of the concepts associated with the 'process model' component that are substituted by method descriptions. The ODE SWS environment is composed of three main layers. The 'data source' layer is devoted to the integration of external applications. The 'domain' layer includes the main modules of the environment, as the SWSOntologiesManager or the SWSInstanceCreator – this last module creates, from the graphical description of a particular service, the corresponding instance of a concept pertaining to the OWL-S ontology. The 'presentation layer' consists of a SWSDesigner module, i.e., a user-friendly graphical interface that the user employs to describe a Service – according to the authors, this graphically-oriented process is more simple and less error prone than manipulating directly instance of the internal OWL-S ontology.

We can conclude this short notes about Semantic Web Services by mentioning WSMO (Web Service Modeling Ontology), see (de Bruijn

et al., 2005), whose main characteristic consists in the use of several types of '*mediators*' for the resolution of all the types of heterogeneity that can rise in the context of a Semantic Web service. For example, ooMediators links are used to resolve differences and conflicts among ontologies; wwMediators resolve process and protocol differences between services provided by a given business and other services they depend on; wgMediators resolve protocol and process differences between the requester and the provider; etc.

CONCLUSION

The use of Semantic Web tools in the 'ontological' domain is surely one of the *most interesting novelties* introduced these last years in the Knowledge Representation field. This approach, however, is still far from having reached a real consensus and has raised several criticisms from the beginning.

First of all, languages like RDF and OWL are characterized by a low level of 'expressiveness' because of their intrinsic '*binary*' nature – they cannot make use, in fact, of relations (properties) other then binary (i.e., taking only *two arguments*). This makes it extremely difficult, for example, to use them for dealing with the complex situations and events described in those multimedia documents (*nonfictional narratives*), omnipresent and of a particular economic importance, that correspond to corporate knowledge papers, news stories, normative and legal texts, medical records, intelligence messages, surveillance videos, actuality photos for newspapers and magazines, material (text, image, video, sound…) for eLearning etc., see [Zarri, 2005] in this context.

Moreover, Berners-Lee's original architecture has raised many criticisms in the Computer Science milieus giving that it seems to ignore some fundamental components of Computer Science today, from database technology (the whole world economy runs on SQL) to UML (Unified Modeling Language) – and this in spite of the fact that UML has, e.g., a type hierarchy comparable with OWL, and a class diagrams facility that can be compared to a frame-based language. A general discussion about the proposals for defining transformations between UML and the Semantic Web ontology languages can be found, e.g., in (Falkovych *et al.*, 2003). We can note in this context that Tim Berners-Lee and some colleagues have recently proposed a *wider architectural framework* than that depicted in Figure 1, where the Semantic Web is only one of the components of a general 'Science of the Web' – the science of decentralized information systems – and Web Services, P2P approaches, social factors, natural language processing techniques, Bayesian techniques etc. are also evoked, see [Berners-Lee *et al.*, 2006].

In the context of the present, 'standard' Semantic Web architecture, the choice of OWL as the *paradigmatic language* to be used for ontological work has also been criticized, and several knowledge representation specialists have challenged as hastily the endorsement of OWL by the W3C. Apart from the general RDF/OWL 'binary flaw' mentioned before and the use of a particularly cumbersome syntax, inherited from RDF/XML, specific '*practical*' criticisms about OWL concern, e.g., the conceptual difficulty of making use of this axiomatic and highly formalized language even for experienced software developers and the decision of building the central version of OWL (OWL DL) on Description Logics (DLs). These last were, in fact, already dismissed for concrete tasks at the beginning of the nineties because of their low degree of efficiency, linked to the reduced power of their main reasoning component, the automatic classification mechanism, see (Bertino *et al.* 2001, pp. 153-170). No industrial applications of OWL apparently exist at present, see also (Cardoso, 2007). It is then still unclear whether the use of OWL for setting up ontologies represents really a quantum leap with respect to 'traditional' frame approach in the Protégé style; we can however remark, in this last context, that

an 'OWL plugin' for Protégé has been recently implemented, see (Horridge, 2004) and http://protege.stanford.edu/plugins/owl/. It allows loading and saving OWL and RDF ontologies, editing and visualizing OWL classes and their properties and, mainly, supporting reasoners such as the DLs classifiers.

With respect to DLs we can also note that, according to the OWL's supporters, it is precisely the possibility of making use in OWL of the highly formalized, well-defined and axiomatic semantics of DLs that 'makes all the difference' with respect to the 'old' frame-based systems. These last can only employ, in fact, pragmatically-based inference procedures, whether the OWL-based reasoning tool – see, e.g., RACER (Haarslev and Möller, 2003) – can make use of really sound and complete inference algorithms guaranteed by the fully unambiguous DLs' semantics.

In spite of all the above criticisms, Semantic Web techniques are surely here to stay.

REFERENCES

Ankolekar, A., Burstein, M., Hobbs, J., Lassila, O., Martin, D., McDermott, D., McIlraith, S., Narayanan, S., Paolucci, M., Payne, T., & Sycara, K. (2002). DAML-S: Web service description for the Semantic Web. In *Proceedings of the First International Semantic Web Conference – ISWC 2002* (LNCS 2342). Heidelberg: Springer-Verlag.

Baader, F., Calvanese, D., McGuinness, D., Nardi, D., & Patel-Schneider, P. F., (eds.) (2002). The description logic handbook: Theory, implementation and applications. Cambridge: University Press.

Bechhofer, S., van Harmelen, F., Hendler, J., Horrocks, I., McGuinness, D. L., Patel-Schneider, P. F., & Stein, L.A., (eds) (2004). *OWL Web ontology language reference – W3C recommendation 10 February 2004.* W3C (http://www.w3.org/TR/owl-ref/).

Beckett, D., (ed.) (2004). *RDF/XML syntax specification (Revised) – W3C recommendation 10 February 2004.* W3C (http://www.w3.org/TR/rdf-syntax-grammar/).

Berners-Lee, T., Hall, W., Hendler, J. A., O'Hara, K., Shadbolt, N., & Weitzner, D. J. (2006). A framework for Web science. *Foundations and Trends in Web Science, 1,* 1-130.

Berners-Lee, T., Hendler, J. A., & Lassila, O. (2001). The Semantic Web. *Scientific American, 284*(5), 34-43.

Bertino, E., Catania, B., & Zarri, G. P. (2001). *Intelligent database systems.* London: Addison-Wesley and ACM Press.

Biron, P. V., & Malhotra, A. (eds.) (2001). *XML schema part 2: Datatypes – W3C recommendation 02 May 2001.* W3C (http://www.w3.org/TR/xmlschema-2/).

Boley, H., Tabet, S., & Wagner, G. (2001). Design rationale of RuleML: A markup language for Semantic Web rules. In *Proceedings of SWWS'01, The First Semantic Web Working Symposium.* Stanford: Stanford University.

Bray, T., Paoli, J., Sperberg-McQueen, C. M., Maler, E., & Yergeau, F. (eds.) (2004). *Extensible markup language (XML) 1.0 (Third Edition) – W3C recommendation 04 February 2004.* W3C (http://www.w3.org/TR/REC-xml/).

Brickley, D., & Guha, R. V. (eds.) (2004). *RDF vocabulary description language 1.0: RDF schema – W3C recommendation 10 February 2004.* W3C (http://www.w3.org/TR/rdf-schema/).

Cardoso, J. (2007). The Semantic Web vision : Where are we ? *IEEE Intelligent Systems, 22*(5), 84-88.

Chaudhri, V. K., Farquhar, A., Fikes, R., Karp, P. D., & Rice, J. P. (1998). OKBC: A programmatic foundation for knowledge base interoperability. In *Proceedings of the 1998 National Conference*

on Artificial Intelligence – AAAI/98. Cambridge, MA: MIT Press/AAAI Press.

Christensen, E., Curbera, F., Meredith, G., & Weerawarana, S. (2001). *Web services description language (WSDL) 1.1 – W3C Note 15 March 2001.* W3C (http://www.w3.org/TR/wsdl).

de Bruijn, J., Fensel, D., Keller, U., & Lara, R. (2005). Using the Web service modeling ontology to enable Semantic E-Business. *Communications of the ACM, 48*(12), 43-47.

Dekkers, M., & Weibel, S. (2003, April). State of the Dublin core initiative. *D-Lib Magazine, 9*(4). (http://www.dlib.org/dlib/april03/weibel/04weibel.html).

Falkovych, K., Sabou, & Stuckenschmidt, H. (2003). UML for the Semantic Web: Transformation-based approaches. In *Knowledge Transformation for the Semantic Web,* B. Omelayenko & M. Klein (eds.). Amsterdam: IOS Press.

Gómez-Pérez, A., González-Cabrero, R., & Lama, M. (2004). ODE SWS: A framework for designing and composing Semantic Web services. *IEEE Intelligent Systems, 19*(4), 24-31.

Haarslev, V., & Möller, R. (2003, October 20). Racer: A core inference engine for the Semantic Web. In *Proceedings of the 2nd International Workshop on Evaluation of Ontology Tools (EON2003)*, Sanibel Island, FL..

Horridge, M. (2004). *A practical guide to building OWL ontologies with the protégé-OWL plugin* (Edition 1.0). Manchester: The University of Manchester.

Horrocks, I., Patel-Schneider, P. F., Bechhofer, S., & Tsarkov, D. (2005). OWL Rules: A proposal and prototype implementation. *Journal of Web Semantics: Science, Services and Agents on the World Wide Web, 3,* 23-40.

Horrocks, I., Patel-Schneider, P. F., Boley, H., Tabet, S., Grosof, B., & Dean, M. (2004). *SWRL: A Semantic Web Rule Language Combining OWL and RuleML – W3C Member Submission 21 May 2004*. W3C (http://www.w3.org/Submission/SWRL/).

Manola, F., & Miller, E. (2004). *RDF primer – W3C Recommendation 10 February 2004.* W3C (http://www.w3.org/TR/rdf-primer/).

McGuinness, D. L., Fikes, R., Hendler, J., & Stein, L. A. (2002). DAML+OIL: An ontology language for the Semantic Web. *IEEE Intelligent Systems, 17*(5), 72-80.

Mitra, N., (ed.) (2003). *SOAP Version 1.2 Part 0: Primer – W3C Recommendation 24 June 2003.* W3C (http://www.w3.org/TR/soap12-part0/).

Noy, F. N., Fergerson, R. W., & Musen, M. A. (2000). The knowledge model of protégé-2000: Combining interoperability and flexibility. In *Knowledge Acquisition, Modeling, and Management – Proceedings of EKAW'2000* (LNCS 1937), R. Dieng & O. Corby (Eds.). Berlin: Springer.

Sintek, M., & Decker, S. (2002). TRIPLE – A query, inference, and transformation language for the Semantic Web. In *Proceedings of the First International Semantic Web Conference – ISWC 2002* (LNCS 2342). Heidelberg: Springer-Verlag.

Smith, M. K., Welty, C., & McGuinness, D. L., (eds.) (2004). *OWL Web Ontology Language Guide – W3C Recommendation 10 February 2004.* W3C (http://www.w3.org/TR/owl-guide/).

Thompson, H. S., Beech, D., Maloney, M., & Mendelsohn, N., eds. (2001). *XML Schema Part 1: Structures – W3C Recommendation 02 May 2001.* W3C (http://www.w3.org/TR/xmlschema-1/).

Zarri, G. P. (2005). An *n*-ary language for representing narrative information on the Web. In *SWAP 2005, Semantic Web Applications and Perspectives – Proceedings of the 2nd Italian Semantic Web Workshop* (CEUR-WS.org/Vol-166), Boquet, P., and Tummarello, G., eds. Aachen: Sun SITE Cen-

tral Europe (http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-166/63.pdf).

KEY TERMS

DTD (Document Type Description): A DTD is a formal description in XML Declaration Syntax of a particular type of document: it begins with a <!DOCTYPE keyword and sets out what names are to be used for the different types of markup element, where they may occur, the elements' possible attributes, and how they all fit together. For example, a DTD may specify that every person markup element must have a name attribute, and that it can have an offspring element called id whose content must be text. There are many sorts of DTDs ready to be used in all kinds of areas that can be downloaded and used freely, see, e.g., MathML, for mathematical expressions, CML, Chemical Markup Language, EDI, Electronic Data Interchange, PICS, Platform for Internet Content Selection, etc.

OWL: The Web Ontology Language (OWL) is a semantic markup language for publishing and sharing ontologies on the World Wide Web. OWL is developed as a vocabulary extension of RDF and is derived from the DAML+OIL Web Ontology Language. An OWL ontology is an RDF graph, which is in turn a set of RDF triples. OWL includes three specific sub-languages, characterized by an increasing level of complexity and expressiveness, OWL Lite, OWL DL – DL(s) stands for Description Logics, a particular, logic-oriented, knowledge representation language introduced to supply a formal foundation for frame-based systems – and OWL Full.

RDF (Resource Description Framework): an example of 'metadata' language (metadata = data about data) used to describe generic 'things' ('resources', according to the RDF jargon) on the Web. An RDF document is a list of statements under the form of triples having the classical format: <object, property, value>, where the elements of the triples can be URIs (Universal Resource Identifiers), literals (mainly, free text) and variables. RDF statements are normally written into XML format (the so-called 'RDF/XML syntax').

RDF Schema (RDFS): provides a mechanism for constructing specialized RDF vocabularies through the description of domain-specific properties. This is obtained mainly by describing the properties in terms of the classes of resource to which they apply: for example, we could define the creator property saying that it has the resource document as 'domain' (document is the value or 'object' of this property) and the resource person as 'range' (this property must always be associated with a resource person, its 'subject'). Other basic modelling primitives of RDFS allow setting up hierarchies (taxonomies), both *hierarchies of concepts* thanks to the use of class and subclass-of statements, and *hierarchies of properties* thanks to the use of property and subproperty-of statements. Instances of a specific class (concept) can be declared making use of the type statement.

Semantic Web Architecture: A layered architecture proposed by Berners-Lee for the Semantic Web applications. In this architecture, ontologies occupy a central place: they are built on the top of the RDF (Resource Description Framework) layer, which is in turn built on the top of the XML layer, see below. The XML/RDF base constraints the particular format ontologies assume in a Semantic Web context, inheriting, e.g., all the well-known XML 'verbosity'.

Semantic Web Rules: Still a 'hot' topic in a Semantic Web context. The present proposals (like RuleML, TRIPLE or SWRL) are based on an expansion of the classical 'logic programming' paradigm where the inferential properties of Prolog/Datalog are extended to deal with RDF/OWL knowledge bases. Examples of Semantic Web rules in RuleML are the 'derivation rules' (i.e., rules used to automatically defining derived

concepts), the 'reaction rules' (for specifying the reactive behavior of a given system in response to specific events), the 'transformation rules' (used to implement translators between different versions of RuleML, and between RuleML and other rule languages like Jess), etc.

Semantic Web Services: A Web service is a Web site that does not simply supply static information, but that allows also to execute automatically some 'actions' (services), like the sale of a product or the control of a physical device. To do this, Web services make use of XML-based standards like WSDL, a description protocol, and SOAP, a messaging protocol, characterized by a low level of semantic expressiveness. For example, WSDL can describe the interface of the different services, and how these services are deployed via SOAP, but it is very limited in its ability to express what the overall competences of this service are. Semantic Web Services are Web Services that can specify not only their interfaces, but also describe in full, under the form of OWL-based ontologies, their capabilities, and the prerequisites and consequences of their use. For example, OWL-S is a specification, in the form of an ontology, intended to describe different Semantic Web Services features, enabling then Web users and software agents to automatically discover, invoke, select, compose and monitor Web-based services.

XML (eXtensible Markup Language): Has been created to overcome some difficulties proper to HTML (Hypertext Markup Language) that – developed as a means for instructing the Web browsers how to display a given Web page – is a 'presentation-oriented' markup tool. XML is called 'extensible' because, at the difference of HTML, is not characterized by a fixed format, but it lets the user design its own customized markup languages (a specific DTD, Document Type Description, see below) for limitless different types of documents; XML is then a 'content-oriented' markup tool.

XML Schema: With respect to DTDs, a more complete way of specifying the semantics of a set of XML markup elements. XML Schema supplies a complete grammar for specifying the structure of the elements allowing, e.g., to define the cardinality of the offspring elements, default values, etc.

Chapter XLVII
Matching Relational Schemata to Semantic Web Ontologies

Polyxeni Katsiouli
University of Athens, Greece

Petros Papapanagiotou
University of Athens, Greece

Vassileios Tsetsos
University of Athens, Greece

Christos Anagnostopoulos
University of Athens, Greece

Stathes Hadjiefthymiades
University of Athens, Greece

INTRODUCTION

The Semantic Web (SW; Berners-Lee, Hendler, & Lassila, 2001) is already in its implementation phase and an indication of this is the intense research and development activity in the area of SW tools and languages. SW is based on metadata, which describe the semantics of the Web content. SW envisages the enrichment of data with semantics in order to be machine understandable and enable knowledge reasoning. The core element for achieving such Web evolution is ontology: "Ontology is an explicit specification of a conceptualization" (Gruber).

However, ontologies are just knowledge representation schemata and unless they are populated with real instances, they are of little value. A main problem of the SW is, currently, the lack of ontology instances. What is more challenging is the fact that the ontology elicitation process should be as automatic as possible in order to be effective and provide usable results in the short term. During the last decade, there have been proposed many approaches for producing such ontology instances (Staab & Studer, 2004). Some of them rely on controlled vocabularies and ontologies (e.g., WordNet) in order to create semantics metadata. Others try to provide mappings between existing information

sources, thus adopting an information integration approach. The methodology that will be presented in the rest of this article is more close to the latter approach. Specifically, it is based on matching relational schemata to ontologies.

This seems a quite promising (semi-)automatic ontology population method since it is well known that a lot of information in the Web is stored in relational databases and forms the so-called "Deep Web." In order to manage such information with the SW technologies, it is very important to develop a methodology that performs the migration of the relational data to ontology instances. Data migration relies on a schema matching process between the relational schema and the target ontology. Schema matching is considered a task based on the fact that both schemata (relational and ontological) differ in structure, expressiveness, and reasoning capability. In this article, we propose a methodology for schema matching and present a tool, called RONTO (relational to ontology), which deals with the semantic mapping between the elements of a relational schema to the elements of an ontological schema.

BACKGROUND

Regarding the source schema, we assume a relational database (RDB) schema deployed on a typical relational database management system (RDBMS).

The conceptual schema (CS) of the target ontology (ONT) is expressed in a description logic (DL) language, due to the popularity of DLs in the SW community. DLs are knowledge representation languages (subsets of first-order logic) that express knowledge about concepts and conceptual hierarchies. An ontology expressed in DL language consists of concepts (classes) that can be described by various constructs and may have several restrictions (axioms). Concepts are categorized to primitive and defined concepts. Roles define binary relationships between concepts or between a concept and a datatype. Concepts and roles of an ontology can be both organized in hierarchical structures through the inclusion relation $\sqsubseteq$ (i.e., is-a, generalization, or subsumption). In OWL-DL, which is a sublanguage of the Web ontology language (OWL; Antoniou & van Harmelen, 2001), the term role is referred to as property. A property has a domain and range. When the domain and the range of a property are (primitive or defined) concepts, the property is called object-property. In case the range of a property is a literal (e.g., integer, string), the property is called datatype-property.

In order to better describe the proposed methodology, several intermediate modeling elements are introduced.

Definition 1. A candidate concept for an ontology concept *c*, CC_c , can be (a) an RDB relation, (b) an RDB view, or (c) a combination of them, which is structurally and semantically similar to the concept *c* of the target ontology.

Definition 2. A candidate datatype-property for a datatype-property *p*, CDP_p, is an attribute of an RDB relation,[1] which has the same (or a compatible) datatype, and is semantically similar to the datatype-property *p* of the target ontology.

Definition 3. A candidate object-property for an object-property *p*, COP_p, is (a) a referential constraint, (b) an RDB relation representing an N:M relationship between relations, or (c) an RDB attribute, which is structurally and semantically similar to the object-property *p* of the target ontology.

Definition 4. A candidate concept set (CCS_C) for an ontology concept c is the set of all CCs that can be computed for the concept *c*. Hence, CCS_c is defined as follows.

$$\mathrm{CCS_c} \equiv \bigcup_{i=1}^{n} \{CC_{Ci}\}\,, \qquad (1)$$

where n is the number of CCs for concept c. In case $\mathrm{CCS_c} = \varnothing$, then no mapping exists for the concept c. Similarly, the candidate datatype-property set ($\mathrm{CDPS_p}$) and the candidate object-property set

($COPS_p$) are defined. Between each element e of such sets and each element *r* belonging to the target ontology, a degree of similarity, sim(e, r), is defined. The similarity threshold (i.e., the minimum acceptable similarity value) depends on the user.

Related Systems

Schema matching is a research field that has attracted the interest of the data and knowledge engineering community (Rahm & Bernstein, 2001). Specifically, most research on schema matching refers to a specific context (e.g., relational to object-oriented schema, relational to XML [extensible markup language] schema). In recent years, very intense research has also been performed in the filed of ontology mapping (Choi, Song, & Han, 2006). In the following paragraphs we survey the most closely related works to the issue of relational to ontology matching.

In Stojanovic, Stojanovic, and Volz (2002), the authors reported a semiautomatic tool, named KAON reverse, for migrating data from a relational database schema to an ontological one through a reverse engineering approach. Such schema matching is based on fixed rules, which are defined manually by users. The ontology used by the KAON reverse tool uses the RDF (resource description framework; Antoniou & van Harmelen, 2001) conceptual modeling.

Clio (Miller, Haas, & Hernandez, 2000) is a semiautomatic schema mapping tool that compiles high-level mappings into a low-level representation. Such tool supports the creation of mappings between two relational schemata or between a relational and an XML schema.

COMA++ (Aumueller, Do, Massmann, & Rahm, 2005) is an extension of the COMA tool (Do & Rahm. 2002). COMA++ combines different matchmaking algorithms. Moreover, it provides the user with the ability to compose, merge, and reuse existing mappings.

MapOnto (An, Borgida, & Mylopoulos, 2005) is an ongoing tool based on semantic mappings between database schemata and ontologies as well as between different database schemata. Such a tool produces a set of logical formulae in terms of either Horn clauses (representing mappings between database schemata and ontologies) or more complex expressions (representing mappings between different database schemata).

Similarity Measures

The RONTO methodology for schema matching is mainly based on similarity measures. Similarity may be considered as a measure denoting to which degree elements of different schemata convey equivalent meanings. The proposed methodology is based on different kinds of similarity measures (linguistic and semantics based) in order to effectively perform schema matching.

Definition 5. Linguistic similarity, sim_L, between two labels (i.e., words) that belong to an alphabet L is the quantification of the lexical comparison of these labels, defined as follows.

$$\mathrm{sim}_L : L \times L \rightarrow [0,1] \qquad (2)$$

Hence, for the labels $a, b \in L$, the linguistic similarity $\mathrm{sim}_L(a, b)$ assumes values into the [0,1] interval. Specifically, a value of 0 means that *a* and *b* labels are exactly different words, whilst a value of 1 represents word equivalence. The most common metric for linguistic similarity is the edit distance (Levenshtein, 1966). Such metric computes the minimum number of token insertions, deletions, and substitutions in order to transform one label into another.

We emphasized on calculating the semantic similarity between the elements of the two schemata since the goal of the schema matching process is to find mappings between semantically similar elements. The need for the semantic similarity is amplified by the fact that the RDB and ONT schemata might have been created from diverse domain experts, thus, different words might have been used for presenting the same meaning. Several techniques, based on the lexical database WordNet, have been proposed in order to compute

Figure 1. F2OM algorithm

Algorithm: ForeignKeys2Object-propertiesMapping (F2OM)
Input:
OP: all object-properties of the ontology
CCS: a vector with all CCS_c for each concept c
θ: a threshold for the degree of similarity, $\theta \in [0,1]$
Output: COPS /*a vector of all $COPS_p$ for each object-property p*/
For each object property $p \in$ OP **do**
  Let D be the domain of p
  Let R be the range of p
  For each candidate concept $c \in CCS_D$ **do**
    Let FK be the set of all foreign keys of c
    For each foreign key $f \in$ FK **do**
      If (sim(p, f) ≥ θ **AND** the referenced table of $f \in CCS_R$)
        then $COPS_p \leftarrow COPS_p \cup \{f\}$
    End for
  End for
  Add $COPS_p$ in COPS
End for
Return COPS

the semantic similarity between two elements (Pedersen, Patwardhan, & Michelizzi, 2004).

Definition 6. Semantic similarity, sim_S, between two words that belong to an alphabet L is the quantification of the semantic comparison of these words defined as follows.

$$sim_S : L \times L \rightarrow [0,1] \quad (3)$$

A value of 0 means that the two words do not have equivalent meaning, whereas a value of 1 means that they describe the same concept.

The compound similarity between two schema elements, $a \in$ RDB and $b \in$ ONT in Equation 4 is defined as the linear combination of the linguistic similarity and semantic similarity.

$$sim(a, b) = k \cdot sim_L(f_L(a), f_L(b)) + m \cdot sim_S(f_S(a), f_S(b)), \quad (4)$$

where f_L: $S_{RDB} \cup S_{ONT} \rightarrow L$ is the linguistic description of an element and f_S: $S_{RDB} \cup S_{ONT} \rightarrow L$ is the semantic description of an element. S_{RDB} and S_{ONT} are the set of elements of the RDB and ONT, respectively. Moreover, $k, m \in [0, 1]$, ($k + m = 1$), defined by the domain experts, indicate the relative importance of linguistic similarity over semantic similarity.

SCHEMA MATCHING PROCESS

The proposed schema matching methodology consists of a set of algorithms that perform the semantic mapping between the elements of a source database schema and their analogue elements of a target ontology. The following paragraphs describe these algorithms.

Tables-to-Concepts Mapping

The first algorithm that is implemented in RONTO is the tables-to-concepts mapping algorithm. Linguistic and semantic similarity measures are used in order to find all the CCs for each concept $c \in S_{ONT}$. Note that the database relations representing N:M relationships should not be mapped to concepts and are, thus, excluded from this mapping phase.

Attributes-to-Datatype-Properties Mapping

The proposed methodology proceeds with the computation of the mappings between the relation attributes and the datatype-properties of the ontology. Specifically, we try to match all datatype-properties *dp* of a concept *C* (i.e., the domain of *dp* is *C* or a superconcept of *C*) with the attributes of the candidate concepts of *C*. It should be noted that the auto-increment fields are excluded from such an algorithm since they convey no semantics. Foreign keys are also excluded from this step. In order to implement this matching phase, we defined mappings between the SQL (structured query language) data types of the RDB attributes and the XML schema data types (i.e., range) of the datatype-properties.

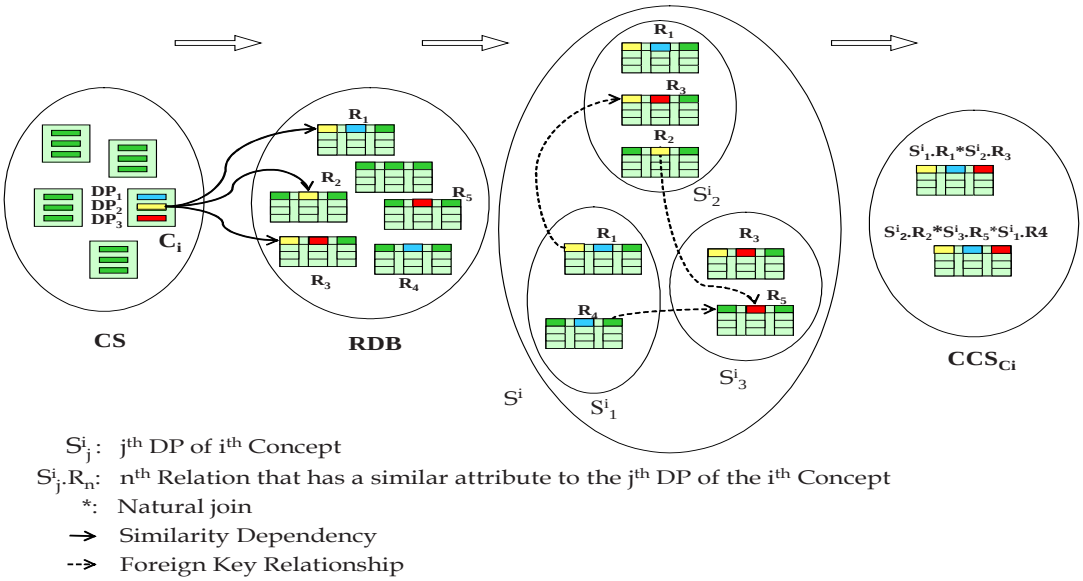
Foreign-Keys-to-Object-Properties Mapping

According to the OWL-DL, an object-property expresses a binary relationship between two concepts of an ontology. In relational databases, relationships among tables are expressed through

Figure 2. J2CM algorithm

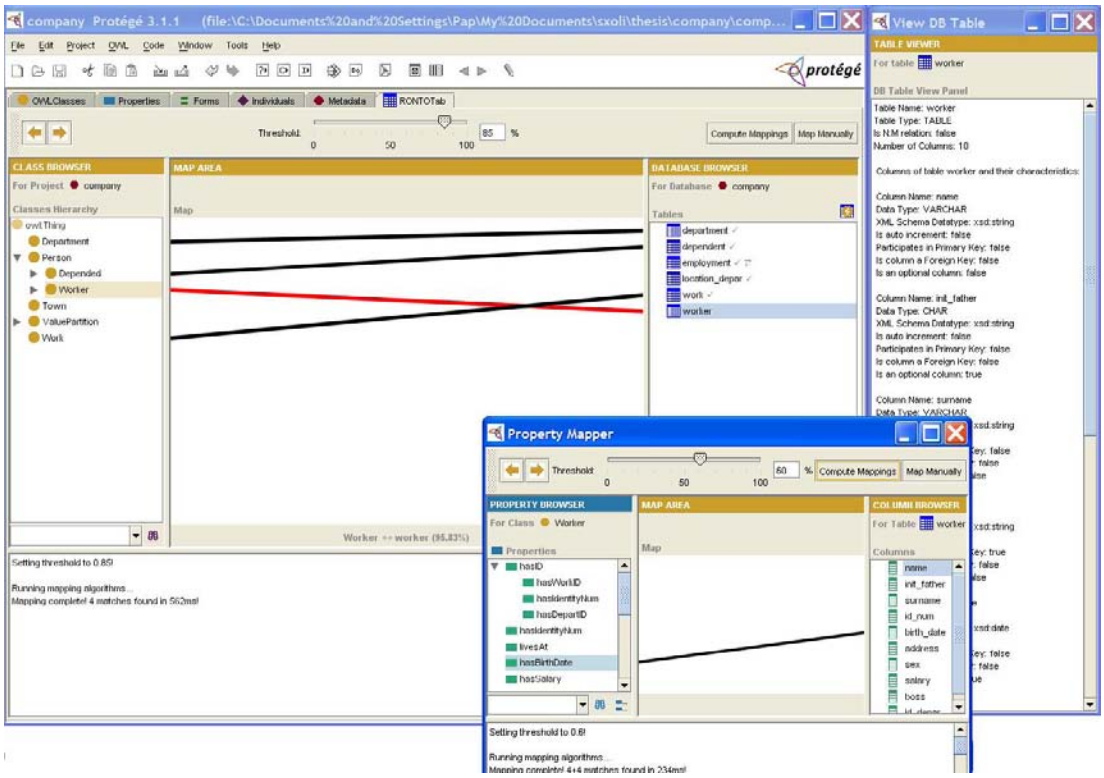
Algorithm: JoinedTables2ConceptsMapping (J2CM)
Input:
C: all concepts of the ontology
T: all tables of the database which participate in referential constraint
Output: CCS /*a vector with the CCS_c for each concept c*/
Begin:
For each concept $c_i \in C$ **do** /*Formation Step*/
  For each datatype-property d_j of c_i **do**
    Set $S^i_j = \{s \in T \mid \exists.attr \in s \wedge [sim(attr, d_j) \geq \theta]\}$
  End For
  Let CCS_{ci} be all possible natural joins between $s \in S^i_j$ and $\rho \in S^i_k$ such that $k \neq j$, $\pi \neq \rho$.
End For
For each concept $c_i \in C$ **do** /*Refinement Step*/
  For each object-property o_j of c_i **do**
    Let D be the domain of o_j /*D = c_i*/
    Let R be the range of o_j
    Do
      $CCS_D \leftarrow CCS_D \setminus \{c \in CCS_D \mid c$ does not contain foreign keys referencing to any element $b \in CCS_R\}$
      $CCS_R \leftarrow CCS_R \setminus \{b \in CCS_R \mid b$ does not contain primary keys being referenced by any element $c \in CCS_D\}$
    Until (all $CC_D \in CCS_D$ contain foreign key referencing to an element of CCS_R) **and**
        (all $CC_R \in CCS_R$ contain primary key being referenced by an element of CCS_D)
End For
Add CCS_{ci} in CCS
End For
Return CCS

Figure 3. J2CM formation step



S^i_j: j^{th} DP of i^{th} Concept
$S^i_j.R_n$: n^{th} Relation that has a similar attribute to the j^{th} DP of the i^{th} Concept
*: Natural join
→ Similarity Dependency
--» Foreign Key Relationship

Figure 4. Screenshot of RONTO plug-in



referential constraints, represented by foreign keys. The subprocess presented in Figure 1 defines mappings between such database elements and the object-properties of the ontology.

N:M-Relations-to-Object-Properties Mapping

Except from the referential constraints, there are database relations that represent binary relationships between two tables. Such relations may constitute the COPs for the object-properties of the ontology. In order to decide whether a relation t, with A(t) the set of its attributes, represents an N:M relationship between two relations t_1 and t_2, such that $t_1 \neq t$ and $t_2 \neq t_1$, the following conditions must be satisfied.

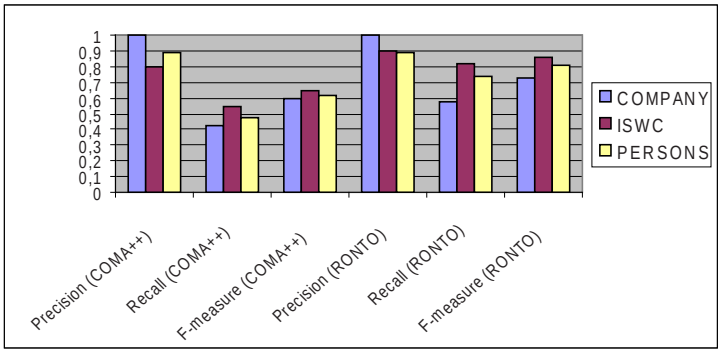
- The A(t) set contains at least two attributes a_1 and a_2, which comprise the primary key of t.
- The attribute a_1 is a foreign-key referencing table t_1.
- The attribute a_2 is a foreign-key referencing table t_2.

Then, a mapping between an object property *op* and table *t* can be defined if the domain of *op* is mapped to *t*1 and the range of *op* is mapped to *t*2, or vice versa.

Joined-Tables-to-Concepts Mapping

There are some cases in which the information content carried by one concept is distributed in more than one relations of the database. Therefore, it is important to exploit more complex combinations (joins and projections) between the database relations in order to find the appropriate CCs of such concepts. RONTO computes all the possible joins between the database relations and uses the following algorithm (see Figure 2) to perform the so-called complex concept mapping. This is also illustrated in Figure 3.

Figure 5. Evaluation of RONTO and Comparison to COMA++



In the first step (i.e., the CCS formation step, see Figure 3), the algorithm clusters the database relations that have an attribute similar to a datatype-property of a specific concept *c*. Next, it computes all possible joins between different relations from different sets (i.e., CCS_c). During the second step, the algorithm eliminates, for each object-property *p* of concept *c* with range *R*, all the CCs from the CCS_c that do not have a foreign key referencing the primary key of a CC that belongs to the CCS_R. The same refinement is also applied to CCs from CCS_R that do not contain a primary key referenced by a foreign key from a CC of CCS_c. In case *R* consists of more than one concept, the refinement step is properly modified.

Implementation Issues

RONTO is a tool that is deployed as a Protégé plug-in. Protégé (Gennari et al., 2002) is the most popular open-source ontology editor with a very large community of users and developers. It adopts a modular architecture that enables the deployment of tools as plug-ins developed by third parties.

The mapping is performed under human supervision through a friendly graphical user interface. Figure 4 depicts a prototype version of the plug-in. As previously stated, the mapping process involves, among others, a variety of semantic and linguistic similarity measurements between ontology and database terms. They can also, at every turn, ask for information concerning the database including all the possible joins between the database relations. RONTO performs the schema matching step by step, giving users the time to view the results, make their modifications (i.e., add a mapping that is not detected or delete some), or change the threshold and the similarity algorithms before continuing to the next step. The results produced by RONTO can be stored either in a proprietary or D2R map format. D2R map (Bizer, 2003) is a declarative language that describes mappings between relational database schemata and OWL/RDFS ontologies.

Evaluation

In order to evaluate the effectiveness of the RONTO tool, we used some data sets[2] to compare our proposed methodology with the one used by COMA++. Actually, we have chosen the latter tool because it is freely available and performs the schema matching with high performance. The mappings derived by the two matchers were compared with the expert mappings that were obtained by performing the task manually. The evaluation was based on common metrics such as precision, recall, and F-measure as defined in Do and Rahm (2002).

Regarding the three data sets used, COMPANY, ISWC, and ONTOTEST, the first one was created from scratch, while the other two were borrowed from other sources. ISWC is a data set used for describing the information of the International Semantic Web Conference and is available on

the Web, while ONTOTEST was obtained from the KAONReverse test suite.

In order to assess RONTO effectiveness, we used a subset of the available similarity measures. Specifically, we used a combination of Jaro (Jaro, 1989), the dice coefficient (Frakes & Baeza-Yates, 1992), and Needleman-Wunch (Needleman & Wunch, 1970) metrics. Concerning COMA++, we used a combination of the NamePath, Leaves, Siblings, and Parents matchers since composite matchers, according to the tool developers, return better results than the single ones.

Both tools gave high precision values (see Figure 5), which indicates that most of the derived mappings between the elements of the schemata were correct. RONTO has also high recall values, especially in large schemata (e.g., ISWC) and schemata with elements that have high degree of semantic similarity (e.g., COMPANY). The key of the matching process in RONTO is the right combination of the linguistic and similarity algorithms. Such combination leads to very good results when using a high threshold value (here, threshold=0.7). Specifically, when the labels between two schema elements $a \in$ RDB and $b \in$ ONT were invalid words (i.e., $sim_S(f_S(a), f_S(b)) = 0$) the total similarity of a and b was based only on linguistic similarity. In other cases (i.e., $sim_S(f_S(a), f_S(b)) \neq 0$) the schema matching process was performed by setting the constants of Equation 4 to $k = 0.3$, $m = 0.7$. The use of semantic similarity algorithms based on WordNet is a main contribution of RONTO methodology, which results in quite effective matching, especially when the two schemata consist of different, but with similar meaning, words. Moreover, the fact that RONTO provides the users with the ability to choose the appropriate set of the similarity algorithms for each step of the matching process enhances the performance of the proposed methodology.

CONCLUSION

We have presented in detail a new methodology for schema matching that addresses in a realistic way the requirement for actual data in the SW. The main contributions of the methodology is the exploitation of a variety of linguistic and semantic similarity techniques providing users with the ability to choose which of them they want to use and how to combine them. There are a lot of open issues in schema matching approaches, especially when the schemata are very different. One of the greatest challenges is the exploitation of complex database attributes since so far only atomic database fields have been considered. Moreover, a database may contain relations that represent n-ary relationships between tables (where $n>2$). It would be very useful if we could, in an automatic manner, identify such relationships and map them to ontological properties. Finally, one could exploit the ontology hierarchy and the cardinality constraints in the two schemata.

REFERENCES

An, Y., Borgida, A., & Mylopoulos, J. (2005). Constructing semantic mappings between XML data and ontologies. *ISWC'05*.

Antoniou, G., & van Harmelen, F. (2004 April). *A semantic Web primer.* The MIT Press.

Aumueller, D., Do, H., Massmann, S., & Rahm, E. (2005). Schema and ontology matching with COMA++. *Proceedings of SIGMOD.*

Berners-Lee, T., Hendler, J., & Lassila, O. (2001, May). *The semantic Web.* Scientific American.

Bizer, C. (2003). *D2R MAP: A database to RDF mapping language.* Poster at the 12th World Wide Web Conference, Budapest, Hungary.

Choi, N., Song, I. Y., & Han, H. (2006). A survey on ontology mapping. *ACM SIGMOD Record, 35*(3), 34-41.

Do, H., & Rahm, E. (2002). *COMA: A system for flexible combination of schema matching approach.* Proceedings of the 28th VLDB Conference, Hong Kong, China.

Frakes, W., & Baeza-Yates, R. (1992). *Information retrieval: Data structures & algorithms.* Prentice Hall.

Gennari, J., Musen, M. A., Fergerson, R. W., Grosso, W. E., Crubezy, M., Eriksson, H., et al. (2002). The evolution of protege: An environment for knowledge-based systems development. *International Journal of Human-Computer Studies.*

Jaro, M. A. (1989). Advances in record linking methodology as applied to the 1985 census of Tampa Florida. *Journal of the American Statistical Society, 64*, 1183-1210.

Levenshtein, V. (1966). Binary codes capable of correcting deletions, insertions and reversals. *Soviet Physics: Doklady, 10*(10), 707-710.

Miller, R. J., Haas, L. M., & Hernandez, M. L. (2000). *Schema mapping as query discovery.* Proceedings of the 26th VLDB Conference, Cairo, Egypt.

Needleman, S. B., & Wunch, C. D. (1970). Needleman-Wunch algorithm for sequence similarity searches. *Journal of Molecular Biology, 48*, 443-453.

Pedersen, T., Patwardhan, S., & Michelizzi, J. (2004). Wordnet::similarity: Measuring the relatedness of concepts. *Proceedings of Fifth Annual Meeting of the North American Chapter of the ACL (NAACL-04).*

Rahm, E., & Bernstein, P.A. (2001). A survey of approaches to automatic schema matching. *VLDB Journal, 10*(4), 334-350.

Staab, S., & Studer, R. (2004). *Handbook on ontologies* (International handbooks on information systems). Springer.

Stojanovic, N., Stojanovic, L., & Volz, R. (2002). A reverse engineering approach for migrating data-intensive Web sites to the semantic Web. *Proceedings of the Conference on Intelligent Information Processing.*

KEY TERMS

Data Migration: The process of translating data from one format to another.

Ontology: A model (usually logic based) representing the main entities and their relationships within a domain of discourse.

Ontology Population: The process of creating instances for an ontology. This process usually involves linking various data sources to the elements of the ontology.

RONTO: A schema-matching methodology and tool that facilitates ontology population from relational data through a schema matching process.

Schema Matching: The process of matching elements of a source data model (i.e., schema) with elements of a target data model.

Semantic Web: The evolution of the current World Wide Web in a way that it is also machine understandable in addition to being human understandable.

Similarity Measure: An algorithmic method that calculates the degree of some aspect of similarity between two entities.

ENDNOTES

1 We assume columns that are not foreign keys.

2 The data sets can be found at http://p-comp.di.uoa.gr/projects/ronto/index.html

Chapter XLVIII
Ontology-Based Semantic Models for Databases

László Kovács
University of Miskolc, Hungary

Péter Barabás
University of Miskolc, Hungary

Tibor Répási
University of Miskolc, Hungary

INTRODUCTION

A key characteristic of database systems is the layered structure and the accomplished independencies as is defined in the ANSI SPARC database reference model. This level-wise approach is also applied in the database development processes. According to the design triangle, the problem area to be mapped into the database is described by a human-oriented description at the initial phase of the design activity. Semantic data model (SDM) as a design tool uses concept-level elements in contrast to database schemas that are models at the logical and physical levels. The main role of semantic models is that they provide an abstract approach; they are easy to understand and they provide database independence.

There are different application areas for semantic models. An SDM can be used for example in

- database schema design,
- knowledge transfer,
- database integration,
- schema validation, and
- knowledge representation.

In the history of database systems, several SDM models were proposed and used, such as the ER, IFO, ODL or UML models. Most of these models are strongly related to the underlying logical models and can not provide the required independency and generality. A new approach to create an efficient SDM for databases is the application of ontology-based description. According to our experiences, many experts on database management field are not aware of meaning of ontology. The aim of this article is to show and to explain the importance and role of ontology in design processes.

SEMANTIC MODELS

With a formal definition (CROSI, 2005), the database schema model Λ can be given as a tuple $\Lambda(\Sigma_\Lambda, P_\Lambda)$, where Σ_Λ is the structure (or signature) and P_Λ is the integrity component. The Σ_Λ component is built up from predefined structure elements. Taking the relational model as an example for Λ, Σ_Λ contains the following elements: field (attribute), relation and database signatures. For a given Σ_Λ, the set of corresponding schema instances is denoted by $I_{\Sigma\Lambda}$. An instance of a relational schema signature is usually given as a set of relation instances. The P_Λ component contains elements to define integrity constraints on the $I_{\Sigma\Lambda}$ set. An integrity rule is used to restrict the valid, permitted schema signature instances.

One of the earliest and most widely used SDM tools are the Entity-Relation (ER) and the Extended Entity-Relation (EER) models (Chen, 1976; Date, 1995). The main building blocks are entities, attributes and relationships. Entities can be considered as high-level structured concepts while attributes are the low-level atomic, or structured concepts. The EER model distinguishes three types of relationships: specialization among entities, encapsulation among entities and association between entities or attributes. In contrast to the structure oriented approach of EER, functional semantic models, like the IFO (Abiteboul, 1987) model, are based on global conceptualization. The IFO model provides a higher level of abstraction as it neglects logical schema elements like cardinality and some other integrity constraints. The ODMG ODL (Cattell, 1997) model is used to describe object

Table 1. Semantic data models

Semantic Model	Signature elements	Integrity elements
EER	Entity Attribute Structure Relationship (association) Specialization	Key Cardinality of association Multiplicity of values
IFO	Concept (entity) Printable element (attribute) Structure Association (function) Specialization	Key Cardinality of association
UML	Class (entity) Data type Field (attributes) Structure References (association) Methods Specialization	Key Cardinality of association Visibility of the elements

Table 2. Logical data models

Logical Model	Signature elements	Integrity elements
Relational	Data type Field Relation Database	Domain Primary key Foreign key Unique field Not null field Check constraint
Object-relational	Data type Field UDT, structure Methods Relation Reference type, locator Specialization Containment Database	Domain User-defined domain Primary key (OID) Not null Unique Check constraint Foreign key (REF)
XML-Schema	Data type Element Attribute Structure Hierarchy	Domain Multiplicity of values Primary Key Foreign key Unique Not null Pattern constraint

databases in object-oriented environment. ODL provides a modeling language near to the logical level with many details related to OO concepts. One of the most widely used modeling language is the UML which provides a general purpose, object-oriented tool set to describe the different aspects of objects and classes. UML is a set of modeling components among which the class diagram can be mentioned as the main schema language. Considering the most frequently used data management oriented semantic and logical models, the following signature and integrity elements comprise these models. The first table is related to the semantic data models and the second table describes the comparison of logical data models.

During the design process, the conversion of a semantic model into a logical model is a usual step. Usually, the initial semantic model has a smaller signature set and has less functionality in integrity elements. On the other hand, logical models contain extra elements related to the physical implementation level. During the transformation

- some new elements are added to the target model, and
- some source elements are eliminated from the target model.

Table 3. Conversion rules

EER	XML-Schema
Entity	Element
Simple attribute	Simple Type
Key	Key
Compound attribute	Complex Type
Multiplicity of values	Multiplicity of values
1:N relationship	Key and Keyref
N:M relationship	Element + Key, Keyref
Mandatory relationship	Not null + Key, Keyref
Specialization	Specialization

Taking for example the EER → XML-Schema transformation, the rules given in Table 3 can be applied.

Considering the traditional SDM models, the transformation of the semantic model into a logical model has the following difficulties that can not be solved without loss of information:

- missing integrity constraints of the problem domain,
- the generated models are always local models due to cost reduction,
- heterogeneity of model languages.

A very promising approach in this field is the application of ontology in semantic modeling.

ONTOLOGY-BASED SEMANTIC MODELING

It is widely assumed that every data model at a high level of information processing must be based on ontology. The term ontology can be defined as an explicit "specification of conceptualization" (Gruber, 1993). The ontology defines a common set of concepts and terms that are used to describe and represent a domain of knowledge. The ontology is used to

- define the concepts, relationships and other distinctions that are relevant for modeling a domain,
- specify the definitions of representational vocabulary which provides meaning and constraints on the usage.

The key element of ontology is a methodology that determines what the primitives of the conceptualization are and how to determine these primitives from the domain. Based on (Erdman, 2001), an ontology can be given formally with a quintuple *O(C, A, isA, aType, F)* where

- C : is a set of concepts (classes),
- A : is a set of attributes for concepts in C,
- isA $\subseteq C \times C$: is the specialization relationship among concepts,
- aType $\subseteq C \times A \times C$: is the type signature of attributes,
- F : are formulas in predicate logic to describe the rules of concept management.

The set of logic formulas has two components: the first is the set of universal rules and the second is the domain specific rules. It is widely accepted that the set of universal rules should include the following elements:

- $\forall\ c \in C$: isA(c,c),
- $\forall\ c_1,c_2,c_3 \in C$: isA($c_1$,$c_2$), isA($c_2$,$c_3$) $\Rightarrow$ isA(c_1,c_3),
- $\forall\ c_1,c_2 \in C$, $a \in A$: isA(c_1,c_2), aType(c_2,a) $\Rightarrow$ aType(c_1,a).

In some ontology models, the ontology contains, beside the concepts, also objects as separate elements that can be given as

- D: a set of objects with a new relation, and
- instanceOf $\subseteq D \times C$.

In this case, the following universal rule is also part of the rule system:

- $\forall\ c_1,c_2 \in C$, $o \in D$: isA(c_1,c_2), instanceOf(o,c_1) $\Rightarrow$ instanceOf(o,c_2).

Comparing the component list of ontology with the elements of semantic models, it can be seen that there is a large overlap between the two sets. Regarding the signature part of semantic models, all of the structure elements are present in the ontology model, too. The main differences relate to the following two points:

- semantic models do not contain a logic language to perform reasoning and validation on the model, while
- ontology model does not contain data management oriented elements, like primary key.

The formal logic component is an important extension in ontology as it provides a tool to verify the correctness of a model instance and it can be used to improve the functionality of information retrieval. Deductive database systems can be referred here as a theoretical foundation of this approach. This functionality emphasis another key differentiating property of ontology, namely ontology is an active metadata component that is used during the normal operation of the system. On the other hand, semantic data models are passive metadata as they are used usually only in the design phase of the data system.

Based on the previous considerations, the ontology model provides more functionality than semantic models but it misses some implementation-oriented elements on data management. The usage of ontology as a general semantic model has several benefits, where the followings can be stressed as key aspects:

- it provides global knowledge background,
- it is a sound, consistent model,
- it can be used for complex validation tasks,
- it supports data management operations,
- it has a standard description format,
- it is more flexible and extensible than the traditional semantic data models.

Based on this approach, the role of the ontology in database design can be given as a source model that will be converted into a logical model during the design phase. The main steps of design can be summarized as follows:

1. Creation of upper ontology as a base.
2. Extension with domain ontology.
3. Extension with data management specific elements.
4. Generation of the logical data model.
5. Coding the data definition and manipulation commands.

The domain-level ontology refers to an ontology that covers only the concept of an application area. This ontology is usually called lower ontology. On the other hand, there is a need to integrate these domain ontologies. The term of upper ontology refers to the conceptualization of more abstract concepts that are common in all lower ontologies.

In (Erdman, 2001) a prototype system is presented where the target model is the XML-DTD model. The XML-DTD has similar role as the XML-Schema but it has a simpler functionality. In the project a DTDMaker module generates a DTD schema description from the ontology description presented in F-logic.

The investigation of the relationship between ontology and traditional semantic modeling in general appeared first more than a decade ago. These investigations are usually related to the UML model. In some approaches, the UML SDM is proposed to be used as an ontology language. The main problem of this direction is that the UML can not provide the required precision and reasoning capability. In (Guizzardi, 2002), the GOL ontology language is used to evaluate the ontological correctness of the UML class model and provides guidelines how UML can be used in conceptual modeling. Beside the traditional SDM models, some new SDM models are also proposed in the literature to capture the conceptualization of the problem domain. The paper of (Motik, 2002) presents a new integrated conceptual model based on the formal logic formalism. The model language is implemented with the KAON tool.

Another main direction in ontology based-modeling is the usage of ontology as a validation tool. In (Raatikka, 2002), a process is defined how to validate metadata with the attached ontology description. The related papers usually highlight three benefits of using ontology prior to semantic modeling: it enables better decision in selection of SDM; it makes the specific domain phenomena more clear; and it makes sense of ambiguous semantics.

Different description languages exist for ontology. One of the first languages for ontology is RDF (Lassila, 1999). RDF is used to describe the concepts in a neutral, machine-readable format. According to the specification, the basic language elements are resources, literals and statements. One of the most widely used formalism is the OWL language that is part of the W3C semantic standards. The OWL standard defines several OWL levels and one of the OWL variants is based on descriptive logic (DL) in order to provide a formal way of constraining and reasoning. DL provides a formal language for defining concepts and individuals (Baader, 2002; Calvanese, 1998). This representation form provides a precise inference framework for reasoning purposes.

FUTURE TRENDS

Ontologies are widely used to overcome the interoperability problem during the integration of heterogeneous information sources. The mapping phase within the integration process can be significantly improved by the application of ontology. An excellent and comprehensive survey on ontology mapping can be found in (Kalfoglou, 2003). The paper first give theoretical grounds for defining the different aspects of ontology merging. It includes analysis of the key technologies, like mediators, matching, translators, algebraic and IF-based methods. Based on the diversities of approaches and of difficulties, the ontology merging is a great challenge and the paper concludes that full automation is not feasible. Euzenat and Shvaiko book (Euzenat, 2007) is devoted to a special problem in ontology merging, namely to ontology matching. Ontology matching finds correspondences and similarities between different ontology models. The book provides a uniform framework on the research efforts in this area and it presents the methods (including probabilistic methods) and tools for similarity matching. In the paper (Dou, 2006) on schema integration, the relational database schema is converted first into an ontology model. In the next step, the missing relationships between the concepts of different domains are built up. This step is done by human experts. The control of schema conversion is based on the generated mapping ontology. The integration model was implemented with a tool named OntoGrate where the ontology is defined in first order logic. A very similar approach is presented in (DuChjarme, 2006), where the ontology for mapping is given in RDF/OWL syntax. In (Ram, 2004) a common ontology called Semantic Conflict Resolution Ontology (SCROL) is proposed that addresses the inherent difficulties in the conventional approaches, i.e., in the federated schema and domain ontology approaches. The proposed SCROL system provides a systematic method for automatically detecting and resolving various semantic conflicts in heterogeneous databases. The (Shanks, 2003) presents an application framework based on ontology for validation of conceptual models.

Not only the schema-level operation, but the database query operations can also benefit from the attached ontology model. The accompanying metadata in databases increases in every new version of DBMS. Metadata as a special kind of ontology can be used among others for reformulate the incoming query operations. In (Freytag, 2004) a simplified ontology model is applied for this purpose where the ontology model contains three basic types of relationships: specialization, subpart and synonym. Using ontology the objects and operators given in the query can be converted

into equivalent objects and operators with lower execution costs. The application of ontology may appear directly at the user interface level, too. In the current Oracle DBMS version, some new query operators are presented that provide a tool for ontology-based semantic matching. The proposed operators for the new matching mechanism can be used in SQL SELECT commands. The operators are defined on character data and the distance of strings is calculated from an ontology description. The kernel of the ontology is a taxonomy based on the specialization relationship of concepts. The ONT_RELATED operator determines if the argument terms are related according to the underlying ontology model. A sample query for semantic matching has the following form:

*SELECT * FROM employees WHERE ONT_RELATED (job,'IS_A','clerk','Ontology') = 1*

The proposed formalism and functionality is strongly related to the text mining module of the Oracle DBMS.

There exist several domains where ontology-based applications were developed and tested in the recent years. Among the first application fields, the clinical information systems, geographic information systems, biologic information systems and integrated knowledge management can be mentioned. Beside these mentioned fields, for further details on current applications of ontology, we refer to (CROSI, 2005).

CONCLUSION

The ontology is used to define concepts, relationships and other distinctions that are relevant for modeling a domain where the specification takes the form of the definitions of representational vocabulary which provides meaning and constraints on the usage. Ontology is getting more and more popular as the description formalism for knowledge representation. There are many benefits of an ontology-based semantic data modeling over the traditional models. Ontology languages provide a universal, general description in standardized formalism. It has a logic foundation that supports reasoning and validation of the model. The ontology can be used as a general and extended semantic model in database design. On the other hand, it supports model validation, schema integration and extends the functionality of information retrieval.

REFERENCES

Abiteboul, S, & Hull, R. (1987). IFO: A Formal Semantic Database Model. *ACM Trans. Database Syst., 12*(4), 525-565.

Baader, F., & Nutt W. (2002). Basic Description Logics. In F. Baader, D. Calvanese, D. L. McGuinness, D. Nardi, & P. F. Patel-Schneider (Eds.), *The Description Logic Handbook* (pp. 47-100). Cambridge University Press.

Calvanese, C., Lenzerini, M., & Nardi, D. (1998). Description Logics for Conceptual Data Modelling. In J. Chomicki & G. Saake (Eds.), *Logics for Databases and Information Systems* (pp. 229-263). Kluwer.

Cattell, R. G. G. et al. (1997). *Object Database Standard: Odmg 2.0*, R. G. G. Cattell, D. K. Barry, D. Bartels, M. Berler, J. Eastman, S. Gamerman, D. Jordan, A. Springer, H. Strickland & D. Wade (Eds.). Morgan Kaufmann Series in Data Management Systems.

Chen, P. (1976). The entity-relationship model-Toward a unified view of data. *ACM Transaction on Database Systems, 1*(1), 9-36.

CROSI (2005). *Capturing Representing and Operationalising Semantic Integration*. University of Southampton and Hewlett Packard Laboratories

Date, J. (1995). *An Introduction to Database Systems*. Addison Wesley.

DuChjarme (2006). Relational database integration with RDF/OWL, http://2006.xmlconference.org/programme/presentations/188.html

Dou, D, & LePendu, P. (2006). Ontology-based integration for relational databases. *Proc. Of 2006 ACM Symposium on Applied computing*, (pp. 461-466).

Erdmann, M. (2001). *Ontologien zur konzeptuellen Modllierung der Semantik von XML*. PhD dissertation,iversity of Karlsruhe.

Euzenat, J., & Shvaiko, P. (2007). *Ontology matching.* Springer Verlag

Freytag J., & Necib, C. (2004). Using ontologies for database query reformulation. *Proc. ADBIS Conference*.

Gruber, T. (1993). Towards principles for the design of ontologies used for knowledge sharing. *Proc of International Workshop on Formal Ontology*, Padova, Italy.

Guizzardi, G., Herre, H., & Wagner, G. (2002). *Towards Ontological Foundations for UML conceptual models.* ODBASE 2002.

Kalfoglou, Y., & Schorlemmer, M. (2003). Ontology mapping: the state of the art. *Knowledge Engineering Review, 18,* 1-31.

Lassila, O. (1999). Resource Description Framework. *Proc in XML'99 Conference*, Philadelphia.

Motik, B., Maedche, A., & Volz, R. (2002). *Conceptual Modeling Approach for Semantics-Driven Enterprise Applications.* ODBASE 2002.

Raatikka, V., & Hyvonen, E. (2002). *Ontology-based semantic metadata validation.*

Ram, S., & Park, J. (2004)**.** Semantic Conflict Resolution Ontology (SCROL): An Ontology for Detecting and Resolving Data and Schema-Level Semantic. *IEEE Transaction on Knowledge and Data Engineering, 16*(2), 189-202.

Shanks, G., Tansley, E., & Weber, R. (2003). Using ontology to validate conceptual models. *Communication of ACM, 46*(10), 85-89.

KEY TERMS

Ontology Merging: The integration of heterogeneous ontology information sources. Due to complexity of integration, it usually includes several distinct processes based on heuristic approach. The main components of integration are: ontology mediation, ontology matching and ontology translation.

Ontology: An explicit specification of conceptualization. A semantic data model that describes the concepts and their relationships. It contains a controlled vocabulary and a grammar for using the vocabulary terms. The ontology enables to make queries and assertions and reasoning. The most popular form to describe ontology is RDF and OWL.

OWL: A language to describe web-ontologies. It uses an XML format and it contains a formal description logic component, too. It provides the following base functionalities: classification, type and cardinality constraints, thesauri, decidability.

RDF: A semantic data model that describes the world with elementary statements. A statement is a triplet having the following form: subject – predicate – object.

Semantic Data Model: A design tool for databases that uses concept-level language elements. The main role of semantic models is that they can provide an abstract approach; they are easy to understand and they provide database independence.

Semantic Network: A graph for knowledge representation where concepts are represented as nodes in a graph and the binary semantic rela-

tions between the concepts are represented by named and directed edges between the nodes. All semantic networks have a declarative graphical representation that can be used either to represent knowledge or to support automated systems for reasoning about knowledge.

Semantic Operators: Operators that are based on the semantic content of the text. Specialization and synonyms are base examples of semantic operators.

Chapter XLIX
Inconsistency, Logic Databases, and Ontologies

José A. Alonso-Jiménez
Departamento de Ciencias de la Computación e Inteligencia Artificial Universidad de Sevilla, Spain

Joaquín Borrego-Díaz
Departamento de Ciencias de la Computación e Inteligencia Artificial Universidad de Sevilla, Spain

Antonia M. Chávez-González
Departamento de Ciencias de la Computación e Inteligencia Artificial Universidad de Sevilla, Spain

INTRODUCTION

Nowadays, data management on the World Wide Web needs to consider very large knowledge databases (KDB). The larger is a KDB, the smaller the possibility of being consistent. Consistency in checking algorithms and systems fails to analyse very large KDBs, and so many have to work every day with inconsistent information.

Database revision—transformation of the KDB into another, consistent database—is a solution to this inconsistency, but the task is computationally untractable. Paraconsistent logics are also a useful option to work with inconsistent databases. These logics work on inconsistent KDBs but prohibit non desired inferences. From a philosophical (logical) point of view, the paraconsistent reasoning is a need that the self human discourse practices. From a computational, logical point of view, we need to design logical formalisms that allow us to extract useful information from an inconsistent database, taking into account diverse aspects of the semantics that are "attached" to deductive databases reasoning (see Table 1). The arrival of the semantic web (SW) will force the database users to work with a KDB that is expressed by

logic formulas with higher syntactic complexity than are classic logic databases.

BACKGROUND

Logic databases are based on the formalisms of first order logic (FOL); thus, they inherit a classical semantics that is based on models. Also, they can be interpreted within a proof–theoretic approach to logical consequence from the logic programming paradigm (Lloyd, 1987). The extended database semantics paradigm is developed to lay before the foundations of query-answering tasks and related questions (see Minker, 1999), but its aim is not to deal with inconsistencies. The data cleaning task may involve—in the framework of repairing logic databases—logical reasoning and automated theorem proving (Boskovitz, Goré, & Hegland, 2003).

On the other hand, new paradigms, such as SW, need new formalisms to reason about data. Description logics (DL) provide logic systems based on objects, concepts, and relationships, with which we can construct new concepts and relations for reasoning (Baader, Calvanese, McGuinnes, Nardi, & Patel- Schneider, 2003). Formally, DL are a subset of FOL, and the classical problems on consistency remains, but several sublogics of DL provide nice algorithms for reasoning services. The ontology web language (OWL; its DL-sublanguage) is a description logic language designed for automated reasoning, not only designed for the classical ask–tell paradigm. With languages such as OWL, ontologies exceed their traditional aspects (e.g., taxonomies and dictionaries) to be essential in frameworks as data integration.

The classical notion of inconsistency in databases mainly deals with the violation of integrity constraints. This notion must be expanded because of the new notion of logic databases in SW, in which ontologies and data both play the same role in knowledge management. Therefore, there are several sources of inconsistency (see Table 2). This role is not only limited to the database but also includes the verification and validation task of knowledge- based systems (Bench-Capon, 2001). Inconsistency arises in the initial steps of ontology building due to several reasons and not only by the updating of data. In general, the repair of a logic database involves the study of the soundness and perhaps completeness (i.e., the method output's only correct solutions and all the relevant solutions). Semantics would support reasoning services such as self-consistency, checking the relations between concepts (as subsumption), and classification of objects according to the ontology.

Systems exist in which both paradigms, classical and SW logic databases, are conciliated

Table 1. Semantics aspects to consider in logic databases

• Classical semantics for First Order Logic
Extended semantics for databases
Reiter's formalization of databases (Reiter, 1984). Closed World Assumption
Relations among a KDB, queries and integrity constraints
Expressive power of recursive definitions
Consistency checking versus intentional part of the KDB
Multivalued semantics
Contextualized semantics for ontologies or data

under the extension of the former (see, e.g., Pan & Heflin, 2003). However, the relation between DL and database models may not be fruitfully formalized because of the limited expressiveness of the DL system selected to make the reasoning feasible (see chapter 4 in Baader et al., 2003).

INCONSISTENCY HANDLING

Solutions that are suggested to work in the presence of inconsistences can be classified according to different views (see Table 3, where several references appear). The first aspect—and maybe the most important—is the compatibility between the original semantics of the KDB source and the logical formalism selected to handle inconsistency. From this point of view there exist paraconsistent logics that limit the inference power of FOL to avoid non desired answers and also modal logics for representing different aspects of the information sources. These approaches manage semantics that are essentially different from the semantics of KDBs. On the other side we can find methods that classify, order, or both, interesting subsets according to the original semantics of the KDB, such as the argumentative approach or the integration of data by fusion rules, but they do not repair the KDB. Other methods also exist that propose how the KDB should be revised (e.g., integrity

Table 2. A list of sources of inconsistencies from the practical knowledge management

Dirty data Some kinds of data dirtiness give rise to fail of integrity constraints (Kim, Choi, Hong, Kim and Lee, 2003).	**Deficient maintenance of KDB** The kind of the selected method for preserving consistence is not robust under every sort of updates.
Neglected development of the intentional database The development of the intentional component part of the database produces an inconsistent theory (the *ontology*, in the SW paradigm. See e.g. Backlawski, Kokar, Waldinger and Kogut 2002).	**Wrong data mining** The output of data mining systems does not satisfy integrity constraints or ontology requirements. In multiagent data mining, the different outputs lead to a problem of data integration.
Logical interpretation in data integration The design of a data integration system -to provide uniform access to multiple and heterogeneous information sources- needs of query reformulation, ontology mapping or integration and, in general, logical interpretation.	**Expressiveness clashes with Model Theory** The logical syntax/semantics (from deductive database paradigm) does not allow new features to be used in the usual knowledge representation in the SW. This absence implies messy definitions that may be incorrect.
Procedural incompleteness Incomplete query-answering algorithms do not produce any witness for some integrity constraint of existential character.	**Bad design of the common knowledge shared by different users** The intentional component part does not describe the users intended requirements. The logical consistent KDB does not fit with user's beliefs. Thus, new updates may produce inconsistencies.
Conflict information in data integration Special case in data integration: the information which is received from different consistent resources is inconsistent.	**Deficient Ontology learning** The ontology acquisition is a tedious task that the user tends to finish before he/she thinks as advisable. A poor ontology associated to consistent data may produce inconsistency.
Deficient specification of the ontology language The specification of the language for ontology representation is inconsistent (Fikes, McGuinnes and Waldinger 2002).	**Skolem Noise** A kind of dirty data produced by the use of an automated theorem prover in data cleaning of logic databases (Alonso, Borrego, Chávez, Gutiérrez and Navarro, 2003)
Inadequate data cleaning Some criteria to take desitions in data cleaning make inconsistent the KDB.	

constraints of the extensional database). However, it is necessary to point out that the automated knowledge revision is an essentially different task in the case of ontologies, because the ontology source represents a key organisation of the knowledge of the owner and, as in every logical theory, minor changes may produce unexpected and dangerous anomalies.

Another point of view concerns the share of the KDB that is repaired when an anomaly is found. According to this, the methods based on arguments mentioned earlier can be used to repair only the anomalous argument. Due to the high complexity of consistency checking algorithms, to preserve consistency under updates is a better option than repairing. In the case of evolving ontologies, new systems such as KAON infrastructure are needed (for more information, see http://kaon.semanticweb.org/kaon).

There are methods dealing with the enforcement of consistent answers (i.e., answers that satisfies integrity constraints) from inconsistent databases; it is done by transforming the self-query or by limiting the inference power of the system.

Table 3. A list of recent solutions for inconsistency handling

• Paraconsistent logics (Hunter, 1998 and Grant and Subrahmanian, 2000)
Non-repairing and merging-oriented techniques:
Pre-orders on information sets (Cantwell, 1998 or Marquis and Porquet, 2003 in the paraconsistent framework).
Argumentative hierachy (Elvang-Goransson and Hunter, 1995), argumentative frameworks (Dung, 1995) and databases (Pradhan, 2003; Huang et al. 2006).
Fusion rules (Bloch et al., 2001).
Merging databases (Cholvy and Moral, 2001).
Contextualizing ontologies (Bouquet, Giunchiglia, van Harmelen, Serafini and Stuckenschmidt, 2003) and data (MacGregor and Ko, 2003).
Measuring the anomalies:
Evaluating by means of a paraconsistent logic (Hunter, 2003).
Measuring inconsistent information (Knight, 2003).
Consistent interpretation of Skolem noise (Alonso et al., 2003).
Repairing techniques:
To apply knowledge reductions in inconsistent systems (Kryszkiewiccz, 2001).
Fellegi-Holt method (Boskovitz, Goré and Hegland, 2003).
Database repairs by tableaux method (Bertossi and Schwind, 2004).
Consistent querying to repair databases (Greco and Zumpano, 2000).
Consistent enforcement of the database by means of greatest consistent specializations (Link, 2003).
Consistent answering techniques without reparation:
Transfomation of the query to obtain consistent answers (Celle and Bertossi, 1994).
Consistent query answer in the presence of inconsistent databases (Greco and Zumpano, 2000)
To use bounded paraconsistent inference (see e.g. Marquis and Porquet, 2003).
Detecting the cause of the inconsistency and retrieving a subset of the original KB(Arieli and Avron, 1999).
Consistency preserving methods:
Consistency preserving updates in deductive databases (Mayol and Teniente, 2003).

Another method is to work in the context of data integration–merging (Levy, 2000). Fusion rules are the most direct treatment of simple information sets. The complex case, in which several ontologies come into play, can be solved by contextualizing the knowledge. The contextualization of ontologies is an extension of the classical method introduced by McCarthy, and it has been used in important ontology projects such as CyC (for more information, see http://www.cyc.com. The use of contexts prevents inconsistences and it allows to build coherent subsets of the ontology target.

Finally, there exist measures to *estimate inconsistency.* Although these measures may be unfeasible by their semantic oriented definitions, this obstacle may be partially solved by weakening metrics that estimate the cognitive difference between the ontology source and the ontology target by using only syntactic features (see, e.g., Gutiérrez-Naranjo, Alonso-Jiménez, & Borrego-Díaz, 2002; Hunter, 2003).

FUTURE TRENDS

To handle inconsistences in the semantic web, future trends must study verification techniques based on sound, limited testing and aided by a powerful automated theorem prover (see Alonso-Jiménez et al., 2003; Boskovitz et al., 2003; Huang et al. 2006). These techniques need a deep analysis of the behaviour of automated theorem provers having a great autonomy, because a slanted behaviour may produce deficient reports about inconsistencies in the KDB. The most promising research line in this field is the design and development of tools that allows us to explain the source of anomalies detected in ontologies.

CONCLUSION

Inconsistency handling has been a prevailing task in important fields as the semantic web, data integration, and data cleaning. Several techniques are proposed, but the need of working with very large databases makes some of them unfeasible, especially those that are applied on the full KDB. In any case, the inconsistency is a persistent matter in the semantic web field. Due to that, several research teams are studying how to manage and control it.

REFERENCES

Alonso-Jiménez, J. A., Borrego-Díaz, J., Chávez-González A., Gutiérrez-Naranjo M. A., & Navarro-Marín, J. D. (2003). Towards a practical argumentative reasoning in qualitative spatial databases. *Proceedings of the 16th International Conference on Industrial & Engineering Applications of Artificial Intelligence & Expert Systems (IEA/ AIE 2003), Lecture Notes in Computer Science, 2850* (pp. 789-798).

Ariel, O., & Avron, A. (1999). A model-theoretic approach for recovering consistent data from inconsistent knowledge bases. *Journal of Automated Reasoning, 22*(2), 263- 309.

Baader, F., Calvanese, D., McGuinness D., Nardi, D., & Patel-Schneider, P. (2003). *The description logic handbook*. Cambridge, UK: Cambridge University Press.

Baclawski, K., Kokar, M., Waldinger, R., & Kogut, P. (2002). Consistency checking of semantic web ontologies. *Proceedings of the First International Semantic Web Conference 2002 (ISWC'02), Lecture Notes in Computer Science, 2342* (pp. 454-459).

Bench-Capon, T. (2001). The role of ontologies in the verification & validation of knowledge-based systems. *International Journal of Artificial Intelligence, 16,* 377-390.

Bertossi, L. E., & Schind, C. (2004). Database repairs and analytic tableaux. *Annals of Mathematics and Artificial Intelligence, 40*(1/2), 5-35.

Bloch, I., Hunter, A., Appriou, A., Ayoun, A., Benferhat, S., Besnard, P., Cholvy, L., Cooke, R., Cuppens, F., Dubois, D., Fargier, H., Grabisch, M., Kruse, R., Lang, J., Moral, S., Prade, H., Saffiotti, A., Smets, P., & Sossai, C. (2001). Fusion: General concepts and characteristics, *International Journal of Intelligent Systems, 16*(10), 1107-1134.

Boskovitz, A., Goré, R., & Hegland, M. (2003). A logical formalisation of the Fellegi-Holt method of data cleaning. *International Conference on Intelligent Data Analysis (IDA 2003), Lecture Notes in Computer Science, 2810* (pp. 554-565).

Bouquet, P., Giunchiglia F, van Harmelen F., Serafini, L., & Stuckenschmidt, H. (2003). C-OWL: Contextualizing ontologies. *Proceedings of the 2nd International Semantic Web Conference 2003 (ISWC'03), Lecture Notes in Computer Science , 2870* (pp. 164-179).

Cantwell, J. (1998). Resolving conflicting information. *Journal of Logic, Language and Information, 7*(2), 191-220.

Cholvy, L., & Moral, S. (2001). Merging databases: Problems and examples. *International Journal of Intelligent Systems, Special Issue on Data and Knowledge Fusion, 16*(10).

Celle, A., & Bertossi, L. (1994). Consistent data retrieval. *Information Systems, 19*(4), 33-54.

Dung, P. M. (1995). On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and *n*-person games. *Artificial Intelligence, 77*(2), 321-358.

Elvang-Goransson, M., & Hunter, A. (1995). Argumentative logics: Reasoning from classically inconsistent information. *Data and Knowledge Engineering, 16*(1), 125- 145.

Fikes, R., McGuinness, D. L., & Waldinger, R. (2002, January). A first-order logic semantics for semantic Web markup languages [Knowledge Systems Laboratory Tech. Rep. No. 02-01]. Retrieved September 16, 2004, from http://www.ksl.stanford.edu/KSL_ Abstracts/ KSL-02-01.html

Grant, J., & Subrahmanian, V. S. (2000). Applications of paraconsistency in data and knowledge bases. *Synthese, 125,* 121-132.

Greco S., & Zumpano, E. (2000) Querying inconsistent databases. *Proceedings of the 7th International Conference of Logic for Programming and Automated Reasoning (LPAR 2000), Lecture Notes in Computer Science, 1955* (pp.308-325).

Gutiérrez-Naranjo, M. A., Alonso-Jiménez, J. A., & Borrego- Díaz, J. (2003). A quasimetric for machine learning. In F. J. Garijo, J. C. Riquelme, & M. Toro (Eds.), *Advances in Artificial Intelligence (IBERAMIA 2002), Lecture Notes in Computer Science, 2527* (pp. 193-203).

Huang Z., Harmelen F., & Teije A. (2006). *Reasoning with Inconsistent Ontologies: Framework, Prototype and Experiment. Semantic Web Technologies: Trends and Research in Ontology-based Systems,* (pp. 71-93).

Hunter, A. (1998). Paraconsistent logics. In D. Gabbay & Ph. Smets (Eds.), *Handbook of defeasible reasoning and uncertain information* (pp. 13-44). Dordrecht: Kluwer.

Hunter, A. (2003). Evaluating the significance of inconsistencies. *Proceedings of the International Joint Conference on AI (IJCAI'03)* (pp. 468-473). San Francisco: Morgan Kaufmann.

Kim, W. Y., Choi, B.-J., Hong, E. K., Kim, S.-K., & Lee, D. (2003). A taxonomy of dirty data. *Data Mining Knowledge Discovery, 7*(1), 81-99.

Knight, K. M. (2003). Two information measures for inconsistent sets. *Journal of Logic, Language and Information, 12*(2), 227-248.

Kryszkiewicz, M. (2001). Comparative study of alternativetypes of knowledge reduction in inconsistent systems. *International Journal of Intelligent Systems, 16*(1), 105-120.

Levy, A. (2000). Logic-based techniques in data integration. In J. Minker (Ed.), *Logic-based artificial intelligence* (pp. 575-596). Dordrecht: Kluwer. Link, S. (2003). Consistency enforcement in databases. *Proceedings of the 2nd International Workshop on Semantics in Databases 2001, Lecture Notes in Computer Science, 2582* (pp. 139-159).

Lloyd, F. (1987). *Foundations of logic programming.* Berlin: Springer.

MacGregor, R., & Ko, I.-Y. (2003). Representing contextualized data using semantic Web tools. *Electronic Proceedings of the ISWC'03 Workshop on Practical and Scaleable Semantic Web Systems Services.* Retrieved September 16, 2004, from http://km.aifb.unikarlsruhe.de/ws/psss03/proceedings/macgregor-etal. Pdf

Marquis, P., & Porquet, N. (2003). Resource-bounded paraconsistent inference. *Annals of Mathematics and Artificial Intelligence, 39*(4), 349-384.

Mayol, E., & Teniente, E. (2003). Consistency preserving updates in deductive databases. *Data and Knowledge Engineering, 47*(1), 61-103.

Minker, J. (1999). Logic and databases: A 20 year retrospective. In F. Pirri & H. Levesque (Eds.), *Logical foundations for cognitive agents* (pp. 234-299). Berlin: Springer.

Pan, Z., & Heflin, J. (2003). DLDB: Extending relational databases to support semantic web queries. *Electronic Proceedings of the ISWC'03 Workshop on Practical and Scalable Semantic web Systems Services.* Retrieved September 16, 2004, from http://km.aifb.uni-karlsruhe.de/ws/psss03/proceedings/pan-et-al.pdf

Pradhan, S. (2003). Connecting databases with argumentation. *Web Knowledge Management and Decision Support, 14th International Conference on Applications of Prolog, (INAP 2001), Lecture Notes in Computer Science, 2543* (pp.170-185).

Reiter, R. (1984). A logical reconstruction of relational database theory. In J. L. Mulopoulos & J. W. Schmidt (Eds.), *On conceptual modelling* (pp. 163-189). New York: Springer.

KEY TERMS

Arguments: An argument in a KDB is a pair (B, F), where B is a subset of the KDB such that B entails F. The basic relation among arguments is *rebutting.*

Closed World Assumption: A principle that claims that every atom not entailed by the KDB is assumed to be false. This principle is sound on KDBs with simple syntax, as logic programs.

Contextualizing Logics: Method to formally represent knowledge associated with a particular circumstance on which it has the intended meaning.

Description Logics: Logical formalism to represent structured concepts and the relationships among them. Formally, it is a subset of First Order Logic dealing with *concepts* (monadic predicates) and *roles* (binary predicates) which are useful to relate concepts. KDB in DL are composed of a *Tbox* (the intensional component) and an *Abox* (Box of asserts, the extensional component part).

Logical Inconsistency: A logical theory is inconsistent if there is no logical model for it. In the logic database paradigm, the notion of inconsistency is usually restricted to express the violation of integrity constraints. This restriction makes sense when the data are atomic formulas.

Paraconsistent Logics: Logic systems that limit, for example, the power of classical inference relations to retrieve non trivial information of inconsistent sets of formulas.

Reiter's Formalization of Database Theory: A set of axioms that, when they are added to a relational database, formalizes the reasoning with them. They are the *Unique Names Principle*, the *Domain Closure Axiom*, the *Completion Axioms* and *Equality Axioms*. This formalization permits to identify the answers with logical consequences.

Skolem Noise: A kind of anomalous answers obtained by resolution oriented theorem provers working on non clausal theories. The classical method of skolemization leads to new function symbols with no intended meaning. If these symbols appear in the output, the answer may not be consistently interpreted.

OWL: Ontology Web Language. Language (based on Description Logics) designed to represent ontologies capable of being processed by machines. The World Wide Web Consortium released OWL as recommendation http://www.w3.org/2001/sw/webOnt.

Chapter L
Data Integration: Introducing Semantics

Ismael Navas-Delgado
University of Málaga, Spain

Jose F. Aldana-Montes
University of Málaga, Spain

INTRODUCTION

The growth of the Internet has simplified data access, which has involved an increment in the creation of new data sources. Despite this increment, in most cases, these large data repositories are accessed manually. This problem is aggravated by the heterogeneous nature and extreme volatility of the information on the Web. This heterogeneity includes three types: intentional (differences in the contents), semantic (differences in the interpretation), and schematic (data types, labeling, structures, etc.). Thus, the increase of the available information and the complexity of dealing with this amount of information have involved a considerable amount of research into the subject of heterogeneous data integration. The database community, one of the most important groups dealing with data heterogeneity and dispersion, has provided a wide range of solutions to this problem. However, this issue has also been addressed and solutions have been offered by the information retrieval and knowledge representa-

tion communities, making this area a connection point between the three communities.

The Web offers a huge amount of structured and unstructured information. The representation mechanisms are simple, and there are no rules on how to represent the information, so accessing it is a fundamental problem. Basically, information can be accessed by browsing texts and graphics, and users can follow links or use search engines (based on keyword searches) to reach Web documents. The query response capability and inference mechanisms of the Web are limited in comparison with relational and deductive databases, which allow concise queries and reasoning mechanisms to facilitate new knowledge discovery.

Using ontologies for data integration has some advantages over keyword-based systems: ontologies provide a common and shared vocabulary (concepts) for representing the information included in the document (contents). In addition, ontologies allow us to define relationships between concepts (roles). Thus, we can make use of these concepts and roles to perform more complex queries and retrieve exactly the information in which the user is interested. In this way, it is possible to obtain not only extensional information but also intentional information.

Currently, data integration based on ontologies is a very active area of research, which is referred to by different names—semantic mediation, conceptual mediation, semantic data integration, and so forth—depending on the goal. Consequently, great advances are being made in the context of the Semantic Web, and some important problems such as semantic interoperability are being analyzed. However, there are many other problems to be solved in this area, and it is necessary to study new proposals and find improvements that will cover current and future needs.

BACKGROUND

Traditional approaches for heterogeneous data integration try to resolve semantic and schematic heterogeneity using solutions based on rich data models. These data models tend to represent the relationships between distributed and heterogeneous data sources. Despite the fact that most traditional systems deal with a small number of structured data sources, more recent approaches deal with a larger number of data sources (both structured and unstructured).

Data integration systems are formally defined as a triple $<G,S,M>$, where G is the global (or mediated) schema, S is the heterogeneous set of source schemas, and M is the mapping that maps queries between the source and the global schemas. Both G and S are expressed in languages over alphabets comprised of symbols for each of their respective relations. The mapping M consists of assertions between queries over G and queries over S. When users send queries to the data integration system, they describe those queries over G, and the mapping then asserts connections between the elements in the global schema and the source schemas.

The most important proposal to solve the data integration problem is the wrapper/mediator architecture (Figure 1). In this architecture, a mediator, which is an intermediate virtual database (with a schema G according to a previous definition of data integration system), is established between the data sources (with a set of schemas S) and the application using them. A wrapper is an interface to a data source that translates data into a common data model used by the mediator. The user accesses the data sources through one or several mediator systems that present high-level abstractions (views) of combinations of source data. The user does not know where the data come from but is able to retrieve the data by using a common mediator query language.

Mediator-based integration has query translation as its main task. A mediator in our context

Figure 1. Wrapper/mediator approach: mediator schema, G, is related with data sources schemas, S, by means of mappings, M

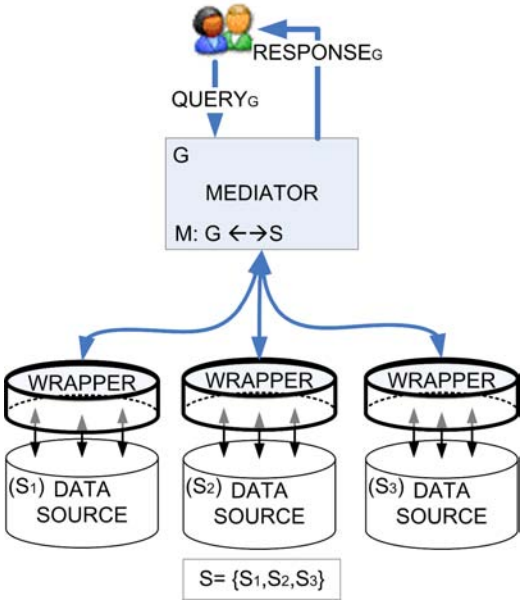
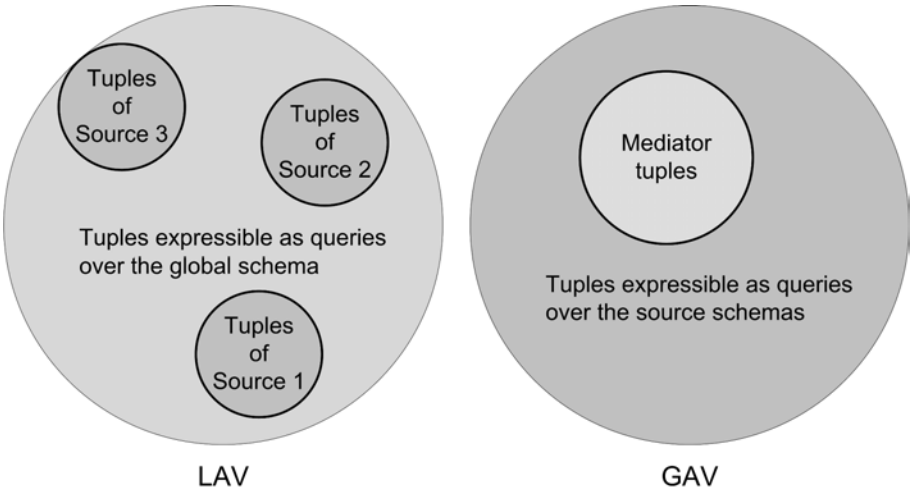


Figure 2. GAV and LAV accessible tuples



is an application that has to solve queries formulated by the user at runtime in terms of either a single or integrated schema. These queries are rewritten in terms of the data source schemas in order to delegate the query resolution to the data sources. This way, expressing the relationships between the integrated schema and the data source schemas is a crucial step in the mediation system development.

The following subsections describe some important problems in data integration systems and the proposed solution. However, due to space limitations, many problems and solutions are not included, such as data source query capabilities, query language, common data model, and so forth.

Approaches

The two main approaches for determining the relationships (mapping) between the integration schema (*G*) and the data source schemas (*S*) are: global-as-view (GAV) and local-as-view (LAV) (Halevy, 2001).

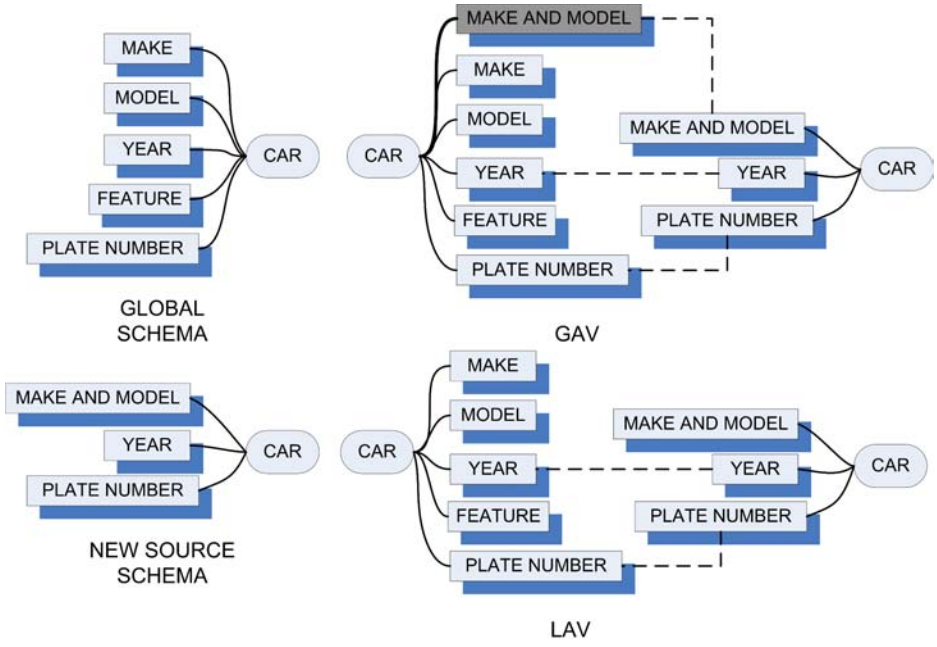
In GAV, each element in the integration schema should be described in terms of a view (a query) over the data sources. In other words, the mapping makes it explicit how to retrieve data from several elements in the integration schema. This approach is effective when the set of data sources is stable (i.e., it remains relatively unchanged). Note, that elements in the integration schema are defined in terms of the data sources, so the addition of a new data source implies the redefinition of some elements in the integration schema. This approach benefits from easier rewriting methods. Most current data integration systems follow the GAV approach. Some important examples are COIN (Madnick & Siegel, 1998), Garlic (Roth, et al., 1996), IBIS (Calì, Calvanese, De Giacomo, Lenzerini, Naggar & Vernacotola, 2003), MOMIS (Beneventano, Bergamaschi, Guerra & Vincini, 2001), Squirrel (Hull & Zhou, 2001), and TSIMMIS (Hammer, Garcia-Molina, Ireland, Papakonstantinou, Ullman & Widom, 1995). These systems usually adopt simple languages for expressing both the global and the source schemas. IBIS (Calì et al., 2003) is the only system we are aware of that takes into account integrity constraints in the global schema. In relation to the mediator schema, which is described in terms of source views, the tuples that are expressible through the mediator schema are a subset of the tuples that can be obtained by the sources (see Figure 3).

In LAV, each element in the data source schemas should be described in terms of the integration schema. This kind of approach is effective when the integration schema is stable and well established in the domain/application. In this case, the extension of the system is easy because it only implies adding the description of the new data source in terms of the integration schema. This approach implies a more difficult query reformulation and evaluation, which contrasts with the benefits of greater scalability. AMOS (Risch, 2004) is the most representative system that applies the LAV approach. In this case, the source schemas are related with the mediator schema, so the tuples that could be expressed with the mediator schema are a superset of the tuples that can be obtained with the data sources (see Figure 3).

The addition of new sources to the mediation system implies different global schema evolution in GAV and LAV approaches. Thus, Figure 3 shows an example of how the global schema and the mappings change when a new source is added. The GAV approach implies modifying the global schema, and the LAV approach could imply that part of the source schema is not taken into account because it is not possible to find a mapping.

Despite the opposing nature of the two approaches, some proposals have tried to combine them. Thus, GLAV (Fagin, Kolaitis, Miller & Popa, 2003; Friedman, Levy & Millstein, 1999) allows GAV and LAV definitions, and the system is in charge of translating from LAV to GAV.

Figure 3. GAV and LAV examples on new source addition



However, the limited success of this approach is because it involves the advantages as well as the drawbacks of both approaches. Another approach, called Both as View (BAV) (McBrien & Poulovassilis, 2003), consists of a specification of the transformation of the local schema into the given global schema in such a way that each one can be seen as defined in terms of the other.

Query rewriting is a crucial problem in data integration because it should determine how the user query is decomposed into queries being solved in data sources. Thus, this task depends on the approach chosen by the mediator developer.

Query Optimization

Query processing and optimization in mediator access to distributed data sources (Web data resources) can present some common problems in traditional database integration systems. However, data distribution introduces several additional problems. Query optimization should analyze the cost of all possible query plans generated at the query rewriting stage and subsequently take this cost into account in order to decide which query plan is best for solving the user query. Unpredictable behavior of data transmissions in the wide area networks and strong autonomy of the remote database systems makes the estimation of query processing time hard and imprecise. It makes a query processing plan optimal on one occasion yet ineffective on the other.

Besides, cost-based optimizations are difficult to develop because data sources do not usually publish statistics. Thus, the mediator should store statistics about the data sources previously used in order to solve a particular user query. However, this may not be enough because querying Web data sources is not as stable as querying a database, and they may have connection overload, be unavailable, and so forth. Thus, some approaches have included different parameters within the cost

of a query. Those parameters relate to the quality of source and data information. The argument is that such quality parameters are as important or even more important in the context of the Web than classical parameters such as response time.

The external factors affecting the performance of query processing promote the reactive query processing techniques. Reactive processing of a query starts from a hypothetically optimal plan, and whenever further processing is impossible due to the network problems or unavailability of data from an external database site, the processing is continued according to a modified plan.

Early works on reactive query processing techniques are based on partitioning (Kabra & DeWitt, 1998) and dynamic modification of query processing plans (Getta, 2000). Partitioning decomposes a query execution plan into subplans at a point when the further computations are no longer possible due to a lack of data. A dynamic modification technique finds a plan equivalent to the original one and such that it is possible to continue integration of the available datasets. Similar techniques include query scrambling (Urhan, Franklin & Amsaleg, 1998) and dynamic scheduling of operators (Urhan & Franklin, 2001). Other works include the new versions of join operation customized to query processing for distributed databases (pipelined join operator Xjoin) (Urhan & Franklin, 2000), ripple join (Haas & Hellerstein, 1999), double pipelined join (Ives, Florescu, Friedman, Levy & Weld, 1999), and hash-merge join (Mokbel, Lu & Aref, 2004).

An adaptive data partitioning (Ives, Halevy & Weld, 2004) technique processes different partitions of the same argument using different data integration plans. The papers on adaptive data partitioning and optimizations of data stream processing (Getta & Vossough, 2004) were the first attempts to use the associativity of join operation to integrate separate partitions of the same arguments with the different integration plans. An adaptive query processing of data integration plans proposed in Getta (2005) also concentrates on the sets of elementary operations for data integration and proposes an algorithm that finds the best integration plan for the outcomes of the most recent transmissions.

Many of the recently developed techniques for data stream processing can be reused for data integration. It is so because the implementations of both groups of systems require algorithms that efficiently recompute a query each time one of the data containers processed by the query is changed.

Due to the difficulty of calculating the cost of a query plan with distributed data sources, some alternative approaches have been proposed. One approach for optimizing query resolution in mediators proposes the use of semantic query optimization (i.e., using available knowledge about the data sources and/or knowledge stored about the previously executed queries). For some systems, on the other hand, the main goal in terms of optimization is to reduce the number of sources to be accessed, as this would reduce the cost of the query evaluation.

Mediator-Based Data Integration Systems

Table 1 presents a list of the most important mediator-based data integration systems. These mediators are classified in terms of the parameters described in the previous section.

CURRENT TRENDS

New developments in the area of data integration have been directed toward the addition of new mechanisms that allow the resource semantics to be made explicit (Noy, 2004; Woehrer et al., 2005). In this way, proposals have changed from integration schemas based on relational databases to ontologies being used as integration schemas. The proposals have included the use of standards

Table 1. Traditional mediator-based data integration systems

System	Approach	Source Capabilities	Query Optimization	Resource Location
AMOS (Active Mediators Object System) II (Risch, 2004)	LAV	Conventional databases, text files, data exchange files, WWW pages, and programs	Database optimization techniques	Internet
Disco (Tomasic, Kapitskaia, Naacke, Bonnet, Raschid & Amouroux, 1997)	GAV	Web-accessible data (use of wrappers)	Cost-based optimization	
Garlic (Roth, 1996)	GAV	WWW pages	Database optimization techniques	Remote Databases
Strudel (Fernandez, Florescu, Kang Levy & Suciu, 1997)	GAV	Databases	---	Local Web site and external resources
Momis (Beneventano et al., 2001)	GAV	Structured and unstructured information sources	Residual filtering and full disjunction operation	Distributed data sources
AutoMed (Boyd, Kittivoravitkul, Lazanitis, McBrien & Rizopoulos, 2004)	BAV	Structured and Semistructured Data	---	Distributed data sources
Clio (Popa, Velegrakis, Miller, Hernandez & Fagin, 2002)	GLAV	Structured and Semistructured Data	---	Distributed data sources

such as XML (W3C, 2004) in order to make the interoperability between applications possible. The data-sharing applications play an important role in emerging domains such as e-commerce, bioinformatics, and ubiquitous computing. Thus, some recent developments should dramatically increase the need for and the deployment of applications that require semantic integration.

Mediators are usually developed as monolithic systems that envelop data source location and semantics. Furthermore, their architecture based on wrappers involves a high degree of coupling among the mediator components. This coupling does not allow services to be shared with other organizations or the dynamic integration of new data sources. Therefore, wrappers must be redesigned and added manually for each mediation system. That is, Semantic mediation systems are developed as black boxes that receive a query and return a result. In this way, normally a full solution is built from scratch. This methodology has some disadvantages, such as the difficulty of reusing previous work (developed in other programming languages where the metadata are stored in different ways), aisle development avoiding shared development among several distributed development teams (because of the monolithic view of the systems), and so forth. Some proposals have tried to solve these problems by dividing the mediator components into distributed services that can interoperate.

The growth of the Semantic Web (Berners-Lee, Hendler & Lassila, 2001) will further fuel data-sharing applications and underscore the key role that semantic integration plays in their deployment. Recently, research has been devoted to the problem of semantic integration (Noy, 2004; Woehrer et al., 2005). Semantic integration adds a few additional considerations to the logical information integration problems. In this scenario, sources export not only their logical schema but also their conceptual model to the mediator, thus exposing their concepts, roles, classification hierarchies, and other high-level semantic constructs to the mediator. Semantic integration allows information sources to export their schema at an appropriate level of abstraction to the mediator.

Mediators are applications that offer transparent access to the data in distributed resources, these being considered by the users as a single database. In this way, a semantic integration system is one that translates instances from distributed resources to instances of the mediator ontology. The main goal of using ontologies is not to have the problem of heterogeneity. However, it is not realistic to think that will be an agreement on a small set of ontologies. We will need to map between ontologies. Thus, we can identify three dimensions of semantic integration:

- **Mapping discovery.** Given two ontologies, determine which concepts and properties represent similar notions.
- **Declarative formal representation of mappings.** Determine how we represent the mappings between two ontologies to enable reasoning with mappings.
- **Reasoning with mappings.** Determine the types of reasoning involved in using the mappings.

Some proposals for using ontologies in mediators are Carnot (Woelk, 1993), SIMS (Yigal, 1997), OntoMerge (Dou, 2003), OIS Framework (Calvanese, 2001), and Observer (Mena, 1996).

CONCLUSION

We have presented the basic characteristics of data integration systems, describing them in a simple way in order to enable an in-depth study of this topic. We have reviewed the most important systems in this area and have divided them into two groups: traditional and ontology-based systems. We have presented a comparative table in order to clarify the differences between the systems presented. Finally, we have introduced some interesting issues being studied in this area and that represent future trends in it.

REFERENCES

Beneventano, D., Bergamaschi, S., Guerra, F., & Vincini, M. (2001). The MOMIS approach to information integration. Proceedings of the 3rd International Conference on Enterprise Information Systems, Setubal, Portugal, 91.

Berners-Lee, T., Hendler, J., & Lassila, O. (2001). The semantic Web. *Scientific American*.

Boyd, M., Kittivoravitkul, S., Lazanitis, C., McBrien, P.J., & Rizopoulos, N. (2004). *AutoMed: A BAV data integration system for heterogeneous data sources*. Proceedings of the CAiSE04, 3084, 82–97.

Calì, A., Calvanese, D., De Giacomo, G., Lenzerini, M., Naggar, P., & Vernacotola, F. (2003). *IBIS: Semantic data integration at work*. Proceedings of the CAiSE 2003, 79–94.

Fagin, R., Kolaitis, P., Miller, R.J., & Popa, L. (2003). *Data exchange: Semantics and query answering*. Proceedings of the ICDT 2003, 207–224.

Fernandez, M., Florescu, D., Kang, J., Levy, A., & Suciu, D. (1997). *STRUDEL: A Web site management system*. Proceedings of the ACM SIGMOD International Conference on Management of Data, 26, 549–552.

Friedman, M., Levy, A., & Millstein, T. (1999). *Navigational plans for data integration*. Proceedings of the AAAI'99, 67–73.

Getta, J.R. (2000). *Query scrambling in distributed multidatabase systems.* Proceedings of the 11th International Workshop on Database and Expert Systems Applications, DEXA' 2000, 647–652.

Getta, J.R. (2005). *On adaptive and online data integration*. Proceedings of the 21st International Conference on Data Engineering, International Workshop on Self-Managing Database Systems, 1212–1220.

Getta, J.R., & Vossough, E. (2004). Optimization of data stream processing. *SIGMOD Record, 33*, 34–39.

Haas, P.J., & Hellerstein, J.M. (1999). *Ripple joins for online aggregation*. Proceedings of the ACM SIGMOD International Conference on Management of Data, 287–298.

Halevy, A.Y. (2001). Answering queries using views: A survey. *VLDB J, 10*(4), 270–294.

Hammer, J., Garcia-Molina, H., Ireland, K., Papakonstantinou, Y., Ullman, J., & Widom, J. (1995). *Information translation, mediation, and mosaic-based browsing in the TSIMMIS system.* Proceedings of the ACM SIGMOD International Conference on Management of Data, San Jose, California, 483.

Hull, R., & Zhou, G. (1996). *A framework for supporting data integration using the materialized and virtual approaches.* Proceedings of the ACM SIGMOD Conference on Management of Data, Montreal, Canada, 481–492.

Ives, Z.G., Florescu, D., Friedman, M., Levy, A.Y., & Weld, D.S. (1999). *An adaptive query execution system for data integration.* Proceedings of the ACM SIGMOD International Conference on Management of Data, 299–310.

Ives, Z.G., Halevy, A.Y., & Weld, D.S. (2004). *Adapting to source properties in processing data integration queries.* Proceedings of the ACM SIGMOD International Conference on Management of Data, 395–406.

Kabra, N., & DeWitt, D.J. (1998). *Efficient mid-query re-optimization of sub-optimal query execution plans.* Proceedings of the ACM SIGMOD International Conference on Management of Data, 106–117.

Madnick, S.E., & Siegel, M. (1998). The COntext INterchange (COIN) project: Data extraction and interpretation from semi-structured Web sources.

McBrien, P.J., & Poulovassilis, A. (2003). *Data integration by bi-directional schema transformation rules.* Proceedings of the ICDE'03.

Mokbel, M.F., Lu, M., & Aref, W.G. (2004). *Hash-merge join: A nonblocking join algorithm for producing fast and early join results.* Proceedings of the 20th International Conference on Data Engineering ICDE2004, 251–263.

Noy, N.F. (2004). Semantic integration: A survey of ontology-based approaches. *SIGMOD Rec., 33*(4), 65–70.

Popa, L., Velegrakis, Y., Miller, R.J., Hernandez, M.A., & Fagin, R. (2002). *Translating Web data.* Proceedings of the International Conference on Very Large Data Bases (VLDB), 598–609.

Risch, T. (2004). Mediators for querying heterogeneous data. In M.P. Singh (Ed.), *The practical handbook of Internet computing.* Chapman & Hall/CRC.

Roth, M.T., et al. (1996). *The garlic project.* Proceedings of the SIGMOD Conference, 557

Tomasic, A., Kapitskaia, O., Naacke, H., Bonnet, P., Raschid, L., & Amouroux, R. (1997). *The distributed information search component (disco) and the World Wide Web.* Proceedings of the ACM SIGMOD Conference on Management of Data.

Urhan, T., & Franklin, M.J. (2000). Xjoin: A reactively-scheduled pipelined join operator. *IEEE Data Engineering Bulletin, 23,* 27–33.

Urhan, T., & Franklin, M.J. (2001). *Dynamic pipeline scheduling for improving interactive performance of online queries.* Proceedings of the International Conference on Very Large Databases VLDB'01.

Urhan, T., Franklin, M.J., & Amsaleg, L. (1998). *Cost based query scrambling for initial delays.* Proceedings of the 1998 ACM SIGMOD International Conference on Management of Data, 130–141.

W3C. (2004). Extensible markup language (XML). http://www.w3.org/XML/.

Woehrer, A., et al. (2005). Towards semantic data integration for advanced data analysis of grid data repositories. Proceedings of the First Austrian Grid Symposium, Schloss Hagenberg, Austria.

KEY TERMS

Data Integration: The problem of combining data from multiple heterogeneous data sources and providing a unified view of these sources to the user. Such unified view is structured according to a global schema. Issues addressed by a data integration system include specifying the mapping between the global schema and the sources and processing queries expressed on the global schema.

Mediator: Systems that filter information from one or more data sources that are usually accessed using wrappers. The main goal of these systems is to allow users to make complex queries over heterogeneous sources as if it were a single one, using an integration schema. Mediators offer user interfaces for querying the system based on the integration schema. They transform user queries into a set of subqueries that other software components (the wrappers), which encapsulate data sources' capabilities, will solve.

Ontology: A logical theory accounting for the intended meaning of a formal vocabulary (i.e., its ontological commitment to a particular conceptualization of the world). The intended models of a logical language using such a vocabulary are constrained by its ontological commitment. An ontology indirectly reflects this commitment (and the underlying conceptualization) by approximating these intended models.

Ontology Mapping: Given two ontologies, A and B, mapping one ontology with another means that for each concept (node) in ontology A, we try to find a corresponding concept (node) that has the same or similar semantics in ontology B, and vice versa.

Semantic Integration: In semantic integration, sources export not only their logical schema but also their conceptual model to the mediator, thus exposing their concepts, roles, classification hierarchies, and other high-level semantic constructs to the mediator. Semantic integration allows information sources to export their schema at an appropriate level of abstraction to the mediator.

Semantic Web: An extension of the current Web in which information is given well-defined meaning, better enabling computers and people to work in cooperation. Berners-Lee, et al. (2001) said that in the context of the Semantic Web, the word *semantic* meant "machine-processable." They explicitly ruled out the sense of natural lan-

guage semantics. For data, the semantics convey what a machine can do with those data.

Wrapper: An interface to a data source that translates data into a common data model used by the mediator. The user accesses the data sources through one or several mediator systems that present high-level abstractions (views) of combinations of source data. The user does not know where the data come from but is able to retrieve the data by using a common mediator query language.

Chapter LI
An Overview of Ontology-Driven Data Integration[1]

Agustina Buccella
Universidad Nacional del Comahue, Argentina

Alejandra Cechich
Universidad Nacional del Comahue, Argentina

INTRODUCTION

New software requirements have emerged because of innovation in technology, specially involving network aspects. The possibility enterprises, institutions and even common users can improve their connectivity allowing them to work as they are at the same time, generates an explosion in this area. Besides, nowadays it is very common to hear that large enterprises fuse with others. Therefore, requirements as interoperability and integrability are part of any type of organization around the world. In general, large modern enterprises use different database management systems to store and search their critical data. All of these databases are very important for an enterprise but the different interfaces they possibly have make difficult their administration. Therefore, recovering information through a common interface becomes crucial in order to realize, for instance, the full value of data contained in the databases (Hass & Lin, 2002).

Thus, in the '90s the term *Federated Database* emerged to characterize techniques for proving an integrating data access, resulting in a set of distributed, heterogeneous and autonomous databases (Busse, Kutsche, Leser & Weber, 1999; Litwin, Mark & Roussoupoulos, 1990; Sheth & Larson, 1990). Here is where the concept of *Data Integration* appears. This concept refers to the process of unifying data sharing some common semantics but originated from unrelated sources. Several aspects must be taken into account when working with Federated Systems because the main characteristics of these systems make more difficult the integration tasks. For example,

the *autonomy* of the information sources, their *geographical distribution* and the *heterogeneity* among them, are some of the main problems we must face to perform the integration. *Autonomy* means that users and applications can access data through a federated system or by their own local system. *Distribution* (Ozsu & Valduriez, 1999) refers to data (or computers) spread among multiple sources and stored in a single computer system or in multiple computer systems. These computer systems may be geographically distributed but interconnected by a communication network. Finally, *heterogeneity* relates to different meanings that may be inferred from data stored in databases. In (Cui & O'Brien, 2000), heterogeneity is classified into four categories: *structural*, *syntactical*, *system*, and *semantic*. Structural heterogeneity deals with inconsistencies produced by different data models whereas syntactical heterogeneity deals with consequences of using different languages and data representations. On the other hand, system heterogeneity deals with having different supporting hardware and operating systems. Finally, semantic heterogeneity (Cui & O'Brien, 2000) is one of the most complex problems faced by data integration tasks. Each information source included in the integration has its own interpretation and assumptions about the concepts involved in the domain. Therefore, it is very difficult to determine when two concepts belonging to different sources are related. Some relations among concepts that semantic heterogeneity involves are: synonymous, when the sources use different terms to refer to the same concept; homonymous, when the sources use the same term to denote completely different concepts; hyponym, when one source contains a term less general than another in another source; and hypernym, when one source contains a term more general than another in another source; etc.

In this paper we will focus on the use of ontologies because of their advantages when using for data integration. For example, an ontology may provide a rich, predefined vocabulary that serves as a stable conceptual interface to the databases and is independent of the database schemas; knowledge represented by the ontology may be sufficiently comprehensive to support translation of all relevant information sources; an ontology may support consistency management and recognition of inconsistent data; etc. Then, the next section will analyze several systems using ontologies as a tool to solve data integration problems.

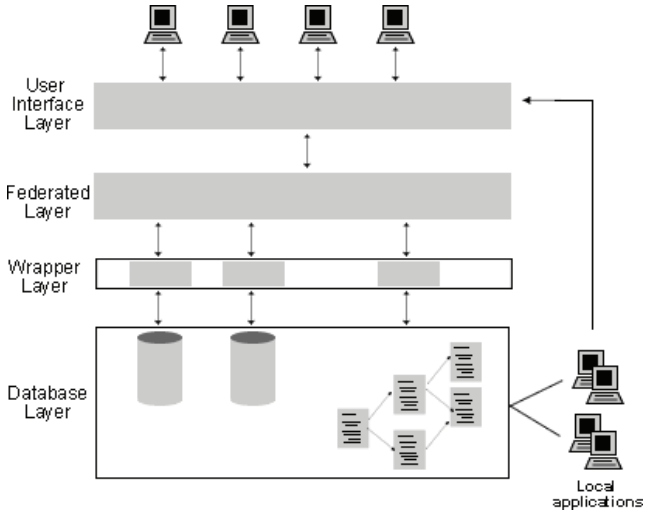
BACKGROUND

Recently, the term *Federated Databases* has evolved to *Federated Information Systems* because of the diversity of new information sources involved in the federation, such as HTML pages, databases, filing, etc., either static or dynamic. A useful classification of information systems based on the dimensions of distribution and heterogeneity can be found in (Busse et al., 1999). Besides, this work defines the classical architecture of federated systems (based on Sheth & Larson (1990)) which is widely referred by many researches. Figure 1 shows this architecture.

In the figure, the *wrapper layer* involves a number of modules belonging to a specific data organization. These modules know how to retrieve data from the underlying sources hiding their data organizations. As the federated system is autonomous, local users may access local databases through their local applications independently from users of other systems. Otherwise, to access the federated system, they need to use the *user interface layer*.

The *federated layer* is one of the main components currently under analysis and study. Its importance comes from its responsibility to solve the problems related to semantic heterogeneity, as we previously introduced. So far, different approaches have been used to model this layer. In general they use ontologies as tools to solve these semantic problems among different sources

Figure 1. Architecture of federated systems



(Stephens, Gangam & Huhns, (2004); Le, Dieng-Kuntz & Gandon, (2004); Giunchiglia, Yatskevich & Giunchiglia, (2005), Buccella, Cechich & Brisaboa, 2003; Buccella, Cechich & Brisaboa, 2004).

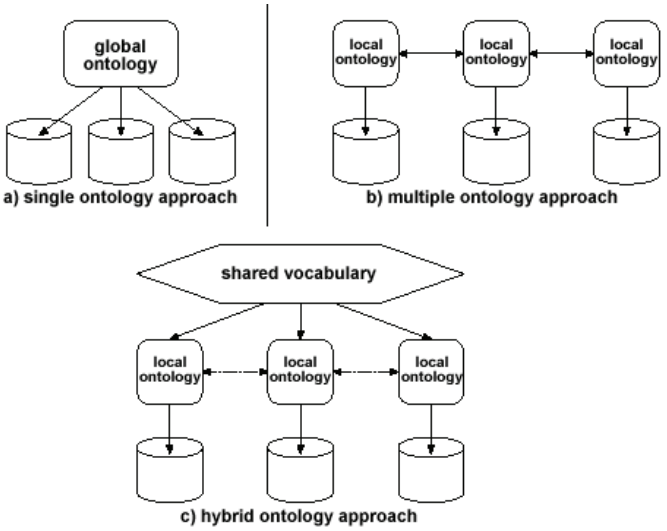
ONTOLOGY-BASED DATA INTEGRATION

The term ontology was introduced by Gruber (1993) as an *"explicit specification of a conceptualization"*. A *conceptualization*, in this definition, refers to an abstract model of how people commonly think about a real thing in the world; and *explicit specification* means that concepts and relationships of the abstract model receive explicit names and definitions.

An ontology gives the name and description of the domain specific entities by using predicates that represent relationships between these entities. The ontology provides a vocabulary to represent and communicate domain knowledge along with a set of relationships containing the vocabulary's terms at a conceptual level. Therefore, because of its potential to describe the semantics of information sources and to solve the heterogeneity problems, the ontologies are being used for data integration tasks.

Recent surveys have emerged describing and analyzing proposals to ontology matching (Euzenat & Shvaiko, 2007; Shvaiko & Euzenat, 2005; Rahm & Bernstein, 2001). But some of these surveys only analyze methodologies and tools to build an integrated system, without analyzing how the whole system works. On the other hand, surveys based on ontology-based systems for data integration can be found in the literature (Euzenat & Shvaiko, 2007; Kalfoglou & Schorlemmer, 2003; Wache et al., 2001). For example, the work presented by Euzenat & Shvaiko (2007) describe and analyze a widely set of ontology matching proposals. Although it is focused on methodologies to improve matchings, new approaches of complete integrated systems are also taked into account. Besides, more information with links to several approaches can be found at OntologyMatching.org[2]. Another example is (Wache et al., 2001) in which authors focus on some aspects of the use of ontologies, the language representation, map-

Figure 2. Classification of the use of ontologies



pings and tools. This work also classifies the use of ontologies into three approaches: *single ontology approach*, *multiple ontology approach* and *hybrid ontology approach*, as Figure 2 shows.

We use this classification to categorize the use of ontologies by each proposal. At same time, with respect to the integration process and based on the approaches aforementioned, two new approaches (Calvanese & Giacomo, 2005; Calvanese, Giacomo & Lenzerini, 2001) have emerged for specifying mappings in an integrated system. One, of them, the global-as-view (GAV) approach, is based on the definition of global concepts as views over the sources, that is, each concept of the global view is mapped to a query over the sources. The other approach, local-as-view (LAV), is based on the definition of the sources as views over the global view. Thus, the information the sources contain is described in terms of this global view.

In this section we will focus on how ontology-based systems address the semantic heterogeneity problems. We have investigated many systems, which follow some of the approaches of Figure 2, considering relevant aspects respect to the *use of ontologies*. They are, *reusability, changeability* and *scalability*. These aspects allow us analyze whether quality properties are improved due a particular use of ontologies in a given system.

The *use of ontologies* refers to how ontologies help solve data integration problems. Commonly, the systems show how ontologies are used to solve the integration problems and how ontologies interact with other architectural components. In this aspect the systems describe its ontological components and the ways to solve the different semantic heterogeneity problems.

Then, we characterize this feature by means of three quality characteristics: *reusability, changeability* and *scalability*. *Reusability* refers to the ability of reusing the ontologies, that is, ontologies defined to solve other problems can be used in a system because of either the system supports different ontological languages (and) or define local ontologies. *Changeability* refers to the ability of changing some structures within an information source, without producing substantial changes in the system components. Finally, *scalability* refers

to the possibility of easily adding new information sources to the integrated system.

In order to give a wide view of ontology-based systems, we divide the approaches into two branches. On one branch, we briefly analyze systems proposed during the 90's decade and no longer under research. Table 1 shows our classification taking into account some of these systems. The columns of the table contain the three relevant aspects within the use of ontologies aforementioned. A more extensive analysis of these systems can be found in (Buccella, Cechich & Brisaboa, 2005a; Buccella, Cechich & Brisaboa, 2005b).

As we can see, SIMS (Ambite et al., 1997) and Carnot (Woelk, Cannata, Huhns, Shen & Tomlinson, 1993) are scored as "Not supported" in the three columns. In the first column this value means that both ontologies are not reusable because both define a global ontology or a single ontology approach in order to integrate data. The same happens with the second column, changeability, because no support is given by these systems to bear changes in the information sources. When one source changes, the global ontology must be rebuilt.

Finally, scalability, is not supported because again by adding a new information source, the rebuilding of the global ontology is generated. The problem of dealing with the global ontology approach is that we must manage a global integrated ontology, which involves administration, maintenance, consistency and efficiency problems that are very hard to solve.

On the other hand, OBSERVER system (Mena, Kashyap, Sheth & Illarramendi, 2000) uses the multiple ontologies approach to alleviate the problems presented by the single approach. As each ontology can be created independently, the reusability is fully supported. However, changeability and scalability are semi-supported because changes in local ontologies also generate changes in the mappings.

Finally, the three last systems following the hybrid ontology approach are scored better. Although they propose different and novel ways to improve these characteristics, all these mechanisms conclude in good results. For example, in InfoSleuth (Bayardo et al., 1997), to add a resource to the system it only needs to have an interface to advertise its services and let other agents make use of them immediately.

On the other branch, we analyze more recent systems widely referred by researchers, although some of them are still under development. We use the same characteristics in order to analyze the evolution of the systems. For example, we notice that most of the new systems do not follow the

Table 1. Ontology-based systems during 90's decade

Approach	*Ontology-based Systems*	*Reusability*	*Changeability*	*Scalability*
Single Ontology Approach	SIMS (Ambite et al., 1997)	Not supported	Not supported	Not supported
	Carnot (Woelk, Cannata, Huhns, Shen & Tomlinson, 1993)			
Multiple Ontology Approach	OBSERVER (Mena, Kashyap, Sheth & Illarramendi, 2000)	Fully-supported	Supported	Supported
Hybrid Ontology Approach	KRAFT (Preece, Hui, Gray, Jones & Cui, 1999)	Supported	Semi-supported	Semi-supported
	COIN (Firat, Madnick & Grosof, 2002; Goh et al., 1999)	Supported	Supported	Supported
	InfoSleuth (Bayardo et al., 1997)	Supported	Supported	Supported

single ontology approach. The inherent disadvantages of this type of systems make difficult to deal with them. However, some systems such as MediWeb (Arruda, Lima & Baptista, 2002) and DIA (Medcraft, Schiel & Baptista, 2003) apply this approach but without better results. Only in the case of DIA the changeability and scalability can be improved because the global ontology does not need to be modified when a source is added; only an ontology-schema matching table is necessary to add a new database to the integrated system.

As a new proposal within the multiple ontology approach we can cite OntoGrate (Dou & LePendu, 2006). Here, first order logic inference is used to find mappings among underlying ontologies. By a set of rules and user assistance, databases (that are the sources of the system) are translated to source ontologies. Then, these ontologies are merged by applying bridging axioms and a reasoner. In this system, reusability is not supported because the ontologies are created based on the underlying databases by using a specific language and particularities. For changeability or scalability, the reasoning process must be re-executed over the information changed or added. The speed of this process will depend on how large the ontologies are.

Recently, new approaches have emerged by applying the advantages the hybrid ontology approach provides. Some novel approaches are MOBS (McCann, Doan, Varadarajan & Kramnik, 2003), AutoMed (Zamboulis & Poulovassilis, 2006) and Alasoud et.al (Alasoud, Haarslev & Shiri, 2005). In MOBS (Mass Collaboration to Build Systems) something different is proposed. Instead of creating a global ontology based on underlying domain ontologies, all of them are created at the same time. That is, in this first step the global ontology contains general concepts users want to recover instead of information extracted from underlying sources. Then, initial mappings are defined by using a random assignment or a schema matching tool. Finally, the mass collaboration of this approach is in the user feedback needed to readjust the system and to define the correct mappings. In this way, the ontologies are reusable because they can be used by this system or by another one. But changeability and scalability can be tedious because the user collaboration process must be re-run each time something is modified or added. In AutoMed and Alasoud et.al approaches materialized views can be defined. In the case of Alasoud et.al an integrated view (as a global ontology) is partially materialized to improve query answering time, and to take advantages of fully materialized and fully virtual approaches. In this way, the answers to queries are retrieved from materialized as well as virtual views. Besides, the mappings between concepts in the source ontologies are described in terms of the integrated view. Thus, adding a new ontology to this system does not require changes in the global view.

Considering the approach to specify mappings, the GAV approach is generally chosen by the proposals. Only Alasoud et.al and OntoGrade systems define something similar to the LAV approach.

Advantages and disadvantages of each approach have to be considered when an integration process is initiated (Cali et.al, 2002). For example, in the LAV approach restrictions over the sources can be defined easily. On the contrary, in the GAV approach it is easier to define restrictions over the global view. With respect to scalability, following the LAV approach, adding a new source to the integrated system does not require changes in the global view. On the other hand, in the GAV approach when a new source is added, changes over the global view are required. However, scalability over the global view is easier in this last approach. Finally, considering query processing, LAV requires reasoning mechanisms in order to answer them. Contrarily, in GAV conventional mechanisms can be implemented.

In order to take advantage of the benefits of both approaches, a new approach, called global-

local-as-view (GLAV), has been proposed by (Friedman, Levy & Millstein, 1999). Although the query process is out of the scope of this work, we consider that a study of query capabilities of each proposal shoud be performed. For example, in (Calvanese, Giacomo & Lenzerini, 2001) two case studies are presented in order to show the need of suitable techniques for query answering.

FUTURE TRENDS

Several systems compared here are still in a development stage and, as we have explained, some problems should be solved in order to reach a good integration. Recent proposals, which are based on the old ones, try to improve several aspects in order to propose an integral solution. For example, new systems do not apply a global ontology approach such as SIMS did, due to problems as scalability and changeability.

Other proposals applying the others approaches (multiple and hybrid), intent to solve some characteristics of the global approach, but additional efforts have to be done to reach them. For example, reusability deserves more attention because several ontologies created for the semantic web might be part (some day) of an integrated system. In this way, thinking of particular ontology with specific requirements, as in OntoGrate, is not such a good idea.

There are several proposals (Euzenat & Shvaiko, 2007), which were not analyzed here, only implementing methods for semantic matching. These proposals are complementary because they propose mechanisms to create the global view or the mappings in an efficient way. Euzenat & Shvaiko (2007) presents a survey that is mainly focused on ontology matching proposals. The task of ontology matching involves the process of finding relationships or correspondences between entities of different ontologies. In this paper these proposals are not considered because we do not focus on this type of problems but on the functionality of the whole integrated system. Nevertheless, semantic matching is a crucial area that must be analyzed to assess the processes the system involves.

Currently, the task of building an integrated system is not easy. By trying to improve some essential aspects, other equally important ones can be forgotten. For example, in order to improve methods for finding mappings, more expressive ontologies are created. But aspects as understandability and complexity are not taken into account. Furthermore, new systems must consider all these aspects in order to give an integral solution.

CONCLUSION

Today, semantic heterogeneity implies many complex problems that are addressed by using different approaches, including ontologies, which give a higher degree of semantics to the treatment of data involved in the integration.

We have analyzed several systems accordingly to the use of the ontologies and we have evaluated three important aspects also related with them – reusability, changeability and scalability. Each system has implemented its own solution with advantages and disadvantages, but some elements in common and some original aspects can be found. There are other important aspects we could have considered for characterizing current ontology-based systems. Among them, we should mention the use of automated tools for supporting the manipulation of ontologies.

Of course, further characterization is needed to completely understand the use of ontologies for data integration. Hope our work will motivate the reader to deeper immerse in this interesting world.

REFERENCES

Alasoud, A., Haarslev, V., & Shiri, N. (2005). A hybrid approach for ontology integration. *Proceedings of the 31st VLDB Conference*. Trondheim, Norway.

Ambite, J. L., Arens, Y., Ashish, N., Knoblock, C. A., & collaborators (1997). The SIMS manual 2.0. Technical Report, *University of Southern, California*. December 22. Retrieved August 01, 2007, from http://www.isi.edu/sims/papers/sims-manual.ps.

Arruda, L., Baptista, C., & Lima, C. (2002). MEDIWEB: A mediator-based environment for data integration on the Web. *Databases and Information Systems Integration*. ICEIS, 34-41.

Baru, C. (1998). Features and requirements for an XML view definition language: Lessons from XML information mediation. In *W3CWorkshop on Query Language (QL´98)*. Boston.

Bayardo Jr., R. J., Bohrer, W., Brice, R., Cichocki, A., Fowler, J., Helal, A., Kashyap, V., Ksiezyk, T., Martin, G., Nodine, M., Rashid, M., Rusinkiewicz, M., Shea, R., Unnikrishnan, C., Unruh, A., & Woelk, D. (1997). InfoSleuth: Agent-based semantic integration of information in open and dynamic environments. *Proceedings ACM SIGMOD International Conference on Management of Data*, 195-206.

Buccella, A., Cechich, A., & Brisaboa, N. R. (2003). An ontology approach to data integration. *Journal of Computer Science and Technology, 3*(2), 62-68.

Buccella, A., Cechich, A., & Brisaboa, N. R. (2004). A federated layer to integrate heterogeneous knowledge. *In VODCA'04 First International Workshop on Views on Designing Complex Architectures*. Bertinoro, Italy. Electronic Notes in Theoretical Computer Science, Elsevier Science B.V, 101-118.

Buccella, A., Cechich, A., & Brisaboa, N. R. (2005a). Ontology-based data integration: Different approaches and common features. *Encyclopedia of Database Technologies and Applications*. Rivero L., Doorn J., and Ferraggine V. Editors. Idea Group.

Buccella, A., Cechich, A., & Brisaboa, N. R. (2005b). Ontology-based data integration methods: A framework for comparison. *Revista Colombiana de Computación 6*(1).

Busse, S., Kutsche, R. D., Leser, U., & Weber, H. (1999). Federated information systems: Concepts, terminology and architectures. *Technical Report. Nr. 99-9*, TU Berlin.

Calí, A., Calvanese, D., Giacomo, D., & Lenzerini, M. (2002). On the expressive power of data integration systems. *In Proceedings of the 21st Int. Conf. on Conceptual Modeling, ER, 2503 of Lecture Notes in Computer Science*. Springer.

Calvanese, D., & Giacomo, G. D. (2005). Data integration: A logic-based perspective. *AI Magazine, 26*(1), 59-70.

Calvanese, D., Giacomo, G. D., & Lenzerini, M. (2001). A framework for ontology integration. *In SWWS*, 303-316.

Cui, Z., & O'Brien, P. (2000). Domain ontology management environment. In *Proceedings of the 33rd Hawaii International Conference on System Sciences*.

Dou, D., & LePendu, P. (2006). Ontology-based Integration for relational databases. *SAC'06*, 461-466.

Euzenat, J., & Shvaiko, P. (2007). *Ontology matching*. Heidelberg: Spinger-Verlag, (DE), isbn 3-540-49611-4. (pp. 341).

Firat, A., Madnick, S., & Grosof, B. (2002). Knowledge integration to overcome ontological heterogeneity: Challenges from financial information systems. *Twenty-Third International Conference on Information Systems, ICIS*.

Friedman, M., Levy, A., & Millstein, T. (1999). Navigational plans for data integration. *AAAI/IAAI*, 67-73.

Goh, C. H. (1996). *Representing and reasoning about semantic conflicts in heterogeneous information sources.* Phd, MIT, Massachusetts Institute of Technology, Sloan School of Management.

Goh, C. H., Bressan, S., Siegel, M., & Madnick, S. E. (1999). Context interchange: New features and formalisms for the intelligent integration of information. *ACM Transactions on Information Systems, 17*(3), 270-293.

Gruber, T. (1993). A translation approach to portable ontology specifications. *Knowledge acquisition, 5*(2), 199-220.

Giunchiglia, F., Yatskevich, M., & Giunchiglia, E. (2005). Efficient semantic matching. *In A. Gomez-Perez and J. Euzenat, editors, ESWC 2005, volume LNCS 3532*, 272-289. Springer-Verlag.

Hass, L, & Lin, E. (2002). IBM federated database technology. Retrieved August 01, 2005, from http://www-106.ibm.com/developerworks/db2/library/techarticle/0203haas/0203haas.html.

Kalfoglou, Y., & Schorlemmer, M. (2003). Ontology mapping: The state of the art. *The Knowledge Engineering Review,* 18(1), 1-31.

Le, B. T., Dieng-Kuntz, R., & Gandon, F. (2004). On ontology matching problems for building a corporate Semantic Web in a multi-communities organization. *In ICEIS 2004 Software Agents and Internet Computing,* 236-243.

Litwin, W., Mark, L., & Roussoupoulos, N. (1990). Interoperability of multiple autonomous databases. *ACM Computing Surveys, 22*(3), 267-293.

McCann, R., Doan, A., Varadarajan, V., & Kramnik, A. (2003). Building data integration systems via mass collaboration. *International Workshop on the Web and Databases.* California: WebDB

Medcraft, P., Schiel, U., & Baptista, P. (2003). DIA: Data integration using agents. *Databases and Information Systems Integration. ICEIS,* 79-86.

Mena, E., Kashyap, V., Sheth, A., & Illarramendi, A. (2000). Observer: An approach for query processing in global information systems based on interoperation across pre-existing ontologies. *Kluwer Academic Publishers,* Boston, (pp. 1-49).

Ozsu, M. T., & Valduriez, P. (1999). *Principles of distributed database systems*, (2nd ed.), Prentice Hall.

Preece, A., Hui, K., Gray, A., Jones, D., & Cui, Z. (1999). The KRAFT architecture for knowledge fusion and transformation. *In 19th SGES International Conference on Knowledge-based Systems and Applied Artificial Intelligence (ES'99).* Berlin: Springer.

Rahm, E., & Bernstein, A. (2001). A survey of approaches to automatic schema matching. *VLDB Journal 4*(10), 334-350. New York: Springer-Verlag Inc.

Shvaiko, P., & Euzenat, J. (2005). A survey of schema-based matching approaches. *Journal of Data Semantics, IV,* 146-171.

Sheth, A. P., & Larson, J. A. (1990). Federated database systems for managing distributed, heterogeneous and autonomous databases. *ACM Computing Surveys, 22*(3), 183-236.

Stephens, L., Gangam, A., & Huhns, M. (2004). Constructing consensus ontologies for the Semantic Web: A conceptual approach. *In World Wide Web: Internet and Web Information Systems, 7,* 421-442. Kluwer Academic Publishers.

Wache, H., Vögele, T., Visser, U., Stuckenschmidt, H., Schuster, G., Neumann, H., & Hübner, S. (2001). Ontology-based integration of information - A survey of existing approaches. *In Proceedings of IJCAI-01 Workshop: Ontologies and Information Sharing,* Seattle, WA, (pp. 108-117).

Woelk, D., Cannata, P., Huhns, M., Shen, W., & Tomlinson, C. (1993). Using Carnot for enterprise information integration. *Second International Conf. Parallel and Distributed Information Systems,* 133-136.

Zamboulis, L., & Poulovassilis, A. (2006). Information sharing for the Semantic Web — A schema transformation approach. *In Proceedings of* DISWeb06, *CAiSE06 Workshop Proceedings,* 275-289.

KEY TERMS

Data Integration: Process of unifying data that share some common semantics but originate from unrelated sources.

Distributed Information System: A set of information systems physically distributed over multiple sites, which are connected with some kind of communication network.

Federated Database: Idem FIS, but the information systems only involve databases (i.e. structured sources).

Federated Information System (FIS): A set of autonomous, distributed and heterogeneous information systems, which are operated together to generate a useful answer to users.

Heterogeneous Information System: A set of information systems that differs in syntactical or logical aspects like hardware platforms, data models or semantics.

Ontological Reusability: The ability of creating ontologies that can be used in different contexts or systems.

Ontological Changeability: The ability of changing some structures of an information source without producing substantial changes in the ontological components of the integrated system.

Ontological Scalability: The ability of easily adding new information sources without generate substantial changes in the ontological components of the integrated system.

Ontology: It provides a vocabulary to represent and communicate knowledge about the domain and a set of relationship containing the terms of the vocabulary at a conceptual level.

Ontology Matching: the process of finding relationships or correspondences between entities of different ontologies.

Semantic Heterogeneity: Each information source has a specific vocabulary according to its understanding of the world. The different interpretations of the terms within each of these vocabularies cause the semantic heterogeneity.

ENDNOTES

[1] This work is partially supported by the UNCOMA project 04/E059.

[2] http://www.ontologymatching.org

Chapter LII
Current Approaches and Future Trends of Ontology-Driven Geographic Integration[1]

Agustina Buccella
Universidad Nacional del Comahue, Argentina

Alejandra Cechich
Universidad Nacional del Comahue, Argentina

INTRODUCTION

Currently there are many domain areas in Computer Science interested in the integration of various information sources. Federated Databases, Semantic Web, and Automated Web Services are some of them. Particularly in the geographic information area, newer and better technologies and devices are being created in order to capture a large amount of information about Earth. All of this geographic information is analyzed and stored at various levels of detail in Geographic Information Systems (GISs), possibly distributed on the Web. Then a fast search for geographic information on the Web will return several links representing parts of our world. But what happens when someone needs information that is divided into more than one system? For example, information about rivers in a country can be obtained by querying two or more systems. Although distribution of information is one of the main problems, there are some others; these systems have been developed by various entities with different points of view and vocabularies, and here is when face heterogeneity problems arise. They are encountered in every communication between interoperating systems where interoperability refers to interaction between information

from various sources involving the task of *data integration* to combine data.

Generally speaking, two essential tasks are involved in the data integration process: *semantic enrichment* and *mapping discovery* (Sotnykova, Vangenot, Cullot, Bennacer & Aufaure, 2005). The main goal of the first one is to reconcile semantic heterogeneity so it involves adding more semantic information about the data. Various proposed approaches add extra semantic information through the use of metadata or ontologies. For example, proposals extending common data models such as Entity-Relationship diagrams (Parent, Spaccapietra & Zimányi, 1999, 2005) and object-oriented ones (Borges, Davis & Laender, 2001) have been presented in order to add geographic features. We are particularly interested in those using ontologies because, by definition, they provide a vocabulary to represent and communicate knowledge about the domain and a set of relationships containing the terms of the vocabulary at a conceptual level. Ontologies are currently extensively proposed as tools to face heterogeneity problems; for example, different proposals are using formal ontologies to enrich the conceptual schema and thus improve the integration (Fonseca, Davis & Camara, 2003; Fonseca, Egenhofer, Agouris & Camara, 2002; Hakimpour, 2003; Hakimpour & Geppert 2002) and the query process (Zhang, 2005).

The semantic enrichment task is essential to reach the second task, *mapping discovery*. Several surveys (Kalfoglou & Schorlemmer, 2003; Klein, 2001) have been presented that analyze various proposals related to semantic matching (i.e., building of mappings); however, they do not take into account geographic information. Considering the hypothesis tested by Mark, Skupin, and Smith (2001) in which "geographic and non-geographic entities are ontologically distinct in a number of ways," a different analysis must be performed when geographic elements are included.

In this chapter, we analyze several proposals that consider geographic information as sources to be integrated. First, we briefly describe basic concepts and conventions that will be used throughout this chapter. Following, an analysis is performed according to the use of ontologies in an integration process of geographic sources. Finally, future trends and conclusions are revealed as a consequence of our analysis.

GIS INTEGRATION: BASIC CONCEPTS

The concept of *data integration* is concerned with unifying data that share some common semantics but originate from unrelated sources. In every integration process, heterogeneity is one of the most common problems. Let us consider two systems sharing data representing rivers, which can be an example to clarify different types of heterogeneity problems (Hakimpour, 2003):

- **Heterogeneity in the conceptual model.** One system represents a river as an object class and the other as a relationship.
- **Heterogeneity in the spatial model.** Rivers can be represented by polygons (or a segment of pixels) in one system, while they are represented by lines in the second system.
- **Structure or schema heterogeneity.** Both systems hold the name of a river, but one of them also keeps information about the border.
- **Semantic heterogeneity.** One system may consider a river as a natural stream of water larger than a creek with a border, and the other defines a river as any natural stream of water reaching from the sea, a lake, and so forth, into the land.

In order to solve several of these heterogeneities, we briefly introduce the concept of *ontologies* as "a formal explicit specification of a shared conceptualization" (Gruber, 1993). A *conceptualization* refers to an abstract model of

how people commonly think about a real thing in the world (e.g., a chair). *Explicit specification* means that the concepts and relations of the abstract model receive explicit names and definitions; a *shared conceptualization* means that a community accepts knowledge described in the ontology. Besides, as ontologies should be formal (a formal explicit specification), a logical formalism is needed to represent them. Thus, ontologies appear to provide semantics to the real world, allowing modelers to define a set of knowledge terms, semantic interconnections, and rules of inference on a particular domain.

In this context, the problem of integrating different sources of information (e.g., databases, pages, etc.) is shifted to integrating different ontologies corresponding to these sources or domains. *Ontology merging* and *ontology mapping* (Klein, 2001) are two of the main tasks within ontology integration. The former creates a new ontology from two or more existing ontologies with overlapping parts, which can be either virtual or physical, while the latter relates similar (according to some metric) concepts or relations from different sources to each other by an equivalence relation. A mapping results in a virtual integration.

As our work is focused on geographic information, the concepts previously mentioned are analyzed according to how this kind of information is used and mapped. The next section further describes some aspects of this integration process.

A CHARACTERIZATION OF CURRENT APPROACHES

In this section, 11 approaches are compared, taking into account a set of features we considered necessary in the integration of geographic information. We have split the comparison into ontological (Table 1) and integration (Table 2) features to facilitate discussion. Besides, each of these features analyzes the proposals by taking into account its own characteristics.

We divided this section into two subsections in order to clarify the analysis. The following subsection provides a description of the meaning of the features (and its characteristics) used to compare the proposals; and the last subsection describes the proposals according to these features.

Description of the Main Characteristics in the Integration of Geographic Information

The first feature to be analyzed in our characterization is focused on the ontologies; thus, the *ontological feature* analyzes how each proposal makes use of the ontologies**.** To do so, three characteristics are considered and compared: the use of formal ontologies (Formal Ontology), the expressiveness of the ontologies (Expressiveness), and the way they are applied (Top-Level Ontology).

By considering the *formal ontology* characteristic, we analyze if the ontologies are formal or nonformal. We assume here that only those ontologies represented by using a formal (or logic) language are considered formal ontologies. Obviously, only with this type of ontologies can inferences be calculated; that is, reasoners can be used to infer implicit relations. Otherwise, nonformal ontologies are those represented by using other structures or data models. In this case, other strategies different from reasoners are used to find relations among ontologies.

By considering the *expressiveness* characteristic, we evaluate the expressiveness of the ontologies. For example, ontologies that are only represented by a taxonomy using super/subclass relations are considered with a lesser degree of expressiveness than those in which other types of associations are allowed. Thus, a *"Low"* scoring means that the ontologies are only taxonomies in which information about generalization/spe-

cialization relations is modeled. A *"Medium"* scoring means that besides these relations, other relations are possible, such as whole-of or part-of relations. Finally, a *"High"* scoring means that there is no restriction in the representation of the ontologies; that is, all previous relations can be used together with others defined by the user (e.g., *isOwner* relation between *Building* and *Owner* entities).

By considering the *top-level ontology* characteristic, the ontological components are analyzed in order to evaluate how they are created and the number of resulting components; for example, if only source ontologies are built, or if there is also a top-level ontology, or if other types of ontologies are necessary to find mappings, and so forth. Let us further discuss a case. Consider a top-level ontology that is an ontology with general terms and minimum constraints and that allows committing to a common vocabulary. A top-level ontology is usually used to create source ontologies. Thus, these source ontologies are built taking into account the elements of source schemas and a top-level ontology. Then, when the integration process starts, the mapping discovery task among the source ontologies will be easier because they manage the same vocabulary and constraints (although each source ontology specializes them). However, the top-level ontology has a main drawback: its independence. That is, systems with previously created ontologies cannot be part of our integrated system (unless we build the source ontologies again). For example, a top-level ontology approach cannot be applied to the Internet community.

The second feature is with respect to *integration features*. Here, an analysis about whether the construction of mappings among the ontologies is a manual process or whether it can be performed semi- or completely automatically is performed. We considered that a manual process is a process in which there is no tool or method to create mappings among ontologies. An expert user must create them by browsing the source ontologies. When we talk about semi- or completely automatic processes, we are considering that a method is proposed in order to calculate similarities; for example, by using a set of similarity functions. Besides, in this last case, we evaluate the tools used to do so. For example, reasoning capabilities (inferences) can be applied or another comparison can be made, such as a syntactic comparison of the elements of the ontologies by using a set of similarity functions.

Proposals Mapped to our Main Characteristics

Table 1 shows the first dimension, *ontological features*, applied to the proposals. As we can see, various combinations of the first two characteristics are possible. For example, proposals as MDSM (Rodríguez & Egenhofer, 2004), MDSM+TR (Janowicz, 2005), and ODGIS (Fonseca, 2001) represent the ontologies by using a structure in which parts, functions, and attributes are defined. It means that the ontologies are not formal because each element in the structure can be described with natural language instead of a formal definition. Thus, a value of "No" is set for these proposals. Following, for the two first proposals, the expressiveness of these ontologies is Medium due to relations as meronym (denoting constituent parts of or members of something) and hypernym (denoting especialization/generalization relationships such as a taxonomy) can be represented. In the case of ODGIS, the expressiveness is Low because only taxonomies are allowed. The same happens with Aerts, Maesen, and van Rompaey (2006) and Hakimpour (2003); however, the ontologies are formal here. On the other hand, the proposals by BUSTER (Visser, 2004), Kavouras, Kokla, and Tomai (2003), and Schwering and Raubal (2005) use different ways to model the ontologies, such as a rule-based mediator in the first one, with a Medium level of expressiveness. Finally, the Sotnykova, et al. (2005) proposal is scored with a High value on the second characteristic because

Table 1. The ontology feature applied to the proposals

Ontologies			
	Formal Ontology	**Expressiveness**	**Top-Level Ontology**
BUSTER	Yes	Medium	No
Kavouras, et al. (2003)	No	Medium	No
Schwering and Raubal (2005)	No	Medium	No
Sotnykova, et al. (2005)	Yes	High	No
Hakimpour (2003)	Yes	Low	Yes
Hess & Iochpe (2002)	No	Medium	Yes
MDSM	No	Medium	No
ODGIS	No	Low	Yes
MDSM+TR	No	Medium	No
GeoNis	Yes	Medium	Yes
Aerts, et al. (2006)	Yes	Low	Yes

all capabilities formal languages provide can be applied to model the ontologies.

The third characteristic of Table 1, top-level ontology, can be analyzed together with the Inferences column in the Integration Process feature (Table 2). Several proposals use a top-level ontology as a mediator among the source ontologies, and thus, a reasoner can be used to calculate inferences (similarities) among them. For example, proposals such as the Hakimpour (2003), GeoNis (Stoimenov, Stanimirovic & Djordjevic-Kajan, 2006), and Aerts, et al. (2006) construct source ontologies based on a committed top-level ontology previously defined, and a reasoner is applied. Something similar happens to the proposal by Hess and Iochpe (2004), although a reasoner is not applied because ontologies here are not formal. A different use is implemented by the ODGIS approach because more than one ontology is built in order to provide more information about the domain and thus facilitate the integration process. But the activity of creating these ontologies is not an easy task. The other proposals do not use top-level ontologies and consequently do not use reasoning capabilities to calculate similarities. To do so in general, a process is proposed that applies a set of functions.

Following, we analyze other things: Does the method consider that the information modeled is geographic? Is there any difference in the integration method if the information is not geographic? In which way is the geographic information considered or represented? We analyze the representation of geographic information in the first column of Table 2. There are several proposals—MDSM, ODGIS, and Hakimpour (2003)—which do not provide additional elements to represent geographic information. However, other proposals such as BUSTER, GeoNis, and Kavouras, et al. (2003) provide a set of topological relations to model source ontologies. In the BUSTER approach, qualitative models are used at the data level to retrieve instances of the sources. Only the proposal by Sotnykova, et al. (2005) provides a full representation allowing models to represent spatio-temporal features. They propose a MADS (Modeling of Application Data with Spatio-temporal features) model allowing to manipulate geographic information through multiple perspectives of the same information. This model provides an advantage with respect

Table 2. The integration process and geographic information feature applied to the proposals

	Geographic Information	Integration Process	
		Manual	Inferences
BUSTER	Quality models	Yes	Yes
Kavouras, et al. (2003)	Topological relations	No	No
Schwering and Raubal (2003)	Topological relations	No	No
Sotnykova, et al. (2005)	MADS models	Yes	No
Hakimpour (2003)	No	No	Yes
Hess and Iochpe (2002)	No	No	No
MDSM	No	No	No
ODGIS	No	No	No
MDSM+TR	No	No	No
GeoNis	Topological relations	No	Yes
Aerts, et al. (2006)	Topological relations	No	Yes

to the others because in addition to a formal language being used to represent it, aspects relating to spatial features are also considered.

With respect to the *integration process*, several proposals represent the ontologies by using a formal language in order to take advantage of the inference mechanisms. But in general, source ontologies must commit to the same top-level ontology to allow the reasoning system to start the integration process. Thus, source ontologies are not independent because various communities must agree with the top-level ontology. For example, proposals by GeoNis, Aerts, et al. (2006), and Hakimpour (2003) use a top-level ontology together with the advantages of a formal language as tools to find more suitable mappings. However, we can find proposals in which they define formal ontologies without applying any inference. An example of this is the proposal by Sotnykova, et al. (2005). It is the only one that allows representing the ontologies as formal ones and by using geographical features. But although a logic language is used, this proposal does not take advantage of it for finding similarities.

Other proposals such as Schwering and Raubal (2005) and Hess and Iochpe (2002) involve a set of functions that analyzes the schema syntactically and semantically. These functions are applied to the source ontologies in order to find mappings among them. The use of these types of functions is useful when the ontologies are not complete (i.e., there is absent information about the domain) and/or as a starting point of an integration process when a top-level ontology is not involved. On one hand, when the ontologies are not complete, a syntactic function could find a mapping between two elements of two ontologies, which would not be found by a reasoner otherwise. In addition, those methods that use a thesaurus to find synonym relations are also useful in this case.

Finally, the proposals of BUSTER, GeoNis, and Sotnykova, et al. (2005) require an expert user to perform the integration process; that is, an expert user is responsible of creating mappings among sources. In the case of BUSTER, the proposal uses inferences during the query process.

FUTURE TRENDS

Several systems compared here are still in a development stage, and as we have explained,

some problems should be solved in order to reach a good integration. In particular, we have presented integration methods based on geographical characteristics of the sources. Among the various proposals, there is a set of mechanisms in common about how to build an integrating system. For example, the use of ontologies is one of them. However, there are some aspects of such mechanisms that have to be taken into account in order to reach a real integration. The following are some examples:

- **Representation of the Geographic Information.** The representation of the geographic information is one of the main aspects to be considered. However, few proposals implement mechanisms to model this information, and then it is not taken into account by the integration process. At this point, we should consider the following hypothesis tested by Mark, et al. (2001): "Geographic and non-geographic entities are ontologically distinct in a number of ways." Their experiments tested the degree to which ordinary people can code the geographic domain at a conceptual level. As a conclusion of this study, there are geographic terms that have higher frequency; that is, they are more recurrent terms. In principle, by knowing what these terms are, the integration process could be simplified. Relating to this item, the expressiveness of the ontologies is also a very important aspect because their elements must represent concepts in the real world. For example, if we can only represent taxonomies, several features about concepts will be absent. Obviously, aspects such as decidability and computational resources must be taken into account when more expressiveness is added and also when the ontologies are large.
- **Formal Representation of Ontologies.** With respect to the formal representation of the ontologies, several proposals have applied formal mechanisms in order to improve both the integration process and the evolution of the integrated system. Although some proposals have used top-level ontologies, which interfere with the independence of the system, all capabilities of a logic language are applied. Thus, inferences such as consistency checking and subsumption of concepts are taken into account. In addition, new research works have been published in order to give more inference capabilities. Thereby, more implicit relations among the ontologies can be found, and spatial reasoning (Cohn & Hazarika, 2001} for the query process can be applied (Zhang, 2005). Nevertheless, there is a lot of work to do in this area. New inference capabilities, performance, and even automatic mapping discovery are still pending tasks.
- **The Integration Process.** In general, the integration proposals described here and those in which sources do not involve geographic information describe a set of steps to find suitable mappings among sources. The elements of the ontologies are compared to each other by considering the represented semantic knowledge. But in particular, when geographic information is represented, processes cannot be the same. Geographic objects, fields, quantitative and qualitative relations, spatio-temporal variation, and scale are new concepts that must be taken into account. For example, a very common feature as granularity in geographic models should be considered and analyzed when mappings are looked for. Two ontologies with different granularities will not have elements in common if mechanisms to solve it are not implemented (Fonseca, 2001). Therefore, integration methods should add new techniques to consider these new features.

Similarly, all these concerns will depend on the source ontologies. There are more than a thousand works that talk about ontologies and represent ontologies in various ways. But how can we evaluate the ontology's quality? How can we determine whether an ontology is complete? Is it representing the real world? If we work with low-quality ontologies, any integration method, even using reasoners or similarity functions, will produce unsatisfactory results.

CONCLUSION

In this work, several proposals of integration of geographic information have been analyzed according to two main aspects: the use of ontologies and the way the integration process is performed.

Several open issues are still under development with respect to the representation of ontologies and, particularly, when geographic information is involved. The geographic information has to be taken into account because particularities of representation of geographic objects can be useful in both the integration and query processes.

With respect to the integration process, some conclusions of our analysis must be taken into account. We think that the use of formal ontologies and consequently the use of reasoners should be mandatory. When the ontologies are formal ones, represented by a logic language, several advantages are inherent. For example, reasoning capabilities such as consistency checking, subsumption of concepts, and detection of cardinality restrictions can be taken into account to perform the integration process. In addition, the addition of a new information source or new information of an existing source will be easier than in other systems because aspects such as redundancy and inconsistency would be automatically checked.

Similarly, all integration processes need user interaction in some parts of the whole process. In general, it is not possible to determine fully automatically all mappings among ontologies, and thus, an expert user must be involved. For example, when inconsistencies are found in the ontologies, an expert user is responsible of solving them.

REFERENCES

Aerts, K., Maesen, K., & van Rompaey, A. (2006). *A practical example of semantic interoperability of large-scale topographic databases using semantic Web technologies.* Proceedings of the AGILE'06: 9th Conference on Geographic Information Science, Visegrd, Hungary, 35–42.

Borges, K., Davis, C., & Laender, A. (2001). OMT-G: An object-oriented data model for geographic applications. *Geoinformatica, 5,* 221–260.

Cohn, A.G., & Hazarika, M. (2001). Qualitative spatial representation and reasoning: An overview. *Fundamenta Informaticae, 46*(1-2), 1–29.

Fonseca, F. (2001). *Ontology-driven geographic information systems* [doctoral thesis]. University of Maine.

Fonseca, F., Davis, C., & Camara, C. (2003). Bridging ontologies and conceptual schema in geographical information integration. *Geoinformatica, 7,* 307–321.

Fonseca, F., Egenhofer, M., Agouris, P., & Camara, C. (2002). Using ontologies for integrated geographic information systems. *Transactions in GIS, 3*(6).

Gruber, T. (1993). A translation approach to portable ontology specifications. *Knowledge Acquisition, 5*(2), 199–220.

Hakimpour, F. (2003). *Using ontologies to resolve semantic heterogeneity for integrating spatial database schemata* [doctoral thesis]. Zurich University.

Hakimpour, F., & Geppert, A. (2002). *Global schema generation using formal ontologies.* Proceedings of the ER2002, LNCS 2503, 307–321.

Hess, G.N., & Iochpe, C. (2002). *Ontology-driven resolution of semantic heterogeneities in GDB conceptual schemas.* Proceedings of the GEOINFO'04: VI Brazilian Symposium on GeoInformatics, Campos do Jordao, 247–263.

Janowicz, K. (2005). *Extending semantic similarity measurement by thematic roles.* Proceedings of the GeoS'05: First International Conference on GeoSpatial Semantics, Mexico City, Mexico, 137–152.

Kalfoglou, Y., & Schorlemmer, M. (2003). Ontology mapping: The state of the art. *The Knowledge Engineering Review, 18*(1), 1–31.

Kavouras, M., Kokla, M., & Tomai, M. (2003). Comparing categories among geographic ontologies. *Computers & Geosciences, Special Issue, Geospatial Research in Europe: AGILE 2003, 31*(2), 145–154.

Klein, M. (2001). *Combining and relating ontologies: An analysis of problems and solutions.* Proceedings of the IJCAI-2001 Workshop on Ontologies and Information Sharing, Seattle, Washington.

Mark, D.M., Skupin, A., & Smith, B. (2001). Features, objects, and other things: Ontological distinctions in the geographic domain. In *Spatial information theory: Foundations of geographic information science* [Lecture Notes in Computer Science (2205)] (pp. 488–502). Berlin: Springer-Verlag.

Parent, C., Spaccapietra, S., & Zimányi, E. (1999). *Spatio-temporal conceptual models: Data structures + space + time.* Proceedings of the GIS '99: 7th ACM International Symposium on Advances in Geographic Information Systems, New York, 26–33.

Parent, C., Spaccapietra, S., & Zimányi, E. (2005). The murmur project: Modeling and querying multi-representation spatio-temporal databases. *Information Systems.*

Rodríguez, M.A., & Egenhofer, M.J. (2004). Comparing geospatial entity classes: An asymmetric and context-dependent similarity measure. *International Journal of Geographical Information Science, 18*(3), 229–256.

Schwering, A., & Raubal, M. (2005). *Spatial relations for semantic similarity measurement.* Proceedings of the *ER'05* Perspectives in Conceptual Modeling, 3770, 259–269.

Sotnykova, A., Vangenot, C., Cullot, N., Bennacer, N., & Aufaure, M. (2005). Semantic mappings in description logics for spatio-temporal database schema integration. *Journal on Data Semantics, III,* 143–167.

Stoimenov, L., Stanimirovic, A., & Djordjevic-Kajan, S. (2006). *Discovering mappings between ontologies in semantic integration process.* Proceedings of the AGILE'06: 9th Conference on Geographic Information Science, Visegrd, Hungary, 213–219.

Visser, U. (2004). *Intelligent information integration for the semantic Web.* Volume 3159. Lecture Notes in Computer Science. Berlin-Heidelberg: Springer.

Visser, P., Jones, D., Bench-Capon, T., & Shave, M. (1997). *An analysis of ontology mismatches; heterogeneity versus interoperability.* Proceedings of the AAAI 1997 Spring Symposium on Ontological Engineering.

Wache, H., et al. (2001). *Ontology-based integration of information—A survey of existing approaches.* Proceedings of the IJCAI-01 Workshop: Ontologies and Information Sharing, Seattle, Washington, 108–117.

Zhang, Z. (2005). *Ontology query languages for the semantic Web: A performance evalua-*

tion [master's thesis]. Athens, GA: University of Georgia, Athens.

KEY TERMS

Data integration: Process of unifying data that share some common semantics but originate from unrelated sources.

Federated information system (FIS): A set of autonomous, distributed, and heterogeneous information systems that are operated together to generate a useful answer to users.

Geographic information: Information about objects or phenomena that are associated with a location relative to the surface of Earth. A special case of spatial information.

Geographic information system (GIS): A computer-based system to efficiently model, capture, store, manipulate, query, retrieve, analyze, and visualize information, where part of the information is of a geographic nature. It is generally based on a structured database that describes the world in geographic terms.

Heterogeneous information system: A set of information systems that differs in syntactical or logical aspects, such as hardware platforms, data models, or semantics.

Ontology: Provides a vocabulary to represent and communicate knowledge about the domain and a set of relationships containing the terms of the vocabulary at a conceptual level.

Ontology mapping: The process of relating similar (according to some metric) concepts or relations from various sources to each other by an equivalence relation. A mapping results in a virtual integration.

Ontology merging: The process of generating the creation of a new ontology from two or more existing ontologies with overlapping parts, which can be either virtual or physical.

Semantic heterogeneity: Each information source has a specific vocabulary according to its understanding of the world. The different interpretations of the terms within each of these vocabularies cause the semantic heterogeneity.

Spatial data: Any information about the location and shape of, and relationships among geographic features. This includes remotely sensed data as well as map data.

ENDNOTE

[1] This work is partially supported by the UNCOMA project 04/E059.

Chapter LIII
Mediation and Ontology-Based Framework for Interoperability

Leonid Stoimenov
University of Nis, Serbia

INTRODUCTION

Research in information systems interoperability is motivated by the ever-increasing heterogeneity of the computer world. New generations of applications, such as geographic information systems (GISs), have much more demands in comparison to possibilities of legacy information systems and traditional database technology. The popularity of GIS in governmental and municipality institutions induce increasing amounts of available information (Stoimenov, Đorđević-Kajan, & Stojanovic, 2000). In a local community environment (city services, local offices, local telecom, public utilities, water and power supply services, etc.), different information systems deal with huge amounts of available information, where most data in databases are geo-referenced. GIS applications often have to process geo-data obtained from various geo-information communities. Also, information that exists in different spatial database may be useful for many other GIS applications. Numerous legacy systems should be coupled with GIS systems, which present additional difficulties in developing end-user applications.

But, information communities find it difficult to locate and retrieve data from other sources in a reliable and acceptable form. In such systems, reuse for geo-data is very often a difficult process because of poor documentation, obscure semantics, diversity of data sets, and the heterogeneity of existing systems in terms of data modeling concepts, data encoding techniques, and storage structures (Devogele, Parent, & Spaccapietra, 1998). Also, available information is always distributed and no one wants to share their own information in public without commitment.

The problem of bringing together heterogeneous and distributed information systems is known as interoperability problem. Today, research on interoperability solutions promises a way to move away the monolithic systems that dominate the GIS market (Sondheim et al., 1999). In that kind of environment, there often arises the problem of semantic heterogeneity and the correctness of the interpretation of data sets obtained from different geo-information communities.

Domain experts use the concepts and terminology specific for their respective field of expertise, and different parameters and different languages

to express their model of a concept. Therefore, very often different data sets can use different terms for the same kind of information. On the other hand, different data sets can use the same term for a completely different piece of information. Humans use their common sense, that is, their knowledge about the world, to translate the meaning of a foreign set of concepts and terms into their own terminology. Software systems usually do not have any knowledge about the world and have to explicitly be told how to translate one term into another. These problems can lead to serious conflicts during discovering and interpreting geographic data.

BACKGROUND

Heterogeneity of Data Sources

The realization of interoperable information systems is a weighty process involving two main system characteristics: distributed data sources and their heterogeneity. Information systems heterogeneity may be considered as structural (schematic heterogeneity), semantic (data heterogeneity), and syntactic heterogeneity (database heterogeneity; Bishr, 1998). Syntactic heterogeneity means that various database systems use different query languages (SQL [structured query language], OQL, etc.). Structural heterogeneity means that different information systems store their data in different structures. Semantic heterogeneity considers the content of an information item and its meaning. Semantic conflicts among information systems occur whenever information systems do not use the same interpretation of the information. Stuckenschmidt, Wache, Vogele, and Vissar (2000) give an introduction to problems concerning the syntactic, structural, and semantic integration. This article also presents technologies for enabling interoperability.

Information Integration

A number of proven and well-established methods exist that allow heterogeneous data sources (databases) to communicate on a technical level. Some of these methods include federated databases and schema integration (Larson, 1998), object-oriented approaches (Chawathe et al., 1994), data warehousing (Matousek, Mordacik, & Janku, 2001; Voisard & Juergens, 1998), and mediators and ontologies (Boucelma & Colonna, 2005; Fonseca & Egenhofer, 1999; Stoimenov et al., 2000).

The data warehousing approach (Voisard & Juergens) implies accumulation of spatial data in a few well-defined and tightly connected data stores, where information integration is precomputed. While efficient for a relatively small number of core spatial data sets, this approach is not readily extensible to a larger number of data sets with semistructured and ad hoc data. Mediator-based systems, alternatively, are constructed for a large number of relatively autonomous sources of data and services communicating with each other over a standard protocol and enabling on-demand information integration (Wiederhold, 1998). Structural and syntactic heterogeneity may be solved by mediation.

Information mediators are originally developed for integrating information in databases. Wiederhold defines a mediator and he is the first that points out the need for mediation for contemporary database systems. Wiederhold therefore sees a mediator as an intermediate abstraction layer between databases and applications that use them. Mediation is primarily an architectural concept, and so the precise implementation of a module is less important. The mediator architecture is therefore introduced as a three layer system: (a) applications, (b) mediators, and (c) DBMS. The three-level architecture of mediator-based systems is constructed of an application layer and a large number of information sources (heterogeneous data sources with wrappers) communicating with each other over a standard protocol.

Mediator architecture provides a transparent view of data sources and the independence of data sources and user applications (Stoimenov, Mitrovic, Đorđević-Kajan, & Mitrovic, 1999). However, mediators are not just a simple interface between applications and databases: They include knowledge that does not exist in the data they work with.

Also, underlying data has to be aggregated in them, depending on criteria dictated by the application layer. The most important fact is that a data integration system lets users focus on specifying what they want rather than how to obtain the answers. As a result, it frees them of combining data from multiple sources, interacting with each source, and finding the relevant sources. Applying the mediator framework to the intranet and Internet environment solves the difficult problem of gaining access to real-world data sources. Internet provides the underlying communication layer and protocols for mediation of distributed systems.

Standardization

In the past few years, the OpenGIS Consortium (OGC; *Open GIS Consortium*, 2006), as a consortium of GIS vendors, agencies, and academic institutions, has emerged as a major force in the trend to openness. The OGC introduced approaches for effective management of interoperability technology in cooperation with industry and universities. The OGC has identified the need for open geo-data sharing and the exchange of open GIS services.

Despite standardization initiatives, the use of standards as the only worthwhile effort to achieve interoperability is not widely accepted. Since heterogeneity arises naturally from a free market of ideas and products, there is no way for heterogeneity to be banished from standards by decree. As a consequence of two main system characteristics, distributed data sources and their heterogeneity, the realization of interoperability systems is a weighty process.

Interoperability

In general, interoperability refers to the ability of two or more systems or components to exchange information and use the information that has been exchanged (Institute of Electrical and Electronics Engineers [IEEE], 1990). Interoperability of information systems implies information integration and relies on the basis of agreement that describes what is shared among information sources. Interoperability means openness in the software industry because open publication of internal data structures allows GIS users to build applications that integrate software components from different developers. Interoperability also means the ability to exchange data freely between systems because each system would have knowledge of other systems' formats.

To enable interoperability, remote systems must be able not only to locate and access data sources, but also to interpret and process retrieved data. In order to achieve this, remote systems have to deal not only with syntactically heterogeneous data objects (objects that are organized following different conceptual schemas), but also with semantically heterogeneous objects (objects that have different meaning; Stoimenov, Stanimirović, & Đorđević-Kajan, 2004; Visser & Stuckenschmidt, 2002).

At least three levels of interoperability based on the required knowledge must be considered (Obrst, 2003). Communication interoperability relies on the infrastructure and standardized protocols where all the necessary data are precisely defined (e.g., encapsulation, frames, checksum algorithms, etc.). No machine knowledge is required, but it may be deployed, for instance, using intelligent agents for network maintenance to avoid bottlenecks in order to provide a better quality of service, and so forth. Syntactic interoperability allows content exchange among multiple software components independent of their implementation languages, run-time environments, and other technological differences.

Semantic interoperability is a crucial problem in open networked systems where many different

and independent enterprise parties need to cooperate and share heterogeneous information resources, often in response to opportunities or challenges that cannot be anticipated and that require a rapid response (Castano, Ferrara, & Montanelli, 2005). Semantic interoperability may be defined as the knowledge-driven interoperability that provides participants involved in a process with the ability to overcome semantic conflicts arising from differences in implicit meanings, perspectives, and assumptions in the data, processes and the like, that are used in heterogeneous environments (Park & Ram, 2004).

Semantic interoperability is a very complex field, especially in GIS. Interoperability in general and semantic interoperability will lead to dramatic organizational changes in the GI community. In order to achieve semantic interoperability in a heterogeneous information system, the meaning of the information that is interchanged has to be understood across the systems. Accessing heterogeneous and distributed information sources requires appropriate semantic interoperability techniques to enable seamless access and retrieval of the right information resources (Castano et al., 2005). Systems must be able to exchange data in such a way that the precise meaning of the data (i.e., semantics) is readily accessible and the data itself can be translated by any system into a form that it understands.

Ontologies

The use of ontologies as semantic translators is a viable approach to overcome the problem of semantic heterogeneity (Castano et al., 2005; Obrst, 2003; Stoimenov& Đorđević-Kajan, 2005; Wache et al., 2001). According to a common definition, an ontology specifies the conceptualization of the world to which the data in an information system refer (Gruber, 2002). In other words, it provides a frame of reference for the vocabulary used in the system. Ontologies provide machine-readable semantics of information sources. An ontology consists of logical axioms that convey the meaning of terms for a particular community. Logical axioms define concepts and their relations, and also constrains on both. An ontology exists as a consensus of members of a community (Bishr, Pundt, Kuhn, & Radwan, 1999), for example, users of single information system or people in one discipline.

Recently, the use of ontology in information systems was discussed in Guarino (1998) and specifically in GIS building in Devogele et al. (1998) and Laurini et al. (1998); the creation of GIS software components from ontologies was discussed in Fonseca and Egenhofer (1999). Research on ontology is becoming increasingly widespread in the computer science community, and its importance is being recognized in a multiplicity of research fields and application areas, including knowledge engineering, database design and integration, and information retrieval and extraction.

Ontology-Driven Interoperability

A mediator-based system is important for a spatial data interoperability architecture (Hess, Iochpe, & Oliveira, 2004; Stoimenov et al., 2000). Mediators and wrappers are one way of achieving communication between services and data sources without semantic loss or errors (Hess et al.). Hakimpour and Timpf (2001) proposed the use of ontology in the resolution of semantic heterogeneities, especially those found in GISs. The goal was to establish equivalences between conceptual schemas or local ontologies. A reasoning system is used to merge formal ontologies. The result of the merging is used by a schema integrator to build a global schema from local ones.

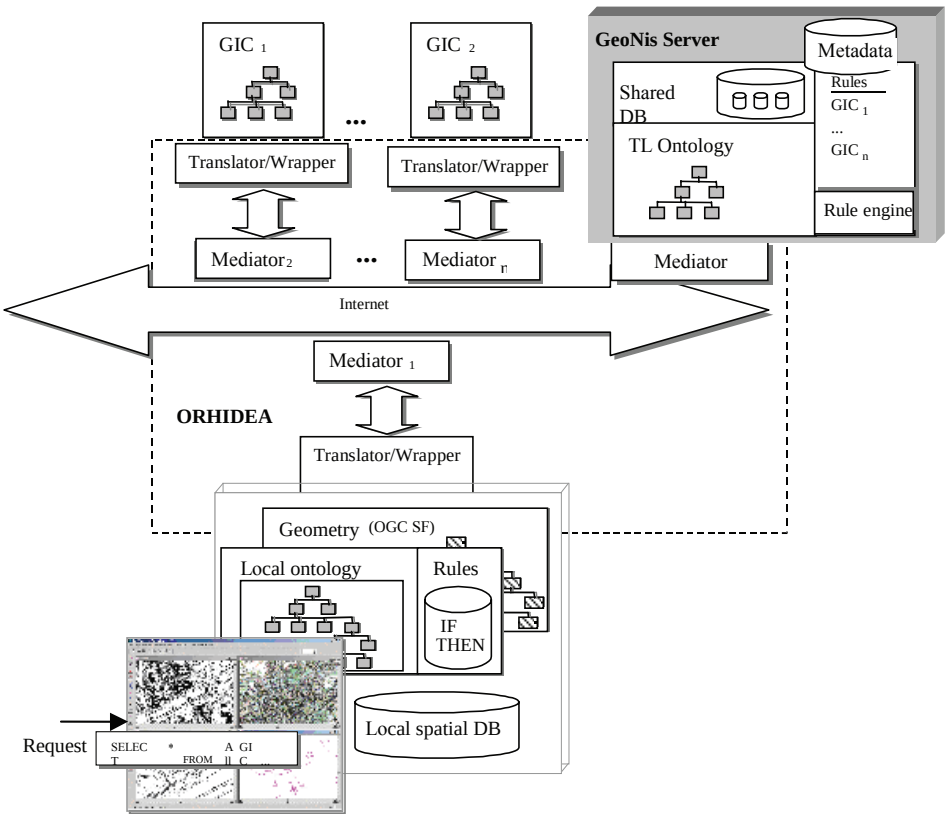
Sotnykova, Vangenot, Cullot, Bennacer, and Aufaure (2005) state that the integration of spatial-temporal information is a three-step process comprising the resolution of syntactic conflicts, interschema correspondence assertions (or resolution of semantic conflicts), and integrated schema generation (resolution of structural conflicts). For semantic conflicts they propose an integration language, which allows formulating correspondences between different database schemas. Matching the

geographical objects based on the matching of their child objects is the proposal of Cruz, Sunna, and Chaudhry (2004). In their approach, there must be a global ontology that is the reference for the alignment of the local ontologies. Alignment is the identification of semantically related entities in different ontologies.

Fonseca, Egenhofer, Davis, and Câmara (2002) proposed an ontology-driven GIS architecture to enable geographic information integration. In that proposal, the ontology acts as a model-independent system integrator. The work of Fonseca, Davis, and Camara (2003) focuses on the application level in which they can work on the translation of a conceptual schema to application ontology. A framework for mapping between ontologies and conceptual schemas defines the mappings between a term in a spatial ontology and an entity in a conceptual schema.

Semantic similarity measures play an important role in information retrieval and information integration (Rodriguez & Egenhofer, 2003). In order to establish correspondence among different representations of the same real-world concept that was defined in different schemas (or ontologies), it is necessary to recognize the common concept through the identification of similarities as well as conflicts among those schemas (Hess et al., 2004). Once ontologies have been integrated, similarity measures are applied to compare concepts. Semantic similarity measurement offers the possibility to define an area of interest and to calculate the distance between the classes within this area (Janowicz, 2005). A recent work presents different measures for comparing concepts whose formal definition supports inferences of subsumption, and local concepts in different ontologies inherit their definitional structures from concepts

Figure 1. GeoNis framework for semantic interoperability



in a shared ontology (Rodriguez & Egenhofer, 2003). The matching-distance similarity measure (MDSM; Janowicz, 2005) is one such (feature-based) measurement theory that was introduced for the geo domain. MDSM is the asymmetric and context-sensitive semantic similarity measurement approach for entity classes developed by Rodriguez and Egenhofer.

ONTOLOGY/MEDIATION FRAMEWORK FOR SEMANTIC INTEROPERABILITY

GeoNis is a generalized framework for the interoperability of GIS applications that have to provide infrastructure for data interchange in the local community environment (Stoimenov & Đorđević-Kajan, 2003). In local community data sources, there are services and offices that own geo-data in some format. Specified communities own GIS applications, often created with different GIS tools and with different underlying database management systems. The goals of research activities in the GeoNis project are the following:

- Defining an interoperability architecture for the integration of distributed and heterogeneous GIS data sources in the local community environment
- Defining a methodology and software support for resolving semantic conflicts in data from different information sources

In order to achieve interoperability in GeoNis, the following six presumptions have to be fulfilled (modified from Levinsohn, 2000).

- **Simple:** Users do not need to understand all details about the data or their source system to import and use them.
- **Transparent:** Complexities associated with data transfer should be hidden from users.
- **Open:** Interoperability should apply to all systems, and data exchange should be independent of the technology used.
- **Equal:** Systems are equal and autonomous.
- **Independence:** Systems have exclusive right to control their information and information processing without centralized control.
- **Effective:** Data transfer should be reliable, and the resultant data should be useful for the intended purposes.
- **Universal:** All geospatial databases should be accessible.
- **Belonging:** Each system belongs to one GI community, and has its own institution, policy, culture, and value viewpoint.

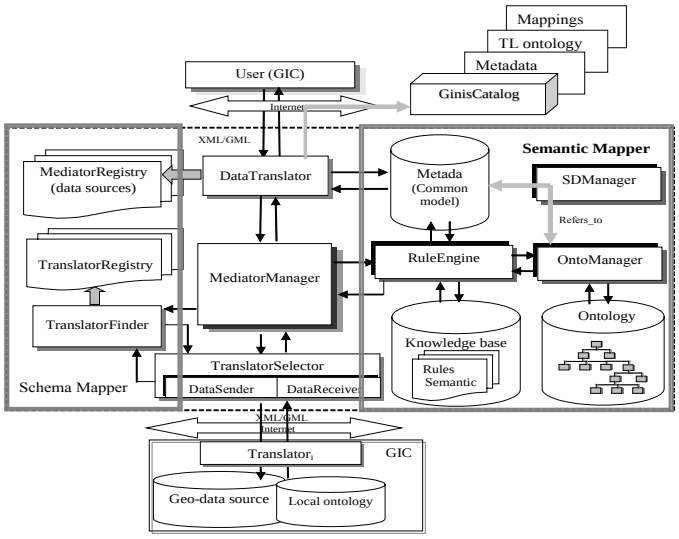
The GeoNis framework implies several different prerequisites.

- **Cooperation between participants.** Participants have knowledge of each other's participants and the data they possess.
- **Institutional willingness for realization of interoperability.** All participants have to agree upon basic principles for realization of interoperability and to provide all needed data and resources.
- **Infrastructure for realization of interoperability.** Network infrastructure (hardware and software), people, organizations and activities, and rules and regulations for information exchange all exist.
- **Definition of communication protocols between participants.**
- **Development of software tools for realization of interoperability.**
- **Local information sources must be adapted in order to work properly in new environment.**

Our framework for interoperability has to provide the following (Stoimenov & Đorđević-Kajan, 2002).

- Integration of information from different sources with ability to add new information sources

Figure 2. Semantic mediator in GeoNis



- Adaptation of existing data sources and queries without possibility for changing existing data
- Independence of user applications from information sources
- Solving problems of semantic inconsistency between user requests and available data

To reach these goals, we proposed a common interoperability kernel called ORHIDEA (Ontology-Resolved Hybrid Integration of Data in E-Applications; Stoimenov & Đorđević-Kajan, 2003). The generic architecture of GeoNis with the ORHIDEA kernel recognizes several different components that have important roles in geo-information discovering and retrieving process (Figure 1).

- **Geographic Information Community (GIC):** In each node of the GeoNis framework there exists a GIS application and corresponding (spatial and nonspatial) data sources. Data in these local data sources are accessible according to user privileges.
- **ORHIDEA Wrapper/Translator:** Component that translates information flow between information source and GeoNis system
- **ORHIDEA Semantic Mediators:** Requests for a specific data set are forwarded through
- **Shared GIS Server (Catalog Server):** Maintains metadata and all shared/common geographic data in addition to domain-oriented GIS applications

The total number of geo-data providers in the local community environment is indeterminable and unlimited. This implies the need for a flexible approach that can deal with the existing and the future geo-data providers in interoperable systems. A standard model for spatial data is the first step to approach the solution for schematic and syntactic heterogeneity. The OGC specification aims to solve the problem of heterogeneity at the spatial data modeling level. Because of that, GeoNis uses the OpenGIS standard as a common data model to represent geo-data at the mediator level. Data models of local information sources are translated into a common model using wrappers.

According to GeoNis architecture, each GIC (i.e., local service or office) contains a GIS application and corresponding (spatial) database. For each data source there is a translator (or wrapper) that logically converts basic data objects

to common information models. The Ginis OLE DB data provider is an example of a translator implementation (Stoimenov et al., 2004). This approach (unified methods for data access) allows a simple chaining of translators. Also, we can easily add new information sources without influencing other GIC environments. In order to do this, we have to provide semantic mapping only for the GIC environment with a new information source.

The next layer (semantic mediator) performs mediator functions, which include transformation of data and mapping between data models.

Architecture and Role of Semantic Mediator

The basic components providing interoperability in the ORHIDEA kernel are semantic mediators. Semantic interoperability in ORHIDEA, resolved by the semantic mediator, is the ability to share geospatial information at the application level without knowing or understanding the terminology of other systems.

The semantic mediator component acts as an access point for a number of independent geo-information sources and allows integration of their information while bridging over the semantic differences among them. The semantic mediator enables users to access multiple information systems as though they were a single system with a uniform way to retrieve information and perform computations. It accepts high-level requests from users and automatically translates them into a series of lower level requests for different GICs.

The semantic mediator architecture and its role in the GeoNis framework are given in Figure 2. The main part of the semantic mediator is MediatorManager. The basic function of this component is to manage the work of other components. It receives the data produced by other mediators (using DataTranslator), initiates the search for the appropriate translator, receives the processed data from the translator, and sends the data to the corresponding mediator (using DataTranslator). TranslatorSelector is the component of mediator that provides communication with the translators. TranslatorSelector sends (through DataSender) and receives (through DataReceiver) data to and from the MediatorManager. Data exchange is done using the standard Internet protocols, XML (extensible markup language) and GML (for more detail about GML, please check the OGC home page at http://www.OpenGIS.org).

MediatorManager initiates the system components that resolve semantic conflicts. Basic components of the semantic mediator for semantic conflict resolution are the following.

- Semantic mapper components that perform mapping between semantically similar data
- Schema mapper components that perform query distribution and reformulation

The semantic mapper is a component that provides access to the shared metadata that reside on the common server. This component implements interfaces that provide means for discovery, access, and management of metadata to the rest of the community. The interface publishing service allows the GIC nodes to register their data and services and to create relations between their local ontologies and top-level ontologies. Every GIC node that needs to publish its geo-information sources must use this service in order to signal other GIC nodes or users that it exists. Only GIC nodes that are registered with this component are visible to the users who can then browse their data and services. The interface query service provides functionality for locating and accessing requested metadata. This interface is used by the semantic mediator component in order to gather information about requested concepts and their location within the system.

The semantic mapper components RuleEngine, OntoManager, and SDManager (SpatialDataManager) participate in resolving semantic conflicts. These components can be a part of the mediator or a part of distinct component that will communicate with the mediator. The components use the metadata, schemas, and knowledge from GinisCatalog.

A local repository contains local ontologies and local schemas describing object sources. The inference module RuleEngine provides inference services that allow more sophisticated queries to be formulated against the catalog feature objects and metadata. Knowledge in the RuleEngine knowledge base is organized as *if-then* rules. Such rules define semantic conflicts in a form that is acceptable for inference processes. RuleEngine can generate rules on user demand, and store them into the local knowledge base (the generated rules are sent to the GinisCatalog server). Also, RuleEngine implements the inference engine that interprets the rules retrieved from the knowledge database.

The schema mapper processes metadata about local data sources that can provide GIS users with requested data. After these information are obtained (concepts from the semantic mapper and information about data sources from the shared server), the schema mapper identifies translators that will be used (using TranslatorFinder and TranslatorRegistry) and that will transform original user requests into a series of small requests. The schema mapper then sends these low-level requests to appropriate translators that can interpret them and can access requested data from local data sources. Results of this low-level request are then combined into a result for the original high-level user request.

The GinisCatalog is a collection of metadata concerning data stored in distributed geo-data sources (geographic databases). These metadata describe the properties of geo-data sources, which can be queried through mediators and translators (namely the ORHIDEA mediator platform). The GinisCatalog stores the following types of metadata.

1. The semantic metadata contains explicit representation of geo-data semantics using
   - Top-level ontology
   - Knowledge from knowledge bases about
     - ontology mappings between the local ontologies
     - mappings from local ontology to reference ontology
     - mapping rules from local ontology to the top-level ontology
2. The geospatial metadata (catalog of features), that is, reference ontology (set of feature classes)
3. The schema mappings metadata contain basic information about mapping user requests (queries) to the local schemas

A GinisCatalog interface provides operations to retrieve semantic mappings between concepts from local ontologies, the geo-object types that it can handle, and the schema definition of a feature type. Using metadata from GinisCatalog, as a response to a user query, the semantic mediator can return sets of geo-objects (features) from different data sources.

All access to geo-information in the local community environment goes through the Web feature service (WFS) defined by the proposed OGC technology standards (*Open GIS Consortium*, 2006). We enhanced this interface with additional functionality in order to support user profiles and privileges. GIS applications (desktop and Web applications) use these interfaces to gather information about users and to provide users with information about capabilities and domain concepts they can browse for. WFS interfaces are implemented by the semantic mediator component.

The generic GeoNis architecture also allows chaining of the semantic mediators. Metadata and top-level ontology must be provided in order to integrate geo-data from several different mediators. The shared GIS server (catalog) maintains this metadata and top-level ontologies as suggested by the GeoNis architecture. In this way several GIC nodes can be combined into a component that can be treated as a GIC node and incorporated in a broader system. For example, all GIC nodes from one local community, in this way, can be incorporated with GIC nodes from other local communities in some state province, state provinces GIC nodes can be incorporated

on a country level, and country GIC nodes can be incorporated on a level of some international organization.

Semantic Ontology Mappings

In the semantic mediator, we propose a semantics-based integration approach that uses multiple ontologies instead of an integrated view (Stoimenov & Đorđević-Kajan, 2005). The ORHIDEA formal ontology consists of definitions of terms, and it includes concepts with associated attributes, relationships, and constraints defined between the concepts and entities that are instances of concepts. In our system architecture, it is assumed that the ontology is shared, and there exists commitment by the clients about data that will be shared. However, in the first phase, there is no need for commitment to common, top-level ontology. The purpose of our formal ontology is sharing, merging, and querying data, but not reading and efficient processing.

We consider ontologies with *is-a* inheritance relations and typed roles between concepts. In an (local or top-level) ontology, inheritance relationships define a partial order over concepts and carry subset semantics. The following short definition describes ontologies as it is used in our scenario. The GeoNis ontology is an abstraction of the domain of interest D, represented by the triple $O=(C, R, isa)$, where $C= \{c_i \mid i=1, n\}$ is a set of concepts, $R= \{r_i \mid i=1, n\}$ is a set of binary typed roles (or relations) between concepts, and *is-a* is a set of inheritance relationships defined between concepts. A set of semantic relations between concepts defines semantics of concepts and their relevance. We have defined the following set of relations between concepts in the ontology: $R = \{synonym, hypernym, hyponym, meronym, T\}$, where T is a set of topological relations $T = \{arc\text{-}node, route, node\text{-}route, point\text{-}event\}$. More details about the formal definition of GeoNis ontologies are given in Stoimenov and Đorđević-Kajan (2003, 2005).

Our semantic mediator uses a hybrid ontology approach. Our solution is to formally specify the meaning of the terminology of each GIC (i.e., local service or office) using local ontologies and to define translation between each GIC terminology (local ontology) and shared domain terminology (in top-level ontology). In this context, ontologies are virtually linked by interontology relationships, which are then used to indirectly support query processing. The semantic mediator provides a methodology and software support for semantic mismatches (conflicts) resolving between terminologies. This methodology uses the defined ontology mappings between each community terminologies and a top-level ontology or the common data model (reference ontology).

The hybrid ontology approach in GeoNis implies three types of semantic mappings between concepts from two local ontologies.

1. Direct semantic relationships between two ontologies
2. Indirect semantic relationships across top-level ontology
3. Semantic mappings across reference (common) models

The relationship between concepts of different information sources (between local ontologies) is the task of the semantic intercorrespondences (Stoimenov & Đorđević-Kajan, 2003). We divide the semantic conflict (semantic intercorrespondences) into four types: semantic equality (similarity), $SEqu(c_1,c_2)$; semantic dissimilarity, $SNEqu(c_1,c_2)$; semantic intersection, $SIntersec(c_1,c_2)$; and semantic contain, $SContain(c_1,c_2)$. Resolving semantic conflicts among ontologies in GeoNis is based on semantic mappings using those intercorrespondences (or semantic interrelationships).

The mapping between local ontologies and the top-level ontology can be based on generic relationships, such as "Subclass_Of" and "Same_Class_As," and on relationships that depend on the application domain. In our approach, this type of mappings can be based on the defined one-to-one semantic interrelationships SEqu, SContain, and SIntersect. In this case, SEqu is used as a "Same_Class_As" type of relationship, and SCon-

tain and SIntersect as "Subclass_Of." Also, we define *Refers_to(RefClass a, OntologyConcept c)* relationships between reference (common) model object classes and application ontology classes (concepts). The predicate *Refers_to* enables the definition of semantic relevance *SRelev(b,c)* intercorrespondence between concepts from different ontologies. With the Refers_to relationship, we can define the relationship between concepts from different application (local) ontologies without any other kind of semantic relationships.

The standard inference process defined in the GeoNis approach implies searching in an ontology tree and in semantic relationships between ontologies. The search can be performed by automatic mapping between concepts in the same ontology (within the same domain) and between two ontologies (two domains) using direct semantic relationships. This is possible by applying a standard terminological reasoner, which can work with concepts described in the description logic.

However, it might be impossible to define direct mappings between concepts from distinct local ontologies. The reason is very simple: For effective semantic mapping, experts have to define a semantic interrelationship between two pairs of local ontologies. So, it is obvious that it becomes impossible to provide explicit mappings between the concepts from local ontologies. Also, there are more complex relationships between general concepts from top-level ontology and concrete concepts from local ontology that we have to define. We have identified three more types of semantic intercorrespondences between top-level and local ontology (Stoimenov & Đorđević-Kajan, 2005). The first one is one-to-many semantic mapping, where two or more classes (description of concepts) from local ontology together are a "Subclass_Of" one general class (description of concept) from top-level ontology. The second one is many-to-one semantic mapping, where one concept from local ontology is the "Subclass_Of" two or more general concepts from top-level ontology. This type of semantic relationship corresponds to multiple inheritance. The last one is many-to-many semantic mapping, where two or more concepts from local ontology together are a "Subclass_Of" two or more general concepts from top-level ontology. This type of relationship is not useful for semantic conflict resolving, but if it exists, that means there is inconsistency in top-level ontology.

We use proposed semantic relationships between local ontologies across top-level ontology for the generation of *if-then* rules, but with user assistance and intervention (online user intervention). The inference module RuleEngine of the semantic mediator uses those rules as additional knowledge for resolving semantic conflicts. Such mapping is required when one system sends a query to another GIC every single time. Because of that, this approach is very expensive and it demands involvement of domain experts.

When one system requests data from another, we need more accuracy from the semantic mapping process to minimize the errors and misspellings. The more GICs get involved, more local ontologies are created. Thus, the mapping of ontologies becomes a core issue. The problem of missing semantic intercorrespondences can lead to interpretation conflicts during interchanging of geo-data. These conflicts can often be resolved by human intervention, as in the proposed GeoNis (online) approach of the generation of *if-then* rules. However, there are lots of problems in the knowledge acquisition process, the process of local ontology construction, and in defining semantic mapping rules. In such processes, we need the additional involvement and commitment of experts from the domains of interest. Nevertheless, the efficiency of that process is not satisfactory enough. The correctness of semantic mapping, if there does not exist direct semantic mapping, is about 30%. As a consequence, automatic or at least semiautomatic techniques have to be developed to reduce the burden of the manual creation and maintenance of mappings. This process of interpretation conflict solving can be automated only if there exits semantic mapping for any pair of ontologies.

Because of that, we try to solve this problem by introducing semantic mappings across global

ontologies (top-level and common model ontologies), and by generating the rules (based on semantic similarity), which are used to expand knowledge about mappings. We try to develop an off-line process of discovering semantic relationships between concepts from different ontologies. This is an on-request (off-line) search of indirect semantic mapping and generation of new rules. The main task of the off-line discovering process is to recognize possible semantic similarity between concepts from two ontologies.

In the proposed approach, we do not use concept attributes to check similarity. Our approach implies the use of semantic relationships for defining probability for the similarity of two concepts. From our point of view, we want to develop a process of the semantic discovering of existing semantic relationships and compare those relationships between two concepts to find similarity among them. We use a definition of similarity as the predicate $Sim(c_1,c_2)$, meaning "probability of similarity between concepts c_1 and c_2," where c_1 and c_2 are the concepts from two different local ontologies. This predicate has values from 0 (with meaning false, or two concepts are dissimilar) to 1 (with meaning true, or two concepts are semantically similar). The predicate $Sim(c_1, c_2) = 1$ means two concepts c_1 and c_2 are similar if there exists relationship r between c_1 and c_2, $r(c_1, c_2)$, and $r \in R$ (see definition of GeoNis ontology $O=(C, R, isa)$). Predicate $Sim(c_1, c_2) = 0$ means this is the start value in the process of discovering mappings; when two objects are different, they do not have any kind of interrelationship. Other values between 0 and 1 ($0<$ and <1) are the probability for the semantic similarity of two concepts. Those values are calculated in the process of discovering mappings.

FUTURE TRENDS

The new generation of information systems including GIS should be able to solve semantic heterogeneity (Fonseca, 2001). The need to share geographic information is well documented (Stoimenov et al., 2000; Vckovsky, 1998). Making local geographic data sets publicly available and establishing a common interoperability framework over shared data interchange protocols are important parts of this research.

Although distributed geo-libraries offer numerous advantages over stand-alone geographic databases, there are institutional and technical problems of geo-data sharing and interoperability. These problems have become, over the last several years, the focus of international research and infrastructure efforts and have been discussed at several international conferences and workshops focused on GIS interoperability (Association of GIS Laboratories in Europe [AGILE], 2004).

One solution for data exchange between different GIS data sources would be a single architecture and set of standards for geospatial data. A broader discussion of geographic information exchange formats can be found in *GDE Geographic Data Exchange Standards* (2002). One of the important strategies for interoperability is the conversion of different data formats into common data structure. This kind of data structure is usually based on one of the existing GIS standards. However, it is all but impossible to conceive that the global GIS community would adopt a single geospatial architecture or data standard worldwide. This means that standardization efforts alone will not produce interoperability.

Today, semantics is a key factor of successful interoperability between information systems. During the past several years, many different solutions for problems of semantic heterogeneity have been proposed (Bernard et al., 2003; Klien, Einspanier, Lutz, & Hübner, 2004; Stoimenov & Đorđević-Kajan, 2005). One way of making the meaning of information more explicit is the use of ontologies (Spaccapietra, Cullot, Parent, & Vangenot, 2004), also in the geographic field. Some efforts important for future development trends in creating geographic ontologies are found in Apinar, Sheth, Ramakrishnan, Usery, Azami, and Kwan (2005). Ontologies are generally recognized as an essential tool for allowing communication and knowledge sharing among distributed users

and applications, by providing a semantically rich description and a common understanding of a domain of interest (Castano et al., 2005; Stoimenov & Đorđević-Kajan, 2005). Fonseca et al. (2002) also address the semantic aspects of geographic information integration. In this context, these semantics aspects are related to the meaning of the entities that compose the ontologies. These ontologies represent concepts of the real world or, more specifically, of the geographic world. Their concern is with semantic granularity rather than spatial granularity. Semantic granularity addresses the different levels of spatial resolution or representation at different scales. The support and use of multiple ontologies should be a basic feature of the modern information systems.

CONCLUSION

A data integration system provides a uniform interface to a multitude of data sources. Such middleware systems, usually based on mediator and wrapper technologies, separate the storage of data management facilities, providing common interfaces for application.

Interoperability in general, and especially semantic interoperability, will lead to dramatic organizational changes in the GI community. The integration of diverse information sources has many advantages.

- Quality improvement of data due to the availability of complete data sets
- Improvement of existing analysis and application of the new analysis
- Cost reduction resulting from the multiple use of existing information sources
- Avoidance of redundant data and conflicts that can arise from redundancy

The GeoNis is a generalized framework in which both schematic and syntactic heterogeneity is resolved by mediation and a common data model. This framework is aimed to resolve interoperability problems in local and municipality environments. Our solution uses a mediator-based architecture for interoperability in the local community environment. OpenGIS Simple Feature is the common model, and local ontologies are for resolving the semantic heterogeneity of data sources. The principles behind the ontology and mediation framework described in this article are extensibility, relative autonomy of infrastructure nodes, and universal access to heterogeneous data sources from a variety of portals.

GeoNis uses the widely accepted OpenGIS standard for geo-data modeling and representation on the mediator level. Changes in the OpenGIS standard have impact only on translators, not on information sources. GeoNis also enables adding legacy data sources in the interoperability process. The only condition is the translator realization (implementation).

Now, the GeoNis architecture provides a solution to the problem of semantic heterogeneity for a specific domain, for example, applications in local city services, which deal with network data structures such as telecom, water and soil pipe services, power supply services, and some local government services (Stoimenov & Đorđević-Kajan, 2003). In this environment, we can treat the semantic mediator as an access point for discovering and retrieving semantically heterogeneous data in that domain. If there are metadata and top-level ontology (shared vocabulary) for a domain provided to a semantic mediator, it can provide semantic consistency for data from that domain and geo-data exchange with terminology translation as described in the GeoNis architecture (Stoimenov & Đorđević-Kajan, 2005). The use of ontologies was proposed as a knowledge base to solve semantic conflicts such as homonyms, synonyms, and taxonomic heterogeneities.

Future research in the GeoNis project refers to the following.

- Generating domain (local) ontologies
- Developing the domain-oriented translators
- Developing tools and methodologies for semiautomatic translator generating

- Development of methodology for semiautomatic recognition of semantic intercorrespondences between domain ontologies

The significance of our work is based on the usefulness of the GeoNis tools and components for realization of interoperable geo-spatial and other (such as B2; see Kajan & Stoimenov, 2005) information nodes in local community organizations.

REFERENCES

Apinar, I. B., Sheth, A., Ramakrishnan, C., Usery, E. L., Azami, M., & Kwan, M.-P. (2005). Geospatial ontology development and semantic analysis. In J. P. Wilson & S. Fotheringham (Eds.), *Handbook of geographic information science.* Blackwell Publishing.

Association of GIS Laboratories in Europe (AGILE). (2004). Tutorial & workshop interoperability for geoinformation. *7th AGILE Conference on Geographic Information Science.* Retrieved from http://agile2004.iacm.forth.gr/

Bernard, L., Einspanier, U., Haubrock, S., Hübner, S., Kuhn, W., Lessing, R., et al. (2003). Ontologies for intelligent search and semantic translation in spatial data infrastructures. *Photogram-metrie, Fernerkundung, Geoinformation 2003* (pp. 451-462).

Bishr, Y. A. (1998). Overcoming the semantic and other barriers to GIS interoperability. *International Journal of Geographic Information Science, 12*(4), 299-314.

Bishr, Y. A., Pundt, H., Kuhn, W., & Radwan, M. (1999). Probing the concept of information communities: A first step toward semantic interoperability. In M. Goodchild, M. Egenhofer, R. Fegeas, & C. Kottman (Eds.), *Interoperating geographic information systems* (pp. 55-69). Kluwer Academic.

Boucelma, O., & Colonna, F. M. (2005). Mediation for online geoservices. In *Lecture notes in computer science: Vol. 3428. Web and wireless geographical information systems, interoperability and security in W2GIS* (pp. 81-93). Berlin, Germany: Springer.

Castano, S., Ferrara, A., & Montanelli, S. (2005, February). *Ontology-based interoperability services for semantic collaboration in open networked systems.* Proceedings of the 1st International Conference on Interoperability of Enterprise Software and Applications (INTEROP-ESA 2005), Geneva, Switzerland.

Chawathe, S., Garcia-Molina, H., Hammer, J., Ireland, K., Papakonstantinou, Y., Ulman, J., et al. (1994). The TSIMMIS project: Integration of heterogeneous information systems. *Proceedings of ISPJ Conference* (pp. 7-18).

Cruz, I. F., Sunna, W., & Chaudhry, A. (2004). Semi-automatic ontology alignment for geospatial data integration. In M. J. Egenhofer, C. Freksa, & H. J. Miller (Eds.), *Lecture notes in computer science* (Vol. 3234, pp. 51-66). GIScience.

Devogele, T., Parent, C., & Spaccapietra, S. (1998). On spatial database integration. *International Journal of Geographic Information Science, 12*(4), 335-352.

Fonseca, F. (2001). *Ontology-driven geographic information systems.* Unpublished doctoral dissertation, University of Maine, ME.

Fonseca, F., Davis, C., & Camara, C. (2003). Bridging ontologies and conceptual schemas in geographic information integration. *GeoInformatica, 7*(4), 307-321.

Fonseca, F., & Egenhofer, M. (1999). Ontology-driven information systems. *7th ACM Symposium on Advances in GIS* (pp. 14-19).

Fonseca, F., Egenhofer, M., Davis, C., & Câmara, G. (2002). Semantic granularity in ontology-driven geographic information systems. *Annals of Mathematics and Artificial Intelligence, 36*(1-2), 131-151.

GDE geographic data exchange standards. (2002). Retrieved October 2006 from http://www.iecapc.jp/06/diffusenew/standards/gis.html

Gruber, T. (2002). *What is an ontology?* Retrieved 2006 from http://www.ksl.stanford.edu/kst/what-is-an-ontology.html

Guarino, N. (1998). Formal ontology in information systems. *Formal Ontology in Information Systems: Proceedings of FOIS'98* (pp. 3-15).

Hakimpour, F., & Timpf, S. (2001, April). *Using ontologies for resolution of semantic heterogeneity in GIS*. 4th AGILE Conference on Geographic Information Science, Brno, Czech Republic.

Hess, G. N., Iochpe, C., & Oliveira, J. P. M. (2004). Analysis patterns for the geographic database conceptual schema: An ontology aided approach. *Proceedings of the International Conference on Semantics of a Networked World (ICSNW)* (pp. 323-324).

Institute of Electrical and Electronics Engineers (IEEE). (1990). *IEEE standard dictionary: A compilation of IEEE standard computer glossaries*. New York: Author.

Janowicz, K. (2005). Extending semantic similarity measurement by thematic roles. *Proceedings of the GeoS'05: First International Conference on GeoSpatial Semantic* (pp. 137-152).

Kajan, E., & Stoimenov, L. (2005). Towards ontology-driven architectural framework for B2B. *Communications of the ACM, 48*(12), 60-66.

Klien, E., Einspanier, U., Lutz, M., & Hübner, S. (2004). *An architecture for ontology-based discovery and retrieval of geographic information*. Proceedings of the 7th Agile Conference on Geographic Information Science, Heraklion, Crete.

Larson, A. P. (1998). Federated databases systems for managing distributed, heterogeneous, and autonomous databases. *ACM Computing Surveys, 22*, 183-236.

Laurini, R., et al. (1998). Spatial multi-database topological continuity and indexing: A step towards seamless GIS data interoperability. *International Journal of Geographic Information Science, 12*(4), 373-402.

Levinsohn, A. (2000). Geospatial interoperability: The holy grail of GIS. *GeoEurope*. Retrieved 2006 from http://www.geoplace.com/gw/2000/1000/1000data.asp

Matousek, K., Mordacik, J., & Janku, L. (2001). On implementing the data warehouse: GIS integration. *Proceedings of the World Multiconference on Systemics, Cybernetics and Informatics: Information Systems Development, 1*, 206-210.

Obrst, L. (2003). Ontologies for semantically interoperable systems. *CIKM'03* (pp. 366-369).

Open GIS Consortium. (2006). Retrieved October 2006 from http://www.opengis.net/

Park, J., & Ram, S. (2004). Information systems interoperability: What lies beneath? *ACM Transactions on Information Systems, 22*(4), 595-632.

Rodrıguez, M. A., & Egenhofer, M. J. (2003). Determining semantic similarity among entity classes from different ontologies. *IEEE Trans. Knowl. Data Eng., 15*(2), 442-456.

Schwering, A., & Raubal, M. (2005). Spatial relations for semantic similarity measurement. In J. Akoka, S. W. Liddle, I.-Y. Song, M. Bertolotto, I. Comyn-Wattiau, S. S.-S. Cherfi, W.-J. van den Heuvel, B. Thalheim, M. Kolp, P. Bresciani, J. Trujillo, C. Kop, & H. C. Mayr (Eds.), *Lecture notes in computer science* (Vol. 3770, pp. 259-269). Springer.

Sondheim, M., et al. (1999). GIS interoperability. In P. Logley, M. Goodchild, D. Maguire, & D. Rhind (Eds.), *Geographical information systems principles and technical issues*. New York: John Wiley & Sons.

Sotnykova, A., Vangenot, C., Cullot, N., Bennacer, N., & Aufaure, M. (2005). Semantic mappings in description logics for spatio-temporal database schemas integration. *Journal of Data Semantic, 3*, pp. 143-167.

Spaccapietra, S., Cullot, N., Parent, C., & Vangenot, C. (2004). *On spatial ontologies.* Proceedings of the VI Brazilian Symposium on Geoinformatica (GEOINFO 2004), Campos do Jordão, Brazil.

Stoimenov, L., & Đorđević-Kajan, S. (2002). Framework for semantic GIS interoperability. *FACTA Universitatis, Series Mathematics and Informatics, 17,* 107-125.

Stoimenov L., & Đorđević-Kajan, S. (2003). Realization of GIS semantic interoperability in local community environment. *6th AGILE Conference on Geographic Information Science: The Science Behind the Infrastructure* (pp. 73-80).

Stoimenov, L., & Đorđević-Kajan, S. (2005). An architecture for interoperable GIS use in a local community environment. *Computers & Geoscience, 31*(2), 211-220.

Stoimenov, L., Đorđević-Kajan, S., & Stojanovic, D. (2000). Integration of GIS data sources over the Internet using mediator and wrapper technology. *Proceedings of MELECON 2000, 10th Mediterranean Electrotechnical Conference, 1,* 334-336.

Stoimenov, L., Mitrovic, A, Đorđević-Kajan, S., & Mitrovic, D. (1999). Bridging objects and relations: A mediator for an OO front-end to RDBMSs. *Information and Software Technology, 41*(2), 59-68.

Stoimenov, L., Stanimirović, A., & Đorđević-Kajan, S. (2004). Realization of component-based GIS application framework. *7th AGILE Conference on Geographic Information Science* (pp. 113-120).

Stuckenschmidt, H., Wache, H., Vogele, T., & Vissar, H. (2000). Enabling technologies for interoperability. *Workshop on the 14th International Symposium of Computer Science for Environmental Protection* (pp. 35-46).

Vckovsky, A. (1998). *International Journal of Geographic Information Science, 12*(4).

Visser, U., & Stuckenschmidt, H. (2002). Interoperability in GIS: Enabling technologies. *5th AGILE Conference on Geographic Information Science* (pp. 291-297).

Voisard, A., & Juergens, M. (1998). Geographic information extraction: Querying or quarrying? In M. Goodchild, M. Egenhofer, R. Fegeas, & C. Kottman (Eds.), *Interoperating geographic information systems.* New York: Kluwer Academic Publishers.

Wache, H., Vögele, T., Visser, U., Stuckenschmidt, H., Schuster, G., Neumann, H., et al. (2001). Ontology-based integration of information: A survey of existing approaches. *IJCAI-01 Workshop: Ontologies and Information Sharing* (pp. 108-117).

Wiederhold, G. (1998). Weaving data into information. *Database Programming & Design, 11*(9), 22-29.

KEY TERMS

Data Integration: The problem of combining data residing at different sources and providing the user with a unified view of these data.

Geographic Information System (GIS): A computerized system for managing data about spatially referenced objects. GISs differ from other types of information systems in that they manage huge quantities of data, require complex concepts to describe the geometry of objects, and specify complex topological relationships between them.

Interoperability: The ability of two or more systems or components to exchange information and to use the information that has been exchanged.

Mediator: A software component of a knowledge-based system that is used to isolate problem solvers from the logistical issues associated with accessing the database. Mediator is as an intermediate abstraction layer between databases and applications that use them.

Ontological Commitments: Agreements to use the vocabulary in a consistent way for knowledge sharing.

Ontology: An ontology is a specification of a conceptualization. In both computer science and information science, an ontology is a data model that represents a domain and is used to reason about the objects in that domain and the relations between them.

Semantic Heterogeneity: Semantic heterogeneity considers the content of an information item and its meaning. Semantic conflicts among information systems occur whenever information systems do not use the same interpretation of the information.

Semantic Interoperability: The ability of two or more computer systems to exchange information and have the meaning of that information accurately and automatically interpreted by the receiving system.

Wrapping/Wrapper: Wrapping a system is the process of defining and restricting access to a system through an abstract interface. A wrapper is a program that is specific to every data source. Wrapper extracts a set of tuples from the source file and performs translation in the data format. A wrapper for information sources accepts queries in a given format, converts them into one or more commands or subqueries understandable by the underlying information source, and transforms the native results into a format understood by the application.

Chapter LIV
Ontologies Application to Knowledge Discovery Process in Databases

Héctor Oscar Nigro
Universidad Nacional del Centro de la Provincia de Buenos Aires, Argentina

Sandra Elizabeth González Císaro
Universidad Nacional del Centro de la Provincia de Buenos Aires, Argentina

INTRODUCTION

Nowadays one of the most important and challenging problems in Knowledge Discovery Process in Databases (KDD) or Data Mining is the definition of the prior knowledge; this can be originated either from the process or the domain. This contextual information may help select the appropriate information, features or techniques, decrease the space of hypothesis, represent the output in a more comprehensible way and improve the whole process.

Most part of this background knowledge is only present as implicit knowledge –in the analyst mind- or textual documentation. Therefore we need a conceptual model to help represent this knowledge. Advances in the field of Knowledge Engineering allow codifying previous knowledge under the ontological formalism. According to Gruber's (2002) ontology definition - explicit formal specifications of the terms in the domain and relations among them - we can represent the knowledge of the Knowledge Discovery process and knowledge about domain. Ontologies are used for communication (between machines and/or humans), automated reasoning, representation and reuse of knowledge. As a result, ontological foundation is a precondition for efficient automated usage of Knowledge Discovery information. As a result, we can perceive the relation between Ontologies and Data Mining in two ways:

- **From ontologies to data mining**, we are incorporating knowledge in the process through the use of ontologies, i.e. how the experts comprehend and carry out the analysis tasks. Representative applications are intelligent assistants for the discover

process (2005) interpretation and validation of mined knowledge, Ontologies for resource and service description and Knowledge Grids (Cannataro et al., 2007).

- **From data mining to ontologies**, we include domain knowledge in the input information or use the ontologies to represent the results. Therefore the analysis is done over these ontologies. The most distinctive applications are in Medicine, Biology and Spatial Data, such as Gene representation, Taxonomies, applications in Geosciences, medical applications and specially in evolving domains (Bogorny el al., 2006; Sidhu et al., 2006)

So far, the proposals or solutions that we find in KDD with ontologies are partial, i.e. they are centered on some of the steps of knowledge discovery. For instance Euler and Scholz (2004) present a metamodel of KDD preprocessing chains that contains an ontology describing conceptual domain knowledge. This metamodel is operational, yet abstract enough to allow the reuse of successful KDD applications in similar domains. Bernstein et al. (2005) propose an intelligent tool (IDA) based on a mining ontology. Brisson and Collar (2007) present the so-called KEOPS approach integrating expert knowledge all along the data mining process in a coherent and uniform manner. An ontology driven information system plays a central role in the approach.

The main goal of this paper is to present the issue of the ontologies application in KDD. As a result of our research, we will propose a general ontology-based model, which includes all discovery steps.

This paper is presented as follows: First, Background: main works in the field are introduced. Second, Main focus section is divided into: KDD Using Ontologies cycle in which we explain the knowledge process and propose a model, Domain Ontologies, Metadata Ontologies and Ontologies for Data Mining Process. Third: Future Trends, Conclusions, References and Key Terms.

BACKGROUND

This section describes the most recent research works in ontologies application to KDD. As you will appreciate, none of the research works alludes to the use of the ontologies in the whole process.

Singh et al. (2003) have developed a context aware data mining framework which provides accuracy and efficiency to data mining outcomes. Context factors were modeled using Ontological representation. Although the context aware framework proposed is generic in nature and can be applied to most of the fields. Hotho et al. (2003) have showed that using ontologies as filters in term selection prior to the application of a K-means clustering algorithm will increase the tightness and relative isolation of document clusters as a measure of improvement.

Pan and Shen (2005) have proposed architecture for knowledge discovery in evolving environments. The architecture creates a communication mechanism to incorporate known knowledge into the discovery process through ontology service facility.

Rennolls (2005) have developed an intelligent framework for Data Mining, Knowledge Discovery and Business Intelligence. The ontological framework will guide the user to the models from an expanded data mining toolkit, and the epistemological framework will assist the user to interpret and appraise the discovered relationships and patterns.

Li and Zhong (2006) have introduced an approach to automatically discover ontologies from data sets in order to build complete concept models for Web user information needs. They proposed a method to capture evolving patterns to refine discovered ontologies and established a process to assess relevance of patterns in an ontology by the dimensions of exhaustively and specificity.

Brezany et al. (2005) have addressed the issues of composing workflows with automated support developed on top of Semantic Web technolo-

gies and the workflow management framework elaborated in Grid data mining project. The kernel part of that support is a tool called the GridMiner Workflow Assistant, which helps the user interactively construct workflow descriptions expressed in a standard workflow specification language. The specification is then passed to the workflow engine for execution. GMA operations are controlled by the Data Mining Ontology.

MAIN FOCUS

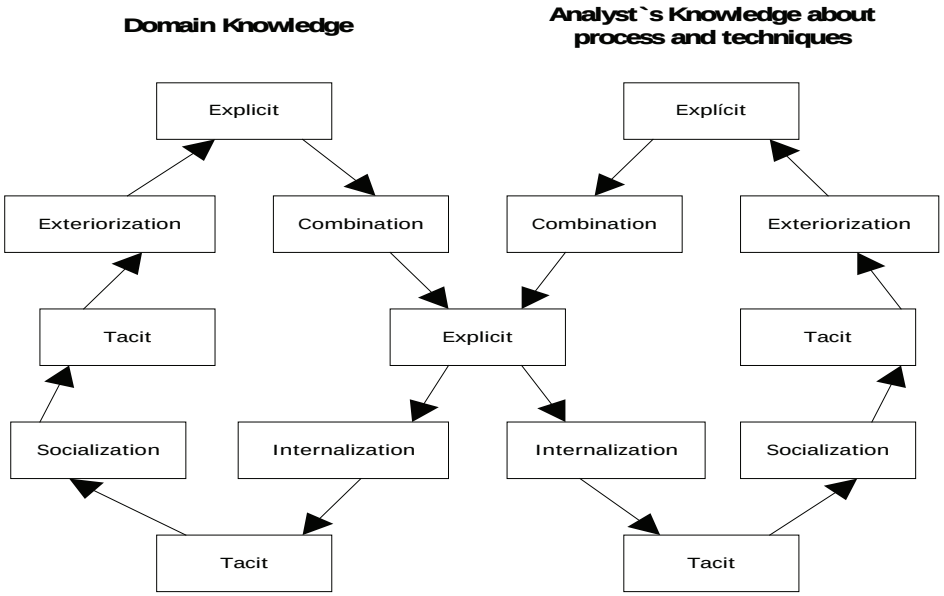
KDD Using Ontologies Cycle

Interactive nature is inherent to a data mining process since communications with experts is necessary for understanding the domain and for interpreting results Brisson and Collar (2007). Therefore there is a strong relation between the knowledge discovery process and the knowledge application domain. The cognitive states of the analyst change according to what it is being done during the analysis of information. The profile of the cognitive states depends on the level of mastery of the analyst (Young et al., 1996), as well.

Based on the process of knowledge by Nonaka and Takeuchi (1995)-this is a model of generation of knowledge by means of a spiral-, we can perceive the knowledge discovery in databases as a double cycle process with epistemological and ontological content (see Figure 1). In the right side of Figure 1, we consider data miner knowledge as regards the techniques and the whole process, i.e. according to his idea of "what to do?" or "what is needed?" or "which is his/her work hypothesis?" He knows what technique or techniques to choose, the variables and the type of desired output model. The analyst's experience is part of tacit knowledge and does have an important value. This analyst knowledge may be called *mental model*. In the left side of Figure 1, knowledge about domain is present and can help to select both the variables and the techniques. If a new knowledge appears in the process, the domain is updated.

Our vision of the KDD with ontologies is more oriented to Computational Discovery of Scientific Knowledge, developed by Langley (2000) than to the traditional one by Fayyad (1996), since:

Figure 1. Double cycle of knowledge in knowledge discovery process in databases

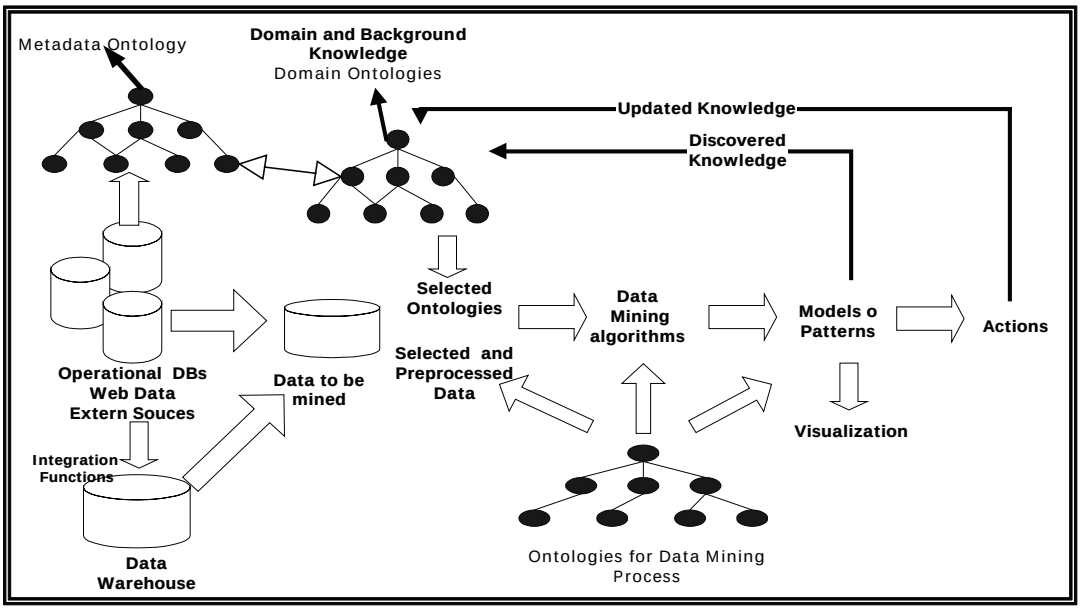


- Knowledge can also assist in the search for useful features (i.e. placing constraints on acceptable combinations of features, providing an initial set of features from which to start search, biasing selection to produce understandable models)
- This approach typically produces models that are more accurate and easier to comprehend than the ones induced from scratch
- Using knowledge to influence discovery can reduce prediction error but also improve model comprehension.
- This is an intensive engineering knowledge process, with human intervention in interpretation and validation.

Most KDD methods mainly focus on discovering knowledge from scratch, and thus offer no way to incorporate scientists' existing knowledge about a domain, (Bridewell et al., 2006) techniques and previous results. As we have exposed in the introductory section, ontologies give a formal language to represent this knowledge. However, none of the models described has an integral vision of whole KDD with Ontologies. So we propose a conceptual model using Ontologies, illustrated in figure 2. This model involves all the steps of the process - it also considers either the model of Fayyad (1996) or that of Crisp-DM - which is based on three types of ontologies:

- ***Metadata Ontologies*:** these ontologies establish how variables are constructed, i.e. which were the processes that allow us to obtain their values, and they can vary using any other method. These ontologies must also express general information about the variable as it is treated.
- ***Domain Ontologies*:** these ontologies express the knowledge about application domain.
- ***Ontologies for Data Mining Process*:** these ontologies codify all knowledge about the process, i.e. select features, select the best algorithms according to the variables and the problem, and establish valid process sequences.

Figure 2. General model of KDD with ontologies



The proposed model allows the interaction between the analyst ideas such as work hypothesis, type of desired output model and previous knowledge. Since the analyst can visualize the domain and metadata ontologies, he/she can then learn about their relationship and properties.

With this knowledge in mind he/she selects the technique/s. Ontologies for Data Mining Process help the user in the choice of the most suitable variables, data instances and algorithm to develop his/her model according to the parameters -chosen technique, the variable type, precision, accuracy, cost sensitivity, comprehensibility, data matrix, properties of Domain and metadata ontologies, among others. Once the algorithm has been selected, preprocessing steps are made on input data -or Ontologies domain- with the help of Ontologies for Data Mining Process. Then the algorithm is applied to preprocessed data set.

Furthermore Ontologies for Data Mining Process give the most appropriate post-processing steps and visualizations for the model; the output model is evaluated and visualized. The analyst may decide the change of Domain Ontologies properties if new knowledge turns up in the output model.

Unquestionably, all the process is interactive, i.e. at any point of discovery process the analyst can change parameters or re-initialize the discovery process.

Domain Ontologies

The models in many scientists work to represent their work hypotheses are generally cause-effect diagrams. Models make use of general laws or theories to predict or explain behavior in specific situations. Currently these cause-effect diagrams can be easily translated to ontologies by means of conceptual maps, which discriminate taxonomy organized as central concepts, main concept, secondary concepts, and specific concepts.

Discovery systems produce models that are valuable for prediction, but they should also produce models that have been stated in some declarative format, that can be communicated clearly and precisely, which help people understand observations, in terms which are well-known (Langley, 2000). Models can be from different appearances and dissimilar abstraction level, but the more complex the fact for which they account; the more important they will be cast in some formal notation with an unambiguous interpretation. This new knowledge can be easily communicated and updated between systems and Knowledge databases. Particularly in the Data Mining field, the knowledge can be represented in different formalisms, such as rules, decision trees and clusters, which are known as models. Discovery systems should generate knowledge in a format that is well known to domain users.

There is an important relation between knowledge structures and discovery process with learning machine. The former ones are important outputs of the discovery process, and are important inputs for discovery (Langley, 2000). Thus knowledge plays as crucial a role as data in the automation of discovery. Therefore, ontologies provide a structure capable of supporting knowledge representation about domain. Examples of this type of ontology are: geo-ontologies (Bogorny et al., 2006) and protein ontologies (Sidhu et al., 2006).

Metadata Ontologies

As Spyns et al. (2002) state ontologies in current computer science language are computer-based resources that represent agreed domain semantics. Unlike data models, the fundamental asset of ontologies is their relative independence from particular applications, i.e. an ontology consists of relatively generic knowledge that can be reused by different kinds of applications/tasks.

In contrast, a data model represents the structure and integrity of the data elements of the, in principle "single", specific enterprise application(s) to be used. Consequently, the con-

ceptualization and the vocabulary of a data model are not intended to be shared by other applications a priori (Gottgtroy et al., 2005).

Similarly, in data modeling practice, the semantics of data models often constitute an informal agreement between the developers and the users of the data model –included when a Data Warehouse is designed- and, in many cases, the data model is updated as it evolves when particular new functional requirements arise without any significant update in the metadata repository. Both ontology model and data model have similarities in terms of scope and task. They are context dependent knowledge representation, that is, it is hard to draw a line between generic and specific knowledge when you are building ontology. Moreover, both modeling techniques are knowledge acquisition intensive tasks and the resulted models represent a partial account of conceptualizations (Gottgtroy et al., 2005).

In spite of the differences, we should consider the similarities and the fact that data models carry a lot of useful hidden knowledge about the domain in its data schemas in order to build ontologies from data and improve the KDD. Since data schemas do not have the required semantic knowledge to intelligently guide ontology construction has been presented as a challenge for Database and Ontology engineers (Gottgtroy et al., 2005).

Ontologies for Data Mining Process

Vision about KDD process is changing over time. In its beginnings, the main objective was to extract a valuable pattern from a fat file as a game of try and error. As time goes by, researchers and fundamentally practitioners discuss the importance of a priori knowledge. Also, the knowledge and understandability about the problem, the choice of the methodology to make the discovery, the expertise in similar situations and an important question arises: up to what extent is such inversion on Data Mining Projects worthwhile?

As practitioners and researchers in this field, we can perceive that expertise is very important, knowledge about domain is helpful and it simplifies the process. To develop the process in a more attractive way for managers, practitioners must do it more efficiently and reusing experience. Hence we can codify all statistical and machine learning knowledge with ontologies and use them. Examples of this type of ontology is exposed in the works of Bernstein et al. (2005), Zhou et al. (2006) and Cannataro et al. (2007)

FUTURE TRENDS

The next step is the formal specification of our model in terms of design. Also, the implementation of the three types the ontologies will be done on ontology languages, such as RDF or RDFS. Once our model has been fully implemented in different domains; opinion test for different types of user will be done.

Potential progress in the algorithms that work on Ontologies and Data Mining will be guided by the techniques to be explored and developed.

CONCLUSION

As we have explained, vision about KDD is changing over time. In its beginnings, the main objective was to extract a valuable pattern from a fat file as a game of try and error. As time goes by, researchers and mainly practitioners discuss the importance of a priori knowledge, the knowledge and understandability about the problem, the choice of the methodology to do the discovery, the expertise in similar situations. In this work, we have explored the most significant and advanced works in KDD with ontologies.

We have included ontologies in KDD since they are useful in order to formalize domain concepts, to express knowledge about the process, to generate models and to allow knowledge and

model comparisons using ontology relationships. We propose a conceptual model which involves all steps of the discovery based on three types of ontologies: *Metadata, Domain* and *for Data Mining Process. Metadata* Ontology helps to establish how the variables are constructed. *Domain* Ontology helps to represent the knowledge about the domain; and Ontologies *for Data Mining Process* codify all knowledge about the process itself.

We do believe this is only the beginning of the subject under discussion; much work must be done. However, this is an interesting and very promising direction, since the advantages on KDD with ontologies are: domain knowledge reuse, more compressive models and common language to communicate between applications. In particular, we focus on how ontologies can help construct models that are obtained by means of the intensive application of the knowledge process but not by a process of try and error.

An effective Knowledge Management is present in our model, as the knowledge is in a permanent transformation cycle allowing the reuse of the old knowledge and its update for the one discovered in the process.

REFERENCES

Bernstein, A., Provost, F., & Hill, S. (2005). Towards Intelligent Assistance for the Data Mining Process:An Ontology-based Approach for Cost/Sensitive Classification. *In IEEE Transactions on Knowledge and Data Engineering, 17*(4), 503-518.

Bogorny, V. et al. (2006). Towards elimination of well known geographic domain patterns in spatial association rule mining. *In Third IEEE International Conference on Intelligent Systems*, (pp. 532-537).

Brezany, P. et al. (2004). GridMiner: A Framework for Knowledge Discovery on the Grid - from a Vision to Design and Implementation. *Cracow Grid Workshop*, Cracow, Poland: Springer.

Bridewell, W. et al. (2006). An Interactive environment for the Modeling on discovery of scientific knowledge. *International Journal of Human-Computer Studies, 64,* 1009-1014.

Brisson, L., & Collard, M. (2007). *An Ontology Driven Data Mining Process.* Research report of University of Nice, France. Retrieved from http://www.i3s.unice.fr/~mcollard/KEOPS.pdf on July 2007.

Cannataro, M. et al. (2007). Using ontologies for preprocessing and mining spectra data on the Grid. *Future Generation Computer Systems, 23*(1), 55-60.

Euler, T., & Scholz, M. (2004). Using ontologies in a kdd workbench. In P. Buitelaar, et al. (Eds.), *Workshop on Knowledge Discovery and Ontologies at ECML/PKDD* '04, (pp. 103-108).

Fayyad, U. et al. (1996). *Advances in Knowledge Discovery and Data Mining.* Merlo Park, CA: AAAI Press.

Gottgtroy P. et al. (2005). Enhancing data analysis with Ontologies and Olap. *Proceedings Data Mining 2005 Conference*, Skialhos, Grecia.

Gruber, T. (2002). *What is an Ontology*? Retrieved from http://www-ksl.stanford.edu/kst/what-is-an-ontology.html on November 2007.

Hotho, A. et al. (2003). Ontologies Improve Text Document Clustering. *In Proceedings of the 3rd IEEE Conference on Data Mining* (pp. 541-544), Melbourne, FL, USA.

Langley, P. (2000). The computational support of scientific discovery. *International Journal of Human-Computer Studies, 53,* 393-410.

Li, Y., & Zhong, N. (2006). Mining Ontology for Automatically Acquiring Web User Information Needs. *IEEE Transactions on Knowledge and Data Engineering, 18*(4), 554-568.

Nonaka, I., & Takeuchi, H. (1995). *The Knowledge-Creating Company.* New York: Oxford University Press, Inc.

Pan, D., & Shen, J. Y. (2005). Ontology service-based architecture for continuous knowledge discovery. In *Proceedings of International Conference on Machine Learning and Cybernetics, 4,* 2155-2160.

Rennolls, K. (2005). An Intelligent Framework (O-SS-E) For Data Mining, Knowledge Discovery and Business Intelligence. *In Proceeding 2nd International Workshop on Philosophies and Methodologies for Knowledge Discovery, PMKD'05,* (pp. 715-719).

Sidhu, A., Dillon, T., & Chang, E. (2006). Advances in Protein Ontology Project. *In Proceedings of 19th IEEE International Symposium on Computer-Based Medical Systems (CBMS 2006).*

Singh, S. et al. (2003). Context-Based Data Mining Using Ontologies. In I. Song et al. (Eds.), *Proceedings 22nd International Conference on Conceptual Modeling,* Lecture Notes in Computer Science, *2813,* 405-418. Springer.

Spyns, P., Meersman, R., & Jarrar, M. (2002). Data modeling versus ontology engineering. *SIGMOD Record Special Issue on Semantic Web, Database Management and Information Systems, 31.*

Young, F., & Thurstone, L. (1996). *A Cognitive Model of Data Analysis and its Implementation as a Human-Computer Interface*. Psychometric Laboratory, University of North Carolina, Chapel Hill, North Carolina.

Zhou, X., Geller, J., Perl, Y., & Halper, M. (2006). An application intersection marketing ontology. Theoretical computer science: Essays in memory of Shimon Even. *Lecture Notes in Computer Science, 3895,* 143-153. Berlin: Springer-Verlag.

KEY TERMS

IDA -Intelligent Discovery Assistant: Helps data miners with the exploration of the space of valid DM processes. It takes advantage of an explicit ontology of data-mining techniques, which defines the various techniques and their properties. (Bernstein et al, 2005).

Knowledge Discovery Process in Databases: "The non-trivial process of identifying valid, novel, potentially useful, and ultimately understandable patterns in data" (Fayyad et al., 1996).

Knowledge Engineering: A field within artificial intelligence that develops knowledge-based systems. Such systems are computer programs that contain large amounts of knowledge, rules and reasoning mechanisms to provide solutions to real-world problems.

Knowledge Grid: A software architecture for geographically distributed PDKD (Parallel and Distributed Knowledge Discovery) applications called Knowledge Grid, which is designed on top of computational Grid mechanisms provided by Grid environments. The Knowledge Grid uses basic Grid services and they are organized into two layers: *Core K-Grid Layer,* which is built on top of generic Grid services, and *High-Level K-Grid Layer,* which is implemented over the core layer.

Knowledge Management: An integrated, systematic approach to identifying, codifying, transferring, managing, and sharing all knowledge of an organization.

Ontology: "A specification of a conceptualization …a description of the concepts and relations that can exist for an agent or a community of agents" (Gruber 2002).

RDF: Resource Description Framework is a formal language to define ontologies. Defines a data model as a series of resources and relations among them

RDFS: Resource Description Framework Schema- It is a vocabulary for describing properties and classes of RDF resources.

Section VI
Data Mining

Chapter LV
Expression and Processing of Inductive Queries

Edgard Benítez-Guerrero
Laboratorio Nacional de Informática Avanzada, Mexico

Omar Nieva-García
Universidad del Istmo, Mexico

INTRODUCTION

The vast amounts of digital information stored in databases and other repositories represent a challenge for finding useful knowledge. Traditional methods for turning data into knowledge based on manual analysis reach their limits in this context, and for this reason, computer-based methods are needed. Knowledge Discovery in Databases (KDD) is the semi-automatic, nontrivial process of identifying valid, novel, potentially useful, and understandable knowledge (in the form of patterns) in data (Fayyad, Piatetsky-Shapiro, Smyth & Uthurusamy, 1996). KDD is an iterative and interactive process with several steps: understanding the problem domain, data preprocessing, pattern discovery, and pattern evaluation and usage. For discovering patterns, Data Mining (DM) techniques are applied.

A number of tools to help the analysts in the KDD process has been proposed. Most of them are stand-alone DM tools that require extracting data from the database and storing them in a file that is then passed as input of the DM engine. Additionally, these tools are often insufficiently equipped with data processing and pattern post-processing capabilities, leaving these tasks to the user, who needs to use other specialized tools with his or her own input and output formats. This results in time-consuming repeated export-import processes and difficulties to develop applications for knowledge discovery.

Analyzing data is then a complicated job because there is no common framework to manipulate data and patterns homogeneously. The Inductive Database (IDB) Framework (Boulicaut, Klemettinen & Mannila, 1999; De Raedt, 2002; Imielinski & Mannila, 1996; Meo, 2005) has been proposed to remedy this situation. In this approach, a database contains, in addition to the raw data, implicit or explicit patterns about the data. The discovery of patterns can then be viewed as a special kind of database interrogation where data and patterns can be queried. In this context, inductive query languages (IQLs) and associated evaluation techniques are being proposed.

This chapter explains the problems involved in the design of an IQL and its associated evaluation techniques, and presents some solutions to those problems. Our proposal (Nieva-García & Benítez-Guerrero, 2006) of an extension to SQL for extracting decision rules of the form *if <conditions> then <class>* to classify uncategorized data and associated relational-like operator will be presented as a case study, and similar existing works will be overviewed. Future trends will then be introduced, and finally, the chapter will be concluded.

BACKGROUND

The Traditional Framework for DB Querying

Research on query languages and associated evaluation techniques has a long tradition in the database area. Several query languages such as SQL, OQL, and XQUERY have been proposed. They enable the user to retrieve data from a database and filter these data according to specific selection criteria. To evaluate a query Q, the traditional process is as follows. First, Q is syntactically and semantically analyzed to check its syntax and verify if the schema elements referenced in Q exist in the database schema. Second, Q is translated into an expression in a query algebra represented as a query tree QT. Third, QT is optimized using heuristics and cost-based functions to devise an execution plan P with the minimal cost. Finally, P is executed to get the final results.

Knowledge Discovery in Databases and Data Mining

As stated earlier, the objective of the KDD process is to find interesting knowledge hidden in vast amounts of data. It involves three main phases: data preprocessing, data mining, and pattern postprocessing. Data preprocessing includes the selection of target data and their preparation (error detection and correction, transformation into alternative representations) for the data mining phase. Data Mining refers to the application of specific techniques and algorithms for extracting patterns from data. Finally, the pattern postprocessing step includes the evaluation of patterns to find relevant knowledge.

Several DM techniques, such as classification and link analysis, have been proposed (Witten & Frank, 2005). Classification is aimed at determining to which class (from a predefined set of categorical classes) a specific data item belongs (e.g., given a dataset about weather conditions, determine if it is possible to play golf or not). Link analysis is aimed at identifying, in a set of data items, relationships between attributes and items such as the presence of one pattern implies the presence of another pattern. These relationships may be associations between attributes within the same data item (association rules) (e.g., "60% of the customers who bought milk also purchased bread") or associations between data items over a period of time (sequential patterns) (e.g., "Customers that buy book A tend to buy book B two weeks later").

The KDD process is iterative and interactive in nature because the results obtained in one step may cause changes in the other steps. It is then important to develop conceptual and software

tools supporting the entire cycle and not only the DM step. Besides, a tight integration of these tools with database systems is also desirable in order to minimize operations to import, export, and transform data among tools.

Query Languages for Data Mining

The lack of a common framework to manipulate data and patterns makes difficult the job of the analysts and application developers. Imielinski and Mannila (1996) were the first to observe that the KDD process could be viewed as a special kind of database querying. The formalization of this observation led to the Inductive Database (IDB) Framework (Boulicaut et al. 1999; De Readt, 2002), where it is considered that a database contains, in addition to raw data, patterns about the data that can be extracted using an inductive query language.

This implies two things (Botta, Boulicaut, Masson & Meo, 2002): first, a data model expressive enough to represent both data and patterns is required, and second, the IQL should provide expressions enabling the user to query and filter them, and satisfy the *closure property* (i.e., the input of a query Q is one or more data structures of a kind; for instance, a relational table), and the result of Q is a data structure of the same kind, so it is possible to compose larger queries from smaller ones. The Object-Relational data model, because of its expressivity, has been used to represent data and patterns homogeneously. SQL is then the language of choice to query them, but it is necessary to augment it with features to support the KDD process. The following sections explain these aspects.

DATA AND PATTERNS MODEL

The Object-Relational (OR) model (Li, Lui & Orlowska, 1998) is basically an extension of the relational data model (attributes, tuples, tables) to include object-oriented concepts such as collections, tuple references, inheritance, and abstract data types (ADTs). An OR schema is a collection of row types, and an OR database is a collection of tables, each one containing tuples conformed to the row type of the table.

Consider, for instance, the database shown in Figure 1, which contains two tables: *Golf* and *Test*. The *Golf* table stores a set of weather conditions that can be used to decide if one can play Golf or not. Its row type is composed by the atomic attributes *outlook*, *temperature*, *humidity*, *windy*, and *play* (which is the attribute class). The *Test* table stores data related to weather conditions but without specifying a class for each tuple.

Tables are the input of an inductive query, and in order to guarantee the closure property, the result of the query should be a table. Each tuple of such a table will represent a pattern extracted from an input table. Thus, it is necessary to have particular row types. In Nieva-García and Benítez-Guerrero (2006), we defined the *decision-rule* row type to create tables storing decision rules. It is composed by the attributes id (a unique rule identifier), predicate (the "body" of the rule to predict a class)[1], class (the label to be assigned), and support and confidence, two accuracy measures of each rule. Figure 2 shows the Decision-Rules table, having *decision-rule* as row type, and containing a set of rules that describes the behavior of the *Golf* table. For instance, rule number one says that when attribute *outlook* is *overcast*, then the predicted class is *yes* with a *support* of 0.28 and a *confidence* of 1.0 (i.e., one can play Golf).

Note that it is possible to define other kinds of patterns in the same way. For instance, Wojciechowski (1999) proposes ADTs to represent the data elements (itemset, sequence, and pattern) manipulated during sequential pattern mining. The proposition of ADTs to represent other kinds of patterns is needed.

Figure 1. The golf database

Outlook	Temperature	Humidity	Windy	Play
sunny	hot	high	FALSE	no
sunny	hot	high	TRUE	no
overcast	hot	high	FALSE	yes
rainy	mild	high	FALSE	yes
rainy	cool	normal	FALSE	yes
rainy	cool	normal	TRUE	no
overcast	cool	normal	TRUE	Yes
sunny	mild	high	FALSE	no
sunny	cold	normal	FALSE	yes
rainy	mild	normal	FALSE	yes
sunny	mild	normal	TRUE	yes
overcast	mild	high	TRUE	yes
overcast	hot	normal	FALSE	yes
rainy	mild	high	TRUE	no

*(a) Table **Golf***

Outlook	Temperature	Humidity	Windy
sunny	hot	high	FALSE
sunny	hot	high	TRUE
overcast	hot	high	FALSE
rainy	mild	high	FALSE
overcast	cool	normal	TRUE
sunny	mild	high	FALSE
sunny	cold	normal	FALSE
rainy	cool	normal	TRUE
rainy	mild	normal	FALSE
sunny	mild	normal	TRUE
overcast	mild	high	TRUE
overcast	hot	normal	FALSE
rainy	mild	high	TRUE
overcast	cool	normal	TRUE

*(b) Table **Test***

Figure 2. Decision-rules table

ID	Predicate	Class	Support	Confidence
1	(<outlook = 'overcast'>, T)	yes	0.28	1.0
2	(<humidity = 'normal'>, <windy = 'false'>, T)	yes	0.28	1.0
3	(<temperature = 'mild'>, <humidity = 'normal'>, T)	yes	0.14	1.0
4	(<outlook = 'rainy'>, <windy = 'false'>, T)	yes	0.21	1.0
5	(<outlook = 'sunny'>, <humidity = 'high'>, T)	no	0.21	1.0
6	(<outlook = 'rainy'>, <windy = 'true'>, T)	no	0.14	1.0

EXPRESSING INDUCTIVE QUERIES

In the design of an IQL, some features need to be considered (Meo, Psaila & Ceri, 1996): (a) selection of the data to be mined, (b) specification of the patterns to be extracted, (c) definition of the constraints of the patterns to be extracted, and (d) postprocessing of the extracted patterns. Consider, for instance, our MINE DECISION RULE extension to SQL for extracting classification rules. Its syntax is as follows:

```
MINE DECISION RULE [<target>]
WITH <attribute> AS CLASS
FROM <table> | <sub-query>
[WHERE <conditions>]
```

The MINE DECISION RULE clause produces a new table that can be optionally renamed as indicated by <*target*>. The WITH <*attribute*> AS CLASS clause specifies the <*attribute*> that contains the classes to be predicted. The FROM clause defines the data source (a table or a subquery) for decision rule mining. Finally, the WHERE clause is optional and lets the user

specify <*conditions*> to filter a set of rules to get only those of interest. Consider the query Q: *Classify the data on the Test table based on rules extracted from the Golf table, considering* ***play*** *as class attribute.*

```
SELECT *
FROM Test AS T, (     MINE DECISION RULE
                      WITH play AS CLASS
                      FROM Golf ) AS DR
WHERE PredictionJoin( T, DR )
```

This query is an example of how to join (using the PredictionJoin function, which is used to classify a set of unclassified data) the table *Test*, containing uncategorized data with a table *DR* containing a set of rules. *DR* is the result of the evaluation of a MINE DECISION RULE expression that extracts rules from the *Golf* table using *play* as attribute class. The final result of this query is a table containing the tuples of *Test* already classified by the rules in *DR*.

Let us note that other SQL extensions have been proposed, enabling the users to express their mining requirements. For instance, the MINE RULE (Meo et al., 1996) and the MINE PATTERN (Wojciechowski, 1999) extensions have been proposed to mine association rules and sequential patterns, respectively.

PROCESSING INDUCTIVE QUERIES

As pointed out previously, query processing is complex, requiring in the first place to translate the query into an equivalent algebraic expression represented as a query tree. For processing inductive queries, we augmented the OR algebra introduced in Li, et al. (1998) with data mining operators. The OR algebra and an example of a mining operator are explained next.

Expressions in the OR algebra consist of OR operands and OR operators. An OR operand is either an OR table, a path expression, or the result of another operation. In this section, the term "table" will be used to refer to all the possible operands as long as no distinction is necessary. The set of OR operators is composed by the object-relational counterparts of basic relational operators (selection, projection, join, etc.), and by operators to handle tuple identity. To the original operator set we incorporated a special *prediction join* operator that applies a set of decision rules over unclassified data to classify them.

We extended this basic set of operators with DM operators to extract patterns from data. For instance, to evaluate MINE DECISION RULES expressions, we designed the MineDR operator (Nieva-García & Benítez-Guerrero, 2006). It is noted as $\Xi\ \boldsymbol{A}_{\mathbf{n}}(r)$, where r is an input table with schema $(A_1, A_2, ...,A_{n-1},A_n)$ such that A_k ($k = 1 ... n - 1$) are general attributes and A_n is a special attribute representing a label or class for each tuple in r. The MineDR operator computes as the result of a table having *decision-rule* as row type containing a set of rules extracted from table r.

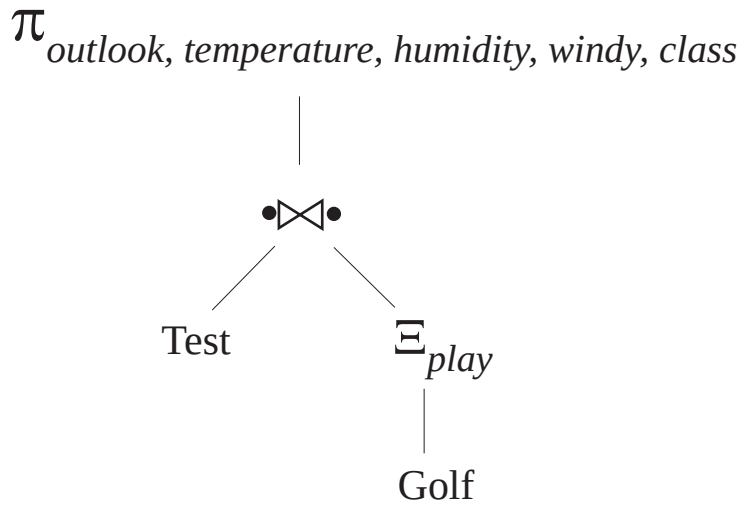
Consider our example query Q. ***Figure 3*** shows the corresponding query tree. A new table is produced by the MineDR operator by extracting decision rules from the *Golf* table. At the left side of the tree is the *Test* table with tuples representing instances to be classified. To obtain the result of the query (classified instances), the Prediction Join of the resulting table of MineDR with *Test* is executed. At the top of the query tree, all the attributes of *Test* plus an attribute containing the assigned class are projected.

Note that this is only an example of the way in which algebraic DM operators can be defined. In Benítez-Guerrero & Hernández-López (2006), we defined an operator for mining sequential patterns. The results obtained can be applied to propose operators for other DM techniques.

RELATED WORK

In order to discuss relevant related work, we have classified it in two categories: query lan-

Figure 3. Query tree for Q



guages and processing techniques. A number of IQLs have been proposed, usually as extensions of SQL, offering ADTs to represent different kinds of patterns and functions to manipulate them. Examples are MINE RULE (Meo et al., 1996), DMQL (Han et al., 1996), MineSQL (Wojciechowski, 1999), and DMX (Chaudhuri, Narasayya & Sarawagi, 2004). It has also been suggested to extend logic-based query languages with DM primitives (Nijssen & De Raedt, 2006). This is because patterns (i.e., rules) are naturally represented in propositional logic, and data and patterns can be also manipulated homogenously. These works are important, and it is expected that their results will be incorporated in the SQL context to achieve a higher degree of usage.

Some IQLs incorporate statements to mine several kind of patterns, while others focus on the extraction of a specific kind. Languages such as DMQL, DMX, and MineSQL incorporate several DM techniques as classification, clustering, and link analysis. MINE RULE is focused on the extraction of association rules, but this work was one of the first to show that it is possible to express pattern generation and postprocessing requirements in a single language.

Regarding query processing, most of these languages are evaluated in an ad-hoc manner. DM operators augmenting the underlying query algebra are implicitly defined. Note also that in order to evaluate an inductive query, current implementations depend on well-known algorithms to carry out the mining tasks. For instance, C4.5 is usually used to mine decision trees that are converted to rules, and a priori is applied to mine association rules. Other algorithms that can be useful are not considered. In our opinion, DM operators need to be explicitly defined as previously shown in order to able to reason about the whole query evaluation process, including the selection of the DM algorithm.

FUTURE TRENDS

Although an important number of DM techniques have been integrated into inductive query languages, it is still important to investigate to what extent other techniques (e.g., *Causal Networks*) can be incorporated. This will lead to ADTs defining data structures to represent knowledge that is different to rules, and functions to manipulate

those structures. New language expressions enabling the user to extract and use those patterns will then be proposed.

Future work also includes research on query optimization. Work is still needed to establish methods to decide which algorithm from a possible set is the most suitable for executing a particular algebraic DM operator. In Benítez-Guerrero and Hernández-López (2006), we proposed a semantic approach based on user expectations and environmental conditions. Similar methods based on data features such as the number of attributes and data volumes should be defined. Finally, the use of materialization and reuse of patterns in order to avoid unnecessary calculations and achieve higher performance levels has also been suggested (Wojciechowski, 1999). These aspects need to be explored.

To completely support the KDD process, some aspects related to data preprocessing and pattern postprocessing operations need to be integrated into an IQL. Particularly, for data preprocessing, it would be useful to have operations to remove or add data features such as noise. Similarly, support for pattern postprocessing needs to be extended from the application of patterns on data (crossover) to operations to manipulate sets of patterns.

CONCLUSION

This chapter presented the main aspects of the research on the expression and evaluation of inductive queries. First, the traditional framework for database querying and some general aspects of the KDD process and Data Mining were explained. Then, our main contributions, particularly the definition of an extension to SQL and a relational-like operator to mine decision rules, were introduced to illustrate the issues one has to face while designing an IQL and some possible solutions to those issues. Current related works were briefly described, comparing them with our proposal. Finally, some current trends were introduced.

REFRENCES

Benítez-Guerrero, E., & Hernández-López, A.R. (2006). *The mine SP operator for mining sequential patterns in inductive databases.* Proceedings of the MICAI 2006: Advances in Artificial Intelligence, 5th Mexican International Conference on Artificial Intelligence, 4293, 684–694.

Botta, M., Boulicaut, F.C., Masson, C., & Meo, R. (2002). *A comparison between query languages for the extraction of association rules.* Proceedings of the 4th International Conference on Data Warehousing and Knowledge Discovery (DaWaK'2000), 1–10.

Boulicaut, J.F., Klemettinen, M., & Mannila, H. (1999). *Modeling KDD processes within the inductive database framework.* Proceedings of the First International Conference on Data Warehousing and Knowledge Discovery (DaWaK'99), 293–302).

Chaudhuri, S., Narasayya, V., & Sarawagi, S. (2004). Extracting predicates from mining models for efficient query evaluation. *ACM Transactions on Database Systems, 29*(3), 508–544.

De Raedt, L. (2002). A perspective on inductive databases. *SIGKDD Explorations Newsletter, 4*(2) 69–77.

Fayyad, U.M., Piatetsky-Shapiro, G., Smyth, P., & Uthurusamy, R. (1996). *From data mining to knowledge discovery: An overview.* AAAI/MIT Press.

Han, J., et al. (1996). *DBMiner: A system for mining knowledge in large relational databases.* Proceedings of the International Conference on Data Mining and Knowledge Discovery (KDD'96), Portland, Oregon, 250–255.

Imielinski, T., & Mannila, H (1996). A database perspective on knowledge discovery. *Communications of the ACM, 39*(11), 58–64.

Li, H., Lui, C., & Orlowska, M. (1998). *A query systems for object-relational databases.* Proceedings of the 9th Australasian Database Conference (ADC'98), Pert, Australia,.39–50.

Meo, R. (2005). *Inductive databases: Towards a new generation of databases for knowledge discovery.* Proceedings of the DEXA'05: 16th International Workshop on Databases and Expert System Applications, 1003–1007.

Meo, R., Psaila, G., & Ceri, S. (1996). A new SQL-like operator for mining association rules. *The VLDB Journal,* 122–133.

Nieva-García, O., & Benítez-Guerrero, E. (2006). *On querying inductive databases to mine decision rules.* Proceedings of the 6th International Conference on Data Mining and Information Systems (ICDIS'06), 55–65.

Nijssen, S., & De Raedt, L.(2006). *IQL: A proposal for an inductive query language*. Proceedings of the 5th International Workshop on Knowledge Discovery (KDID'06), 107–118.

Witten, I., & Frank, E. (2005). *Data mining: Practical machine learning tools and techniques.* 2nd Edition. Morgan Kaufmann.

Wojciechowski, M. (1999). Mining various patterns in sequential data in an SQL-like manner. Proceedings of the Advances in Database and Information Systems Third East European Conference, ADBIS'99, Maribor, Slovenia, 131–138.

KEY TERMS

Abstract Data Type (ADT): Specification of a set of data and the set of operations that can be performed on the data.

Classification Model: A set of rules used to predict the class of an instance based on its attribute values.

Data Mining: The application of algorithms for discovering patterns in data.

Decision Rule: A rule of the form 'if <condition> then <class>', where <condition> is a Boolean combination of attribute tests and <class> is the class assigned to an instance satisfying the conditions.

Inductive Databases: Databases that besides raw data contain inductive generalizations about that data.

Inductive Query Language: A query language to perform various operations on data such as data preprocessing, pattern discovery, and pattern postprocessing.

Knowledge Discovery in Databases: The nontrivial process of identifying valid, novel, potentially useful, and ultimately understandable patterns in data.

Object-Relational Databases: The extension of relational databases to include object-oriented concepts such as collections, ADTs, tuple references, and inheritance.

Pattern: An expression in some language representing a high-level description of a dataset.

Prediction Join: An operator for applying a set of decision rules to classify uncategorized data.

ENDNOTE

[1] The type of the *predicate* attribute is *dr-pattern*. It is an ADT defining a structure with two attributes, one to represent the conditions of the rule and another to indicate if the rules must be tested against the data in order. It also defines methods to construct/manipulate decision rules.

Chapter LVI
Privacy-Preserving Data Mining

Alexandre Evfimievski
IBM Almaden Research Center, USA

Tyrone Grandison
IBM Almaden Research Center, USA

INTRODUCTION

Privacy-preserving data mining (PPDM) refers to the area of data mining that seeks to safeguard sensitive information from unsolicited or unsanctioned disclosure. Most traditional data mining techniques analyze and model the data set statistically, in aggregated form, while privacy preservation is primarily concerned with protecting against disclosure of individual data records. This domain separation points to the technical feasibility of PPDM.

Historically, issues related to PPDM were first studied by the national statistical agencies interested in collecting private social and economical data, such as census and tax records, and making it available for analysis by public servants, companies, and researchers. Building accurate socio-economical models is vital for business planning and public policy. Yet, there is no way of knowing in advance what models may be needed, nor is it feasible for the statistical agency to perform all data processing for everyone, playing the role of a trusted third party. Instead, the agency provides the data in a sanitized form that allows statistical processing and protects the privacy of individual records, solving a problem known as privacy-preserving data publishing. For a survey of work in statistical databases, see Adam and Wortmann (1989) and Willenborg and de Waal (2001).

The term privacy-preserving data mining was introduced in the papers Agrawal and Srikant (2000) and Lindell and Pinkas (2000). These papers considered two fundamental problems of PPDM: privacy-preserving data collection and mining a data set partitioned across several private enterprises. Agrawal and Srikant devised a randomization algorithm that allows a large number of users to contribute their private records for efficient centralized data mining while limiting the disclosure of their values; Lindell and Pinkas invented a cryptographic protocol for decision tree construction over a data set horizontally partitioned between two parties. These methods were subsequently refined and extended by many researchers worldwide.

Other areas that influence the development of PPDM include cryptography and secure multiparty computation (Goldreich, 2004; Stinson, 2006), database query auditing for disclosure

detection and prevention (Dinur & Nissim, 2003; Kenthapadi, Mishra, & Nissim, 2005; Kleinberg, Papadimitriou, & Raghavan, 2000), database privacy and policy enforcement (Aggarwal et al., 2004; Agrawal, Kiernan, Srikant, & Xu 2002), database security (Castano, Fugini, Martella, & Samarati, 1995), and of course, specific application domains.

SURVEY OF APPROACHES

The naïve approach to PPDM is "security by obscurity," where algorithms have no proven privacy guarantees. By its nature, privacy preservation is claimed for all data sets and attacks of a certain class, a claim that cannot be proven by examples or informal considerations (Chawla, Dwork, McSherry, Smith, & Wee, 2005). We will avoid further discussion of this approach in this forum. Recently, however, a number of principled approaches have been developed to enable PPDM, some listed below according to their method of defining and enforcing privacy.

Suppression

Privacy can be preserved by simply suppressing all sensitive data before any disclosure or computation occurs. Given a database, we can suppress specific attributes in particular records as dictated by our privacy policy. For a partial suppression, an exact attribute value can be replaced with a less informative value by rounding (e.g., $23.45 to $20.00), top coding (e.g., age above 70 is set to 70), generalization (e.g., address to zip code), using intervals (e.g., age 23 to 20-25, name Johnson to J-K), and so forth. Often the privacy guarantee trivially follows from the suppression policy. However, the analysis may be difficult if the choice of alternative suppressions depends on the data being suppressed, or if there is dependency between disclosed and suppressed data. Suppression cannot be used if data mining requires full access to the sensitive values.

Rather than protecting the sensitive values of individual records, we may be interested in suppressing the identity (of a person) linked to a specific record. The process of altering the data set to limit identity linkage is called de-identification. One popular definition for de-identification privacy is *k*-anonymity, formulated in Samarati and Sweeney (1998). A set of personal records is said to be *k*-anonymous if every record is indistinguishable from at least $k - 1$ other records over given quasi-identifier subsets of attributes. A subset of attributes is a quasi-identifier if its value combination may help link some record to other personal information available to an attacker, for example, the combination of age, sex, and address.

To achieve *k*-anonymity, quasi-identifier attributes are completely or partially suppressed. A particular suppression policy is chosen to maximize the utility of the *k*-anonymized data set (Bayardo & Agrawal, 2005; Iyengar, 2002). The attributes that are not among quasi-identifiers, even if sensitive (e.g., diagnosis), are not suppressed and may get linked to an identity (Machanavajjhala, Gehrke, Kifer, & Venkitasubramaniam, 2006). Utility maximization may create an exploitable dependence between the suppressed data and the suppression policy. Finally, *k*-anonymity is difficult to enforce before all data are collected in one trusted place; however, a cryptographic solution is proposed in Zhong, Yang, and Wright (2005) based on Shamir's secret sharing scheme.

Suppression can also be used to protect from the discovery of certain statistical characteristics, such as sensitive association rules, while minimizing the distortion of other data mining results. Many related optimization problems are computationally intractable, but some heuristic algorithms were studied (Atallah, Elmagarmid, Ibrahim, Bertino, & Verykios, 1999; Oliveira & Zaïane, 2003).

Randomization

Suppose there is one central server, for example, of a company, and many customers, each having

a small piece of information. The server collects the information and performs data mining to build an aggregate data model. The randomization approach (Warner, 1965) protects the customers' data by letting them randomly perturb their records before sending them to the server, taking away some true information and introducing some noise. At the server's side, statistical estimation over noisy data is employed to recover the aggregates needed for data mining. Noise can be introduced, for example, by adding or multiplying random values to numerical attributes (Agrawal & Srikant, 2000) or by deleting real items and adding bogus items to set-valued records (Evfimievski, Srikant, Agrawal, & Gehrke, 2002; Rizvi & Haritsa, 2002). Given the right choice of the method and the amount of randomization, it is sometimes possible to protect individual values while estimating the aggregate model with relatively high accuracy.

Privacy protection by data perturbation has been extensively studied in the statistical databases community (Adam & Wortmann, 1989; Willenborg & de Waal, 2001). In contrast to the above scenario, this research focuses mainly on the protection of published views once all original data are collected in a single trusted repository. Many more perturbation techniques are available in this case, including attribute swapping across records and data resampling by imputation.

A popular privacy definition to characterize randomization has its roots in the classical secrecy framework (Shannon, 1949) and in the work on disclosure risk and harm measures for statistical databases (Lambert, 1993), but received its current formulation only recently (Blum, Dwork, McSherry, & Nissim, 2005; Dinur & Nissim, 2003; Evfimievski, Gehrke, & Srikant, 2003). To deal with the uncertainty arising from randomization, the data miner's knowledge (belief) is modeled as a probability distribution. A simplified version of the definition is given in the next paragraphs.

Suppose Alice is a customer and Bob is a company employee interested in mining customers' data. Alice has a private record x and a randomization algorithm R. To allow Bob to do the mining while protecting her own privacy, Alice sends Bob a randomized record $x' \sim R(x)$. Let us denote by $p_R(x'|x)$ the probability that algorithm R outputs x' on input x. We say that algorithm R achieves ε-leakage (also called ε-privacy or at most γ-amplification) at output x' if for every pair of private records x_1 and x_2 we have

$$p_R(x'|x_1) / p_R(x'|x_2) \leq \gamma, \quad \text{where } \gamma = \exp(\varepsilon).$$

We assume that Bob has some a priori belief about Alice's record, defined as the probability distribution $p(x)$ over all possible private records. Once Bob receives a randomized record, his belief changes to some a posteriori distribution. If randomization R achieves ε-leakage at output x', then randomized record x' gives Bob only a bounded amount of knowledge of Alice's unknown private record x. In fact, for every question Q about Alice's record, Bob's a posteriori belief $p(Q|x')$ that the answer to Q is yes is bounded with respect to his a priori belief $p(Q)$ as follows.

$$\frac{p(Q|x')}{1-p(Q|x')} \leq \gamma \frac{p(Q)}{1-p(Q)}$$

If R achieves ε-leakage at every output, Bob's knowledge gain about Alice's record is always bounded; if R achieves ε-leakage at some outputs but not others, Bob's knowledge gain is bounded only with a certain probability.

The above definition assumes that Bob cannot gain any knowledge of Alice's record by collecting data from other customers, that is, that all customers are independent. The parameter ε is chosen to attain the right balance between privacy and the accuracy of the aggregate estimators used by the data miner (Dwork, McSherry, Nissim, & Smith, 2006). One advantage of randomization is that privacy guarantees can be proven by just studying the randomization algorithm, not the data mining operations. One disadvantage is that the results are always approximate; high-enough accuracy often requires a lot of randomized data (Evfimievski et al., 2002).

Cryptography

The cryptographic approach to PPDM assumes that the data are stored at several private parties who agree to disclose the result of a certain data mining computation performed jointly over their data. The parties engage in a cryptographic protocol; that is, they exchange messages encrypted to make some operations efficient while others computationally intractable. In effect, they blindly run their data mining algorithm. Classical works in secure multiparty computation such as Yao (1986) and Goldreich, Micali, and Wigderson (1987) show that any function $F(x_1, x_2, \ldots, x_n)$ computable in polynomial time is also securely computable in polynomial time by n parties, each holding one argument, under quite broad assumptions regarding how much the parties trust each other. However, this generic methodology can only be scaled to database-sized arguments with significant additional research effort.

The first adaptation of cryptographic techniques to data mining is done by Lindell and Pinkas (2000) for the problem of decision tree construction over horizontally partitioned data; it was followed by many papers covering different data mining techniques and assumptions. The assumptions include restrictions on the input data and permitted disclosure, the computational hardness of certain mathematical operations such as factoring a large integer, and the adversarial potential of the parties involved: The parties may be passive (honest but curious, running the protocol correctly but taking advantage of all incoming messages) or malicious (running a different protocol), some parties may be allowed to collude (represent a single adversary), and so forth. In addition to the generic methodology such as oblivious transfer and secure Boolean circuit evaluation, the key cryptographic constructs often used in PPDM include homomorphic and commutative encryption functions, secure multiparty scalar product, and polynomial computation. The use of randomness is essential for all protocols.

The privacy guarantee used in this approach is based on the notion of computational indistinguishability between random variables. Let X_k and Y_k be two random variables that output Boolean vectors of length polynomial in k; they are called computationally indistinguishable if for all polynomial algorithms A_k (alternatively, for any sequence of circuits of size polynomial in k), for all $c > 0$, and for all sufficiently large integers k,

$$|\text{Prob}\,[A_k(X_k) = 1] - \text{Prob}\,[A_k(Y_k) = 1]| < 1/k^c.$$

The above essentially says that no polynomial algorithm can tell apart X_k from Y_k. To prove that a cryptographic protocol is secure, we show that each party's view of the protocol (all its incoming messages and random choices) is computationally indistinguishable from a simulation of this view by this party alone. When simulating the view of the protocol, the party is given everything it is allowed to learn, including the final data mining output. The exact formulation of the privacy guarantee depends on the adversarial assumptions. Goldreich (2004) and Stinson (2006) provide a thorough introduction into the cryptographic framework.

Scalability is the main stumbling block for the cryptographic PPDM; the approach is especially difficult to scale when more than a few parties are involved. Also, it does not address the question of whether the disclosure of the final data mining result may breach the privacy of individual records.

Summarization

This approach to PPDM consists of releasing the data in the form of a summary that allows the (approximate) evaluation of certain classes of aggregate queries while hiding the individual records. In a sense, summarization extends randomization, but a summary is often expected to be much shorter, ideally of sublinear size with respect to the original data set. The idea goes back to statistical databases, where two summarization techniques were studied and widely applied: sampling and tabular data representation (Adam &

Wortmann, 1989; Willenborg & de Waal, 2001). Sampling corresponds to replacing the private data set with a small sample of its records, often combined with suppression or perturbation of their values to prevent re-identification. Tabular representation summarizes data in a collection of aggregate quantities such as sums, averages, or counts aggregated over the range of some attributes while other attributes are fixed, similarly to OLAP (online analytical processing) cubes. Verifying privacy guarantees for tabular data is challenging because of the potential for disclosure by inference. Privacy in OLAP cubes was also studied by Wang, Jajodia, and Wijesekera (2004) and Agrawal, Srikant, and Thomas (2005).

Some of the more recent summarization methods are based on pseudorandom sketches, a concept borrowed from limited-memory data-stream processing. Here is an illustration of one such method. Suppose Alice has a small private set S of her favorite book titles and wants to send to Bob a randomized version of this set. Alice splits S into two disjoint subsets, $S = S_0 \cup S_1$, then constructs her randomized record S_R by including S_1, excluding S_0, and for every book not in S including it into S_R at random with probability 1/2. If there are 1,000,000 possible book titles, S_R will contain around 500,000 items, most of them purely random. Luckily, however, S_R can be shortened. Let $G(\xi, i)$ be a pseudorandom generator that takes a short random seed ξ and a book number i and computes a bit b_i. Now Alice has a better strategy: Once she selects S_0 and S_1 as before, she sends to Bob a randomly chosen seed ξ such that $G(\xi, \#\text{book}) = 0$ for all books in S_0 and $G(\xi, \#\text{book}) = 1$ for all books in S_1. Bob can use G and ξ to reconstruct the entire randomized record; and if G is sufficiently well mixing, every book not in S still satisfies $G(\xi, \#\text{book}) = 1$ with probability 1/2. Thus, the short seed ξ serves as the summary of a randomized record. For complete analysis, see Evfimievski et al. (2003) and Mishra and Sandler (2006).

The summarization approach is still in its infancy, so more results are likely to come in the future. There has also been some work on combining sketches and approximation techniques with the cryptographic approach (Feigenbaum, Ishai, Malkin, Nissim, Strauss, & Wright, 2001; Halevi, Krauthgamer, Kushilevitz, & Nissim, 2001). Feigenbaum et al. observe that the disclosure of an approximate function $f_{\text{appr}}(x) \approx f(x)$ over private data x may be unacceptable even if the exact result $f(x)$ is permitted to disclose; indeed, just by learning whether $f_{\text{appr}}(x) \le f(x)$ or $f_{\text{appr}}(x) > f(x)$, the adversary may already get an extra bit of information about x. This issue is important to keep in mind when designing sketch-based PPDM protocols.

APPLICATION SCENARIOS

- **Surveys and Data Collection.** Companies collect personal preferences of their customers for targeted product recommendations, or conduct surveys for business planning; political parties conduct opinion polls to adjust their strategy. The coverage of such data collection may significantly increase if all respondents are aware that their privacy is provably protected, also eliminating the bias associated with evasive answers. The randomization approach has been considered as a solution in this domain (Agrawal & Srikant, 2000; Evfimievski et al., 2002; Warner, 1965).
- **Monitoring for Emergencies.** Early detection of large-scale abnormalities with potential implications for public safety or national security is important in protecting our well-being. Disease outbreaks, environmental disasters, terrorist acts, and manufacturing accidents can often be detected and contained before they endanger a large population. The first indication of an impending disaster can be difficult to notice by looking at any individual case, but is easy to see using data mining: an unusual increase in certain health symptoms or nonprescription drug purchases, a surge in car accidents, a change in online traffic pattern, and so forth.

To be effective, an early-detection system would have to collect personal, commercial, and sensor data from a variety of sources, making privacy issues paramount (DeRosa, 2004; Perez-Pena, 2003).

- **Product Traceability.** Before a product (e.g., a car or a drug) reaches its end user, it usually passes through a long chain of processing steps, such as manufacturing, packaging, transportation, storage, and sale. In the near future, many products and package units will carry a radio frequency identification (RFID) tag and will be automatically registered at every processing step (Finkenzeller, 2003; Garfinkel & Rosenberg, 2005). This will create a vast distributed collection of RFID traces, which can be mined to detect business patterns, market trends, inefficiencies and bottlenecks, criminal activity such as theft and counterfeiting, and so on. However, such extremely detailed business process data are a highly valuable and sensitive asset to the companies involved. Privacy safeguards will be very important to enable cooperative RFID data mining.
- **Medical Research.** Personal health records are one of the most sensitive types of private data; their privacy standards have been codified into law in many countries, for example HIPAA (Health Insurance Portability and Accountability Act) in the United States (Office for Civil Rights [OCR], 2003). On the other hand, data mining over health records is vital for medical, pharmaceutical, and environmental research. For example, a researcher may want to study the effect of a certain gene *A* on an adverse reaction to drug *B* (Agrawal, Evfimievski, & Srikant, 2003). However, due to privacy concerns, the DNA sequences and the medical histories are stored at different data repositories and cannot be brought together. Then, PPDM over vertically partitioned data can be used to compute the aggregate counts while preserving the privacy of records.
- **Social Networks.** In business as well as in life, the right connections make a huge difference. Whether it is expertise location, job search, or romance matchmaking, finding new connections is notoriously difficult, not least because the publicly available data are often very scarce and of low quality. Most of the relevant information is personal, copyrighted, or confidential, and therefore kept away from the Web. It is possible that PPDM techniques can be utilized to allow limited disclosure options, prompting more people to engage in productive social networking and guarding against abuse.

FUTURE TRENDS

The main technical challenge for PPDM is to make its algorithms scale and achieve higher accuracy while keeping the privacy guarantees. The known proof techniques and privacy definitions are not yet flexible enough to take full advantage of existing PPDM approaches. Adding a minor assumption (from the practical viewpoint) may slash the computation cost or allow much better accuracy if the PPDM methodology is augmented to leverage this assumption. On the other hand, proving complexity lower bounds and accuracy upper bounds will expose the theoretical limits of PPDM.

One particularly interesting minor assumption is the existence of a computationally limited trusted third party. Computer manufacturers such as IBM produce special devices called secure coprocessors (Dyer et al. 2001) that contain an entire computer within a sealed tamper-proof box. Secure coprocessors are able to withstand most hardware and software attacks, or destroy all data if opened. For practical purposes, these devices can be considered trusted parties, albeit very restricted in the speed of computation, in the volume of storage, and in communication with the untrusted components. It is known that secure coprocessors can be leveraged to enable privacy-preserving operations over data sets much larger than their storage capacity

(Agrawal, Asonov, Kantarcioglu, & Li, 2006; Smith & Safford, 2000). Thus, applying them to PPDM looks natural.

If a data mining party cannot get accurate results because of privacy constraints enforced by the data contributors, it may be willing to pay for more data. Kleinberg, Papadimitriou, and Raghavan (2001) suggest measuring the amount of private information in terms of its monetary value as a form of intellectual property. The cost of each piece of data must be determined in a fair way so as to reflect the contribution of this piece in the overall profit. The paper borrows the notions of fairness from the theory of coalitional games: the core and the Shapley value. Bridging game theory and PPDM could lay the theoretical foundation for a market of private data, where all participants receive appropriate compensations for their business contribution.

Among potential future applications for PPDM, we would like to emphasize data mining in health care and medical research. During the last few years, the attention of the U.S. government has been focused on transitioning the national health care system to an infrastructure based upon information technology (President's Information Technology Advisory Committee [PITAC], 2004); a similar trend occurs or is expected in countries around the world. Within a short time, millions of medical records will be available for mining, and their privacy protection will be required by law, potentially creating an urgent demand for PPDM. In addition to the traditional data mining tasks, new health-care-specific tasks will likely become important, such as record linkage or mining over ontology-based and semistructured data, for example, annotated images.

CONCLUSION

Privacy-preserving data mining emerged in response to two equally important (and seemingly disparate) needs: data analysis in order to deliver better services and ensuring the privacy rights of the data owners. Difficult as the task of addressing these needs may seem, several tangible efforts have been accomplished. In this article, an overview of the popular approaches for doing PPDM was presented, namely, suppression, randomization, cryptography, and summarization. The privacy guarantees, advantages, and disadvantages of each approach were stated in order to provide a balanced view of the state of the art. Finally, the scenarios where PPDM may be used and some directions for future work were outlined.

REFERENCES

Adam, N. R., & Wortmann, J. C. (1989). Security-control methods for statistical databases: A comparative study. *ACM Computing Surveys, 21*(4), 515-556.

Aggarwal, G., Bawa, M., Ganesan, P., Garcia-Molina, H., Kenthapadi, K., Mishra, N., et al. (2004). Vision paper: Enabling privacy for the paranoids. *Proceedings of the 30th International Conference on Very Large Data Bases (VLDB'04)* (pp. 708-719).

Agrawal, R., Asonov, D., Kantarcioglu, M., & Li, Y. (2006). *Sovereign joins.* Proceedings of the 22nd International Conference on Data Engineering (ICDE'06).

Agrawal, R., Evfimievski, A., & Srikant, R. (2003). Information sharing across private databases. *Proceedings of ACM SIGMOD International Conference on Management of Data (SIGMOD'03)* (pp. 86-97).

Agrawal, R., Kiernan, J., Srikant, R., & Xu, Y. (2002). *Hippocratic databases.* Proceedings of the 28th International Conference on Very Large Data Bases (VLDB'02), Hong Kong, China.

Agrawal, R., & Srikant, R. (2000). *Privacy preserving data mining.* Proceedings of ACM SIGMOD Conference on Management of Data (SIGMOD'00), Dallas, TX.

Agrawal, R., Srikant, R., & Thomas, D. (2005). Privacy preserving OLAP. *Proceedings of ACM*

SIGMOD International Conference on Management of Data (SIGMOD'05) (pp. 251-262).

Atallah, M., Elmagarmid, A., Ibrahim, M., Bertino, E., & Verykios, V. (1999). Disclosure limitation of sensitive rules. *Proceedings of the 1999 Workshop on Knowledge and Data Engineering Exchange* (pp. 45-52).

Bayardo, R. J., & Agrawal, R. (2005). Data privacy through optimal *k*-anonymization. *Proceedings of the 21st International Conference on Data Engineering (ICDE'05)* (pp. 217-228).

Blum, A., Dwork, C., McSherry, F., & Nissim, K. (2005). *Practical privacy: The SuLQ framework.* Proceedings of ACM Symposium on Principles of Database Systems (PODS'05), Baltimore.

Castano, S., Fugini, M., Martella, G., & Samarati, P. (1995). *Database security.* Addison Wesley.

Chawla, S., Dwork, C., McSherry, F., Smith, A., & Wee, H. (2005). Towards privacy in public databases. *Proceedings of 2nd Theory of Cryptography Conference* (pp. 363-385).

DeRosa, M. (2004). *Data mining and data analysis for counterterrorism.* Center for Strategic and International Studies.

Dinur, I., & Nissim, K. (2003). *Revealing information while preserving privacy.* Proceedings of ACM Symposium on Principles of Database Systems (PODS'03), San Diego, CA.

Dwork, C., McSherry, F., Nissim, K., & Smith, A. (2006). *Calibrating noise to sensitivity in private data analysis.* Proceedings of Theory of Cryptography (TCC'06), New York.

Dyer, J. G., Lindemann, M., Perez, R., Sailer, R., Smith, S. W., van Doorn, L., et al. (2001). The IBM secure coprocessor: Overview and retrospective. *IEEE Computer*, pp. 57-66.

Evfimievski, A., Gehrke, J., & Srikant, R. (2003). Limiting privacy breaches in privacy preserving data mining. *Proceedings of ACM Symposium on Principles of Database Systems (PODS'03)* (pp. 211-222).

Evfimievski, A., Srikant, R., Agrawal, R., & Gehrke, J. (2002). Privacy preserving mining of association rules. *Proceedings of 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'02)* (pp. 217-228).

Feigenbaum, J., Ishai, Y., Malkin, T., Nissim, K., Strauss, M., & Wright R. (2001). Secure multiparty computation of approximations. *Proceedings of 28th International Colloquium on Automata, Languages and Programming* (pp. 927-938).

Finkenzeller, K. (2003). *RFID handbook: Fundamentals and applications in contactless smart cards and identification* (2nd ed.). John Wiley & Sons.

Garfinkel, S., & Rosenberg, B. (2005). *RFID: Applications, security, and privacy.* Addison-Wesley Professional.

Goldreich, O. (2004). *Foundations of cryptography* (Vol. I, II). Cambridge University Press.

Goldreich, O., Micali, S., & Wigderson, A. (1987). How to play any mental game. *Proceedings of 19th Annual ACM Conference on Theory of Computing (STOC'87)* (pp. 218-229).

Halevi, S., Krauthgamer, R., Kushilevitz, E., & Nissim, K. (2001). Private approximation of NP-hard functions. *Proceedings of the 33rd Annual ACM Symposium on Theory of Computing (STOC'01)* (pp. 550-559).

Iyengar, V. S. (2002). Transforming data to satisfy privacy constraints. *Proceedings of 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'02)* (pp. 279-288).

Kenthapadi, K., Mishra, N., & Nissim, K. (2005). Simulatable auditing. *Proceedings of ACM Symposium on Principles of Database Systems (PODS'05)* (pp. 118-127).

Kleinberg, J. M., Papadimitriou, C. H., & Raghavan, P. (2000). Auditing Boolean attributes. *Pro-*

ceedings of ACM Symposium on Principles of Database Systems (PODS'00) (pp. 86-91).

Kleinberg, J. M., Papadimitriou, C. H., & Raghavan, P. (2001). On the value of private information. *Proceedings of the 8th Conference on Theoretical Aspects of Rationality and Knowledge.*

Lambert, D. (1993). Measures of disclosure risk and harm. *Journal of Official Statistics, 9*(2), 313-331.

Lindell, Y., & Pinkas, B. (2000). Privacy preserving data mining. In *Lecture notes in computer science: Vol. 1880. Proceedings of Advances in Cryptology: Crypto'00* (pp. 20-24). Springer-Verlag.

Machanavajjhala, A., Gehrke, J., Kifer, D., & Venkitasubramaniam, M. (2006). *l-diversity: Privacy beyond k-anonymity.* Proceedings of 22nd IEEE International Conference on Data Engineering (ICDE'06), Atlanta, GA.

Mishra, N., & Sandler, M. (2006). Privacy via pseudorandom sketches. *Proceedings of ACM Symposium on Principles of Database Systems (PODS'06)* (pp. 143-152).

Office for Civil Rights (OCR). (2003). *Summary of the HIPAA privacy rule.* U.S. Department of Health and Human Services.

Oliveira, S. R. M., & Zaïane, O. R. (2003). Protecting sensitive knowledge by data sanitization. *Proceedings of the 3rd IEEE International Conference on Data Mining (ICDM'03)* (pp. 613-616).

Perez-Pena, R. (2003, April 4). An early warning system for diseases in New York. *New York Times*.

President's Information Technology Advisory Committee (PITAC). (2004). *Revolutionizing health care through information technology.* Author.

Rizvi, S. J., & Haritsa, J. R. (2002). *Maintaining data privacy in association rule mining.* Proceedings of the 28th International Conference on Very Large Data Bases (VLDB'02), Hong Kong, China.

Samarati, P., & Sweeney, L. (1998). *Protecting privacy when disclosing information: k-anonymity and its enforcement through generalization and suppression.* Proceedings of the IEEE Symposium on Research in Security and Privacy, Oakland, CA.

Shannon, C. E. (1949). Communication theory of secrecy systems. *Bell System Technical Journal, 28*(4), 656-715.

Smith, S. W., & Safford, D. (2000). *Practical private information retrieval with secure coprocessors* (Research Rep. No. RC-21806). IBM T. J. Watson Research Center.

Stinson, D. R. (2006). *Cryptography theory and practice* (3rd ed.). Chapman & Hall/CRC.

Wang, L., Jajodia, S., & Wijesekera, D. (2004). Securing OLAP data cubes against privacy breaches. *Proceedings of 2004 IEEE Symposium on Security and Privacy* (pp. 161-175).

Warner, S. L. (1965). Randomized response: A survey technique for eliminating evasive answer bias. *Journal of the American Statistical Association, 60*(309), 63-69.

Willenborg, L., & de Waal, T. (2001). Elements of statistical disclosure control. In *Lecture notes in statistics* (Vol. 155). Springer-Verlag.

Yao, A. C. (1986). How to generate and exchange secrets. *Proceedings of the 27th Annual Symposium on Foundations of Computer Science (FOCS'86)* (pp. 162-167).

Zhong, S., Yang, Z., & Wright, R. N. (2005). Privacy-enhancing *k*-anonymization of customer data. *Proceedings of the 24th ACM Symposium on Principles of Database Systems (PODS'05)* (pp. 139-147).

KEY TERMS

Data Mining: The process of automatically searching large volumes of data for patterns.

De-Identification: Altering a data set to limit identity linkage.

Privacy: Individuals or groups' right to control the flow of sensitive information about themselves.

Pseudorandom Generator: An algorithm that takes a short random seed and outputs a long sequence of bits that look independent random under a certain class of tests.

Randomization: The act of haphazardly perturbing data before disclosure.

Secure Multiparty Computation: A method for two or more parties to perform a joint computation without revealing their inputs.

Statistical Database: A database with social or economical data used for aggregate statistical analysis rather than for the retrieval of individual records.

Summarization: Transforming a data set into a short summary to allow statistical analysis while losing the individual records.

Suppression: Withholding information due to disclosure constraints.

Chapter LVII
Mining Frequent Closed Itemsets for Association Rules

Anamika Gupta
University of Delhi, India

Shikha Gupta
University of Delhi, India

Naveen Kumar
University of Delhi, India

INTRODUCTION

Association refers to correlations that exist among data. Association Rule Mining (ARM) is an important data-mining task. It refers to discovery of rules between different sets of attributes/items in very large databases (Agrawal R. & Srikant R. 1994). The discovered rules help in strategic decision making in both commercial and scientific domains.

A classical application of ARM is market basket analysis, an application of data mining in retail sales where associations between the different items are discovered to analyze the customer's buying habits in order to develop better marketing strategies. ARM has been extensively used in other applications like spatial-temporal, health care, bioinformatics, web data etc (Han J., Cheng H., Xin D., & Yan X. 2007).

Association Rule mining is decomposed in two steps 1) Generate all frequent itemsets (FI) 2) Generate confident rules using the frequent itemsets discovered in first step. The first step is computationally more expensive task and has attracted attention of most researchers. Many researchers have given different algorithms for mining FI (Han J., Cheng H., Xin D., & Yan X. 2007). However set of FI is often very large. Frequent Closed Itemsets (FCI) is a reduced, complete and loss less representation of FI and is often much less in number than FI. *Closed itemset* is an itemset whose support is not equal to support of any of its proper superset (Zaki M. J. & Hsiao C. J. 1999). Closed itemsets with support greater than the user specified support threshold (ms) are *frequent closed itemsets (FCI)*. Thus mining FCI instead of FI in association rule discovery procedure saves computation and memory efforts.

In this article we discuss the importance of mining FCI instead of FI in association rule discovery procedure. We explain different approaches and techniques for mining FCI in datasets.

BACKGROUND

Let *D* denotes the database of *N* transactions and *I* denotes the set of *n* items in *D*. A set of one or more items belonging to set *I* is termed as an itemset. A k-itemset is an itemset of cardinality k. A transaction $T \in D$ contains an itemset and is associated with a unique identifier *TID*. The probability of an itemset *X* being contained in a transaction is termed as *support* of *X*.

$$Support(X) = P(X) = \frac{No\_of\_transactions\_containing\_X}{N}$$

An itemset having support greater than the user specified support threshold (*ms*) is termed as *frequent itemset (FI)*.

An *association rule* is an implication of the form $X \rightarrow Y$ where $X \subset I$, $Y \subset I$ and $X \cap Y = \varnothing$. *Support* and *Confidence* are rule evaluation metrics of an *association rule*. *Support* of a rule $X \rightarrow Y$ in D is 'sup' if *sup*% of transactions in *D* contain $X \cup Y$. It is computed as:

$$Support(X \rightarrow Y) = P(X \cup Y) = \frac{No.\_of\_transactions\_containing\_X \cup Y}{N}$$

Confidence of a rule $X \rightarrow Y$ in *D* is '*conf*' if *conf*% of transactions in *D* that contain *X*, also contain *Y*. It is computed, as the conditional probability that *Y* occurs in a transaction given *X* is present in the same transaction, i.e.

$$Confidence(X \rightarrow Y) = P(Y/X) = \frac{P(X \cup Y)}{P(X)} = \frac{Support(X \cup Y)}{Support(X)}$$

A rule generated from frequent itemsets is *strong* if its *confidence* is greater than the user specified confidence threshold (*mc*).

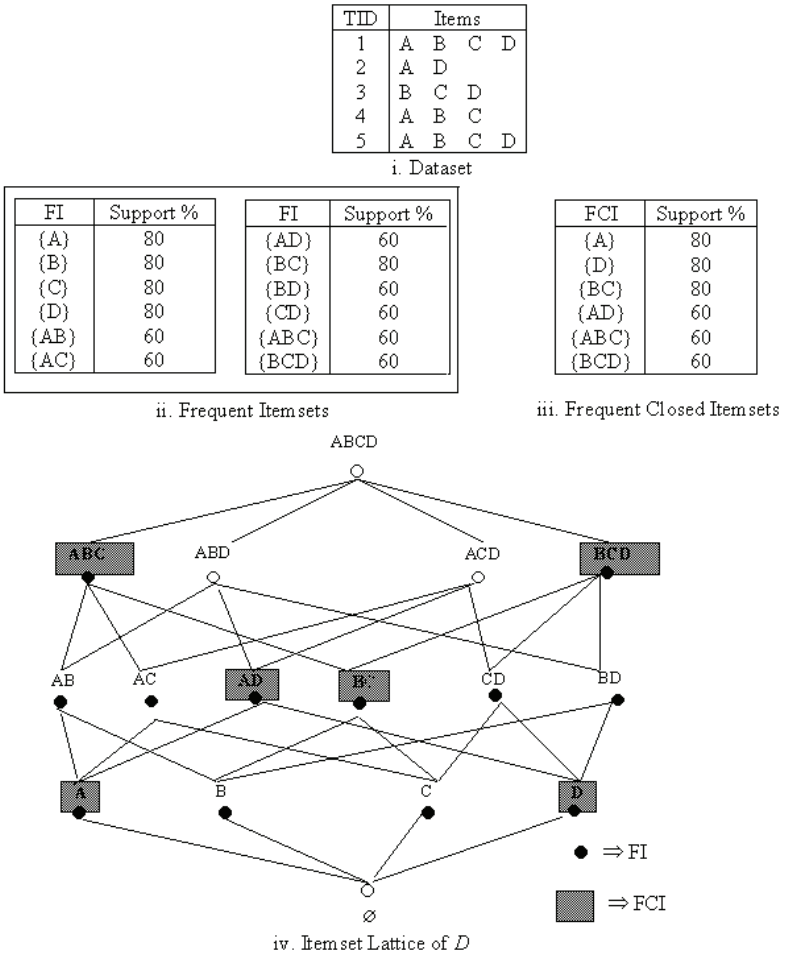
Three independent groups of researchers (Pasquier N., Bastide Y., Taouil R. & Lakhal L. 1999a), (Zaki M.J. & Hsiao C. J. 1999), (Stumme G., 1999) introduced the notion of mining FCI instead of FI and have given the following definitions of FCI:

Zaki et al. defines *closed itemset* as an itemset whose support is not equal to support of any of its proper superset (Closure Property) (Zaki M. J. & Hsiao C. J. 1999). In other words *X* is a closed itemset if there exists no proper superset *X'* of *X* such that support(*X*) = support(*X'*). Closed itemsets with support greater than the user specified support threshold (ms) are *frequent closed itemsets (FCI)*.

Pasquier et. al. defines c*losed itemset* in terms of G*alois closure operator* (Pasquier N., Bastide Y., Taouil R. & Lakhal L. 1999a). *Galois closure operator h(X)* for some $X \subseteq I$ is defined as the intersection of transactions in *D* containing itemset *X*. An itemset *X* is a *closed itemset* if and only if *h(X) = X*.

Stumme G. (1999) defines c*losed itemset* as the intent of *formal concept* where *formal concept* is a core structure in Formal Concept Analysis (FCA), a branch of mathematics based on c*oncepts* and c*oncept hierarchies* (Ganter B. & Wille R. 1999). A *formal concept* (*A,B*) is defined as a pair of set of objects *A* (known as *extent*) and set of attributes *B* (known as *intent*) such that set of all attributes belonging to *extent A* is same as *B* and set of all objects containing attributes of *intent B* is same as *A*. That is, no object other than objects of set *A* contains all attributes of *B* and no attribute other than attributes in set *B* is contained in all objects of set *A*. Stumme G. (1999) discovered that *intent B* of the *Formal Concept* (*A,B*) represents the closed itemset. Fig. 1 shows an example dataset (*D*) of five transactions with its itemset lattice and the list of frequent itemsets, frequent closed itemsets in the same example.

Figure 1. Example dataset (D), Frequent itemsets, Frequent closed itemsets and itemset lattice, ms = 60%



ADVANTAGES OF MINING FCI OVER FI

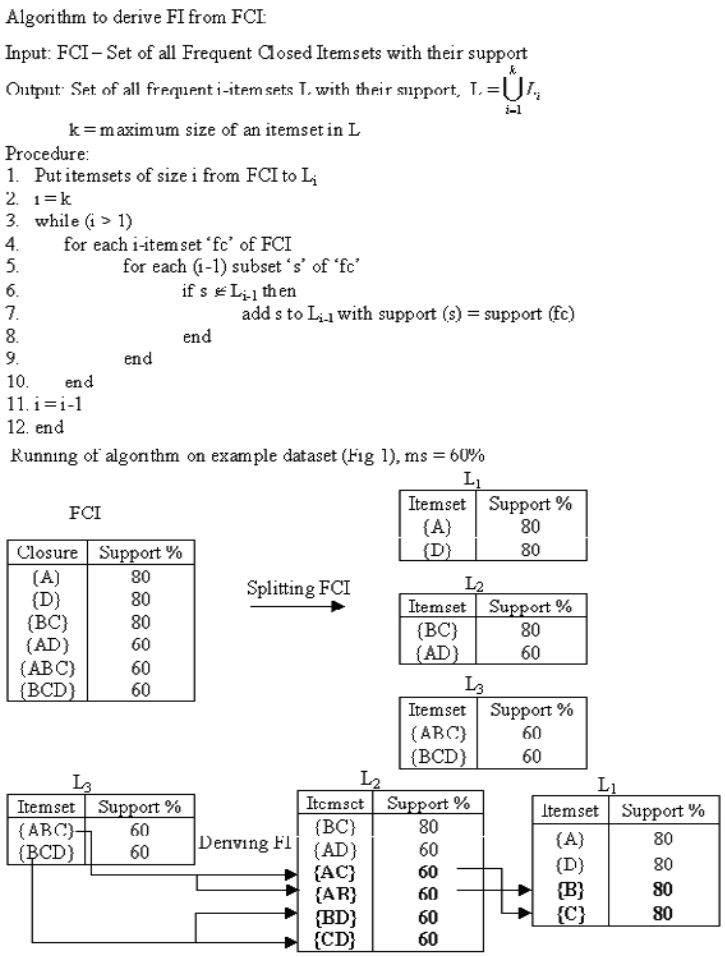
Mining FCI instead of FI has following advantages:

1. FCI represents the complete set of FI i.e. all FI can be derived from FCI without any information loss.
2. FCI are much less in number than FI so mining FCI is faster and less memory consuming.
3. FCI enables discovery of non-redundant set of association rules instead of complete set of rules discovered using FI thereby reducing the information presented to the user.

Derivation of FI from FCI

FCI represents the complete set of FI and all FI can be derived from set of FCI without any loss of information. Pasquier et. al. presented an algorithm of deriving FI from FCI (Pasquier N., Bastide Y., Taouil R. & Lakhal L., 1999a). The

Figure 2. Algorithm to dervice FI from FCI, Running of algorithm on example dataset (Figure 1)



algorithm works as follows. For a given frequent closed itemset '*fc*' of size *k*, the algorithm finds all its subset '*s*' of size (*k-1*). If any of the subset '*s*' found in earlier step is not a closed itemset then '*s*' is a frequent itemset with support same as the support of '*fc*'. Fig 2 shows the algorithm and its running example.

Number of FCI vs. FI

In real life datasets, it is often observed that number of FCI is much less than number of FI (Pei J., Han J. & Mao R., 2000). Because of this reason, mining FCI instead of FI reduces the computational expense of association rule mining. Comparison of number of FCI and number of FI generated on the example dataset 1 is shown in fig 1. Fig. 3 shows the experiments conducted by Pei et. al. on dataset *Connect-4* (Pei J., Han J. & Mao R., 2000). *Connect-4* is a real dataset having 67,557 transactions and 43 items and was taken from UC-Irvine Machine Learning Database Repository. It is a dense dataset with a large set of long FI.

Non-Redundant Association Rules

Zaki (2000) has shown that FCI can be used to generate non-redundant association rules where

Figure 3. Number of FCI and FI in dataset Connet-4 (Adapted from Pei J., Han, J., & Mao, R., 2000)

Support %	No. of FCI	No. of FI
95	812	2,205
90	3,486	27,127
80	15,107	533,975
70	35,875	4,129,839

non-redundant rules are defined as follows. Let R_i denote the rule $X_1^i \rightarrow X_2^i$, where $X_1, X_2 \subseteq I$. Rule R_1 is more general than rule R_2 provided R_2 can be generated by adding additional items to either the antecedent or consequent of R_1. Rules having the same support and confidence as more general rules are the redundant association rules. Remaining rules are non-redundant rules.

The method of generating strong association rules from set of FI that finds subsets of all FI and tests the strength generates lot of redundant rules (Han J., & Kamber M., 2006). For example, the association rules with confidence 75%, generated from the frequent itemsets {AB, AC, ABC} given in example 1 are {A→B, A→C, B→A, C→A, A→BC, B→AC, C→AB, BC→A} and the set of non-redundant rules based on FCI using the method given by Zaki are {A→B, A→C, B→A, C→A}(Zaki M. J., 2000).

Approaches for Mining FCI

Based on different definitions of FCI, different algorithms have been proposed for generating FCI.

1. **Closure property:** Charm, Charm-L, Closet algorithms
2. **Galois closure operator:** Close, A-Close, Apriori-Close algorithms
3. **Formal concept analysis:** all algorithms to compute Formal Concepts in FCA e.g. Ganter's naïve algorithm, Titanic algorithm

Algorithms Based on Closure Property

Zaki & Hsiao proposed the Charm algorithm to compute FCI (Zaki M. J. & Hsiao C. –J., 1999). It uses the pruning strategies of *subset infrequency* (an itemset is infrequent if any of its subsets is infrequent) and of *non-closure property* (if a superset has the same support as the itemset then the itemset is not closed). Initially the algorithm considers 1-itemsets along with their tidsets (list of transaction ids) and repeats the following for each level i ($i > 1$):

- Sorts the elements in increasing order of their support.
- Deletes elements that do not satisfy the pruning strategies.
- Computes children for the each element (say X_1) by combining X_1 with each element X_2, such that X_2 comes after X_1 in the ordered list.
- Computes the tidsets of the new itemsets $X_1 \cup X_2$: If tidsets of X_1 and X_2 are same then replace X_1 by $X_1 \cup X_2$ and remove X_2 from further consideration, if tidsets of X_1 (or X_2) is a subset of that of X_2 (or X_1) then replace X_1 (X_2) by $X_1 \cup X_2$, if tidsets of X_1 and X_2 are different then nothing can be eliminated.

The algorithm stops when no new children are generated. The elements remaining at each level constitute the set of FCI.

Charm-L is similar to Charm but also creates closed itemset lattice that may be used for generating association rules (Zaki M. J. & Hsiao C. –J., 2005).

Closet Algorithm (Pei J., Han J. & Mao R., 2000) computes FCI set using Frequent Pattern tree (FP-tree) data structure and conditional databases (Han J., Pei J. & Yin Y., 2000). *FP-tree* is a prefix tree structure that stores compressed information about FI for a database. It consists of a root labeled as "null" and each branch starting at root corresponds to a transaction in the database. The transactions are arranged in decreasing order of support of items. Transactions with the same prefix share the corresponding portion of the path from the root. The *X-conditional database* (where *X* is a FI) is a projected database for frequent itemset *X*. The *conditional database* is derived from the database by collecting all transactions containing *X* and removing from them (1) infrequent items, (2) all frequent items that come after *X* in the sorted FI list, and (3) *X*. Initially the algorithm extracts FI from the database and sorts them in decreasing order of their support. It builds the FP-tree for the database items, forms the *X-conditional databases* for each *X* in the sorted FI list and computes the local sorted FI lists for each of the conditional databases. For each *X-conditional database*, it finds set of items *Y* in sorted FI list such that *Y* appears in every transaction of the conditional database. It inserts $X \cup Y$ in FCI set if $X \cup Y$ is not a subset of any element of the FCI set with the same support. It repeats the steps till the sorted FI lists become empty.

ALGORITHMS BASED ON GALOIS CLOSURE OPERATOR

Pasquier et. al. proposed Close method to find Frequent Closed Itemsets (FCI) (Pasquier N., Bastide Y., Taouil R. & Lakhal L. 1999a). Close method finds the closures based on Galois closure operators. Galois closure operator $h(X)$ for some $X \subseteq I$ is defined as the intersection of transactions in *D* containing itemset *X*. An itemset *X* is a closed itemset iff $h(X) = X$. Close method is based on Apriori algorithm and computes generators at each level where generator of an itemset *X* is one of the smallest arbitrarily chosen itemset *p*, such that $h(p) = X$. It starts from 1-itemsets, and goes up level-by-level computing generators, their closures (i.e. FCI), and support at each level. At each level, candidate generator itemsets of size *k* are found by joining generator itemsets of size *k-1* using the combinatorial procedure used in Apriori algorithm. The candidate generator itemsets are pruned using two strategies i) remove candidate generator itemsets whose all subsets are not frequent ii) remove the candidate generator itemsets if closure of one of its subsets is superset of the generator. Subsequently, algorithm finds the support of pruned candidate generator itemsets. Each level requires one pass over the database to construct the set of FCI and count their support.

A-Close algorithm is a variation of Close algorithm (Pasquier N., Bastide Y., Taouil R. & Lakhal L., 1999b). The main difference as compared to Close is that A-Close generates all the generators together, prunes the infrequent ones and only then computes their closures. The algorithm scans the database to derive generators and their support in a level-wise manner. It prunes the infrequent generators and those having same closure as any of their immediate subsets. The set of FCI are obtained in a single database scan by applying the Galois operator on the frequent and useful generators.

Apriori-Close algorithm is an extension of Apriori algorithm and computes FI and FCI simultaneously (Pasquier N., Bastide Y., Taouil R. & Lakhal L., 1999c). The algorithm initializes the set of FI as the 1-itemsets. At each successive level, candidate itemsets are generated by application of combinatorial phase of Apriori. An itemset is pruned if either any of the subsets are infrequent or if the itemset itself is infrequent. The algorithm then checks the support of all the supersets of FI to find they are closed.

ALGORITHMS BASED ON FORMAL CONCEPT ANALYSIS

Stumme (1999) discovered that intent of Formal Concept in FCA is same as FCI. This implies that all algorithms to compute Formal Concepts can be used as an algorithm to compute FCI. Several researchers have analyzed performance of various algorithms for computing Formal Concepts (Kuznetsov S.O., & Obiedkov S.A. 2002) (Carpineto C. & Romano G. 2004). Among all algorithms, Ganter's naïve method is the first and simplest algorithm to compute Concepts. Later Ganter proposed Next-Closure algorithm for the same purpose. These algorithms compute all Concepts, irrespective of the support of intent. Stumme et. al. proposed Titanic algorithm, which computes only the frequent Concepts where support of intent is greater than the user specified threshold (ms) (Stumme G., Rafik T., Bastide Y., Pasquier N., & Lakhal L. 2002).

Close Algorithm is one of the first few algorithms to compute FCI. We explain the algorithm in Fig. 4 and show a running example in Fig. 5.

Figure 4. Close algorithm

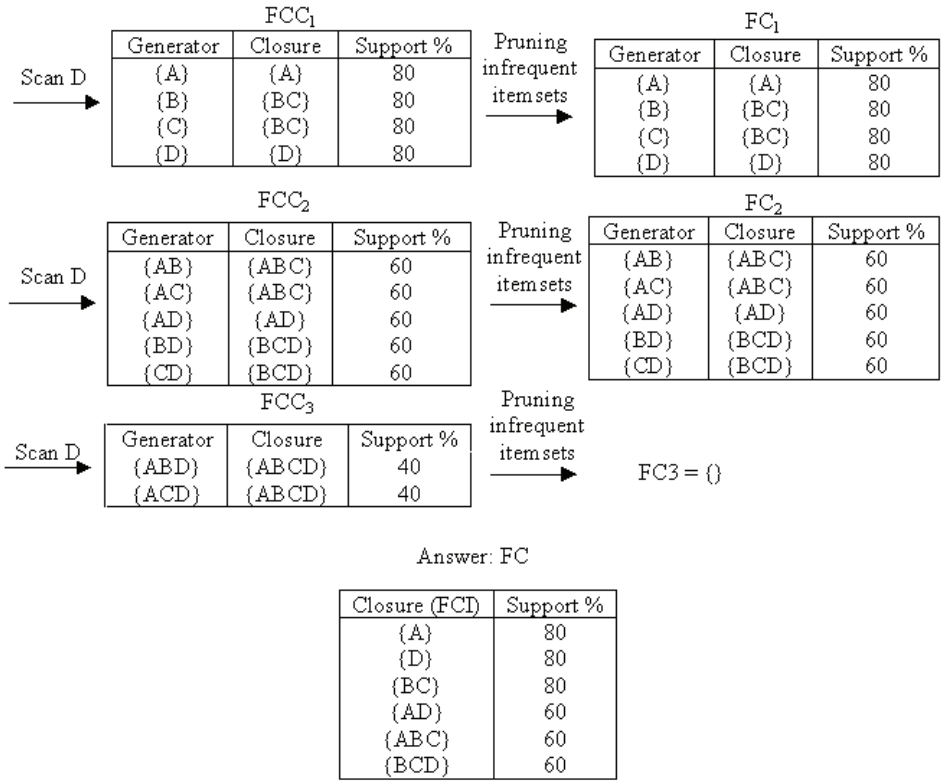
Input: Database of N transactions, ms
Output: Set FC of frequent closed itemsets
Notations: FCCi (Frequent Candidate Closed itemset of size i), FCi (Frequent Closed itemset of size i). FCCi, FCi are structures containing generator, closure and support components.

Procedure:
(1) Sort the items in lexicographic order.
(2) Initialize FCC_1.generators to 1 – itemsets.
(3) i = 1
(4) while FCC_i.generator ≠ ϕ
(5) FCC_i.closure = ϕ
(6) FCC_i.support = 0
(7) for each 'fcc' ∈ FCC_i
(8) fcc.closure = h(fcc.generator)
(9) Scan the database to get fcc.support = support (fcc.closure)
(10) if (fcc.support ≥ ms) then
(11) $FC_i = FC_i \cup$ {fcc}
(12) end
(13) end
(14) FCC_{i+1}.generator = gen_generator(FC_i)
(15) FCC_{i+1}.generator = prune_generator(FCC_{i+1})
(16) i++
(17) end
18. return $FC = \bigcup_i (FC_i.closure, FC_i.support)$

gen_generator (FC_i)
// Generates potential (i+1) generators
1. FCC_{i+1}.generator = ϕ
2. for each 'fc_1' ∈ FC_i.generator
3. for each 'fc_2' ∈ FC_i.generator
4. if ($fc_1[1] = fc_2[1]$) ∧ ($fc_1[2] = fc_2[2]$) ∧ … ∧ ($fc_1[i-1] = fc_2[i-1]$) ∧ ($fc_1[i] < fc_2[i]$) then
5. fcc = $fc_1[1]$ $fc_1[2]$ … $fc_1[i-1]$ $fc_1[i]$ $fc_2[i]$
6. end
7. FCC_{i+1}.generator = FCC_{i+1}.generator ∪ {fcc}
8. end
9. return FCC_{i+1}.generator

prune_generator (FCC_i)
1. for each 'fcc' ∈ FCC_i.generator
2. for each i-subset 's' of 'fcc'
3. if (fcc ⊆ s.closure) then
4. remove 'fcc' from FCC_i.generator
5. end
6. end
7. end
8. return FCC_i.generator

Figure 5. Discovering FCI in the dataset (Figure 1) using Close algorithm, ms= 60%



FUTURE TRENDS

Mining frequent closed itemsets (FCI) in association rule discovery promises a great improvement in terms of time and memory efforts. However mining FCI in incremental databases has not been explored much. Parallel and distributed algorithms for FCI also seem to be useful in large and distributed databases. On line mining of FCI for streaming data is another interesting direction to work on.

CONCLUSION

The article introduces frequent closed itemsets (FCI) and its importance in association rule mining. Advantages of mining FCI over FI are explained in details with suitable examples. The article elaborates on different definitions of FCI given by different researchers and then it explains briefly the algorithms for computing FCI based on those definitions. 'Close', a popular algorithm for mining FCI, is elaborated in detail and a running example is presented.

REFERENCES

Agrawal R. & Srikant R. (1994). Fast Algorithms for Mining Association Rules in Large Databases. Proceedings of the 20th International Conference on Very Large Data Bases, Santiago, Chile, 478-499.

Carpineto C. & Romano G. (2004). Concept Data Analysis: Theory and Applications. John Wiley & Sons, England.

Ganter B. & Wille R. (1999). Formal Concept Analysis: Mathematical Foundations. Springer, Heidelberg.

Han J., & Kamber M. (2006). Data Mining: Concepts and Techniques. Morgan Kaufmann Publishers.

Han J., Pei J. & Yin Y. (2000). Mining Frequent Patterns without Candidate Generation. Proceedings 2000 ACM-SIGMOD International Conference Management of Data (SIGMOD '00), Dallas, TX.

Han J., Cheng H., Xin D., & Yan X. (2007). Frequent Pattern Mining: Current Status and Future Directions. Journal of Data Mining and Knowledge Discovery, 15:55-86.

Kuznetsov S. O., & Obiedkov S.A. (2002). Comparing Performance of Algorithms for Generating Concept Lattices. Journal of Experimentation and Theoretical Artificial Intelligence.

Pasquier N., Bastide Y., Taouil R. & Lakhal L. (1999a). Efficient Mining of Association Rules using Closed Itemset Lattices. Journal of Information Systems, 24(1), 25-46.

Pasquier N., Bastide Y., Taouil R. & Lakhal L. (1999b). Discovering Frequent Closed Itemsets for Association Rules. Proceedings of International Conference on Database Theory (ICDT).

Pasquier N., Bastide Y., Taouil R. & Lakhal L. (1999c). Closed Set based Discovery of Small Covers for Association Rules. Proceedings of BDA conference, 361-381.

Pei J., Han J. & Mao R., (2000). Closet: An Efficient Algorithm for Mining Frequent Closed Itemsets. Proceedings of the ACM-SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery, 21-30.

Stumme, G., (1999), Conceptual Knowledge Discovery with Frequent Concept Lattices, FB4-Preprint 2043, TU Darmstadt

Stumme G., Rafik T., Bastide Y., Pasquier N., & Lakhal L. (2002). Computing Iceberg Concept Lattices with Titanic. Journal on Knowledge and Data Engineering, 42(2), 189-222.

Zaki M. J. (2000). Generating Non-Redundant Association Rules. 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 34-43.

Zaki M. J. & Hsiao C.–J. (1999). Charm: An Efficient Algorithm for Closed Association Rule Mining. Computer Science Dept., Rensselaer Polytechnic Institute. Technical Report 99–10.

Zaki M. J. & Hsiao C.–J. (2005). Efficient Algorithms for Mining Closed Itemsets and their Lattice Structure. IEEE Transactions on Knowledge and Data Engineering, 17(4), 462-478.

KEY TERMS

Association Rule: An Association rule is an implication of the form $X \rightarrow Y$ where $X \subset I$, $Y \subset I$ and $X \cap Y = \emptyset$, I denotes the set of items.

Closure: For any itemset $C \subseteq I$, $h(C)$ is the closure of C where $h = f \, o \, g$ is the Galois Closure Operator.

Data Mining: Extraction of interesting, non-trivial, implicit, previously unknown and potentially useful information or patterns from data in large databases

Frequent Closed Itemset (FCI): An itemset X is a closed itemset if there exists no itemset X' such that

i. X' is a proper superset of X,

ii. Every transaction containing X also contains X'.

A closed itemset X is frequent if its support exceeds the given support threshold.

Frequent Itemset (FI): A set of one or more items belonging to set I is termed as an itemset. An itemset X is a frequent itemset if support of X is greater than the user specified support threshold (ms).

Formal Concept: A formal context $K = (G,M,I)$ consists of two sets G (objects) and M (attributes) and a relation I between G and M. For a set $A \subseteq G$ of objects

$A' = \{m \varepsilon M \mid gIm \text{ for all } g \varepsilon A\}$ (the set of all attributes common to the objects in A). Correspondingly, for a set B of attributes we define

$B' = \{g \varepsilon G \mid gIm \text{ for all } m \varepsilon B\}$ (the set of objects common to the attributes in B).

A formal concept of the context (G,M,I) is a pair (A,B) with $A \subseteq G, B \subseteq M$, such that

$A' = B$ and $B' = A$

A is called the extent and B is the intent of the Formal Concept (A,B).

Formal Concept Analysis (FCA): It is a branch of mathematics based on Concept and Concept Hierarchies.

Generator Itemset: A generator p of a closed itemset c is one of the smallest itemsets such that $h(p) = c$.

Galois Closure Operator: Let $D = (O,I,R)$ be a data mining context where O and I are finite sets of objects (transactions) and items respectively. P $\subseteq$ O x I is a binary relation between objects and items. For $O \subseteq O$, and $I \subseteq$ I, we define:

$f(O)$: $2^O \rightarrow 2^I$ $\quad$ $g(I)$: $2^I \rightarrow 2^O$

$f(O) = (i \in I \mid \forall o \in O, (o,i) \in R\}$ $\quad$ $g(I) = (o \in O \mid \forall i \in I, (o,i) \in R\}$

$f(O)$ associates with O the items common to all objects $o \in O$ and $g(I)$ associates with I the objects related to all items $i \in I$. The operators $h = f \circ g$ in 2^I and $h' = g \circ f$ in 2^O are Galois closure operators. An itemset $C \subseteq I$ from D is a closed itemset iff $h(C) = C$.

Non-Redundant Association Rules: Let R_i denote the rule $X_1^i \rightarrow X_2^i$, where $X_1, X_2 \subseteq I$. Rule R_1 is more general than rule R_2 provided R_2 can be generated by adding additional items to either the antecedent or consequent of R_1. Rules having the same support and confidence as more general rules are the redundant association rules. Remaining rules are non-redundant rules.

Chapter LVIII
Similarity Retrieval and Cluster Analysis Using R* Trees

Jiaxiong Pi
University of Nebraska at Omaha, USA

Yong Shi
University of Nebraska at Omaha, USA
Graduate University of the Chinese Academy of Sciences, China

Zhengxin Chen
University of Nebraska at Omaha, USA

INTRODUCTION

Data mining is aimed at the extraction of interesting (i.e., nontrivial, implicit, previously unknown, and potentially useful) patterns or knowledge from huge amounts of data. In order to make data mining manageable, data mining has to be database centered. Yet, data mining goes beyond the traditional realm of database techniques; in particular, reasoning methods developed from machine learning techniques and other fields in artificial intelligence (AI) have made important contributions in data mining. Data mining thus offers an excellent opportunity to explore the interesting fundamental issue of the relationship between data and knowledge retrieval and inference and reasoning. Decades ago, researchers made an important remark stating that since knowledge retrieval must respect the semantics of the representation language, knowledge retrieval is a limited form of inference operating on the stored facts (Frisch & Allen, 1982). The inverse side of this statement has also been explored, which views inference as an extension of retrieval. For example, Chen (1996) described a computer model that is able to generate suggestions through document structure mapping based on the notion of reasoning as extended knowledge retrieval; the model was implemented using a relational approach. However, although the issue of foundations of data mining has attracted much attention among data mining researchers (ICDM, 2004), little work has been done on the important relationship between retrieval and inference (or mining). A possible reason of lacking such kind of research is the difficulty of identifying an appropriate common ground that can be used to examine both data retrieval and data mining.

On the other hand, from the database perspective, an effective way to achieve efficient

data mining is by exploiting important features of database primitives. For example, as a multi-dimensional index structure for spatial data, R* tree (Beckmann, Kriegel, Schneider, & Seeger, 1990; Gaede & Günther**,** 1998) is a powerful database primitive and is widely used in database applications. A rich literature exists in regard to the application of R* trees for data mining as well (e.g., Keogh, Chakrabarti, Pazzani, & Mehrotra, 2000). Since the R* tree was originally developed for spatial data retrieval, recent developments in using R* tree for data mining reveals that R* tree structure can serve as a common ground to explore the relationship between retrieval and mining as discussed above.

As our first step to explore this interesting issue, in this article we examine time-series data indexed through R* trees, and study the issues of (a) retrieval of data similar to a given query (which is a plain data retrieval task), and (b) clustering of the data based on similarity (which is a data mining task). Along the way of examination of our central theme, we also report new algorithms and new results related to these two issues. We have developed a software package consisting of components to handle these two tasks. We describe both parts of our work, with an emphasis on dealing with the challenges of moving from retrieving individual queries similar to a given query to clustering the entire data set based on similarity. Various experimental results (omitted due to space limitation) have shown the effectiveness of our approaches.

BACKGROUND

Just like a B Tree, an R Tree (Guttman, 1984) relies on a balanced hierarchical structure, in which each tree node is mapped to a disk page. However, whereas B Trees are built on single-value keys and rely on a total order on these keys, R Trees organize rectangles according to a containment relationship. Each object to be indexed will be represented by a minimum bounding box (MBB) in the index structure except the point for which an MBB simply degrades to a point. All indexed objects will eventually be put in leaf nodes. A leaf node contains an array of leaf entries. A leaf entry is a pair (*mbb, oid*), where *mbb* is the MBB and *oid* is the object ID. Each internal node is associated with a rectangle, referred to as the directory rectangle (*dr*), which is the minimal bounding box of the rectangle of its child nodes. The structure of R Tree satisfies the following properties.

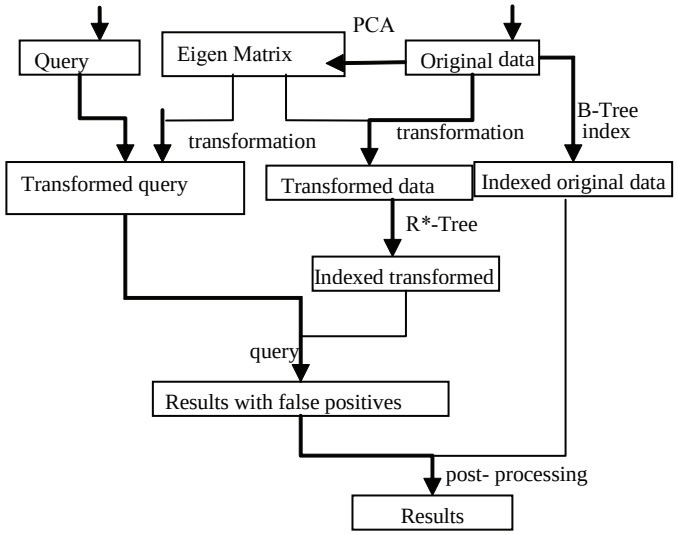
- For all nodes in the tree (except for the root), the number of entries is between *m* and *M*, where $0 \leq m\ M/2$.
- For each entry (*dr, node-id*) in a nonleaf node *N*, *dr* is the directory rectangle of a child node of *N*, whose page address is *node-id*.
- For each leaf entry (*mbb, oid*), *mbb* is the minimal bounding box of the spatial component of the object stored at address oid.
- The root has at least two entries (unless it is a leaf).
- All leaves are at the same level.

R* Tree is a variant of the R Tree that provides several improvements to the insertion algorithm. Among other things, R* tree reinserts entries upon overflow, rather than splitting. See Beckmann et al. (1990), and Gaede and Günther (1998) for more detail.

R* TREE FOR SIMILARITY ANALYSIS

As shown in Agrawal, Faloutsos, and Swami (1993), when R* tree is used for time-series data indexing, each time series of length *n* is mapped to a point in *n*-dimension space. Thus, a similarity query problem can be converted to finding those points close to a given point. The whole data set is indexed through an R* tree, and a similarity query is then carried out on the R* tree. Since R* tree indexes spatial objects according to spatial proximity and close points tend to be put in the same leaf node, a small amount of leaf nodes will be traversed before similar points are found. As a result, fast similarity analysis can be achieved. However, due

Figure 1. Data flow diagram in the similarity analysis tool when PCA module is used



to the so-called "dimensionality curse," R* tree's performance degrades rapidly when the dimension is larger than 10. Provided that some data sets and databases are made up of time series with long lengths (>>10), and for R* tree to work efficiently, a dimensionality reduction technique is necessary. Over the years, various dimensionality reduction techniques have been proposed; for example, Agrawal et al. adopted discrete Fourier transform (DFT) as a dimensionality reduction method, and Keogh et al. (2000) proposed a simple data transformation technique, piecewise aggregation approximation (PAA). However, although principal component analysis (PCA) is a well-known method for dimensionality reduction, its application for time-series analysis has not been reported.

We have explored using PCA for dimensionality reduction for time-series data and developed a tool for similarity analysis using PCA and other methods. The following are general steps for performing PCA.

Step 1: Construct a covariance matrix of column vectors.
Step 2: Calculate eigenvectors.
Step 3: Determine principle components and perform dimensionality reduction.

Compared with DFT and PAA, PCA has the following virtues:

- PCA is an orthogonal transformation and can guarantee distance conversation if all eigenvectors are used.
- PCA operates on the whole data set and can capture the primary features such as data distribution and variation of the data set. After dimensionality reduction, those primary features can be maintained.

We have developed a similarity analysis tool that is made up of three modules, namely, the R* tree module, PCA module, and B tree module. In this similarity analysis tool, starting from a data set and a query and ending with similarity results found, data go through phases as shown in Figure 1.

R* TREES FOR CLUSTER ANALYSIS

Similarity querying on R* trees takes advantage of R* tree's feature of grouping objects together based on spatial proximity. A natural extension of this study would be to push the task of retrieval a step further: to the task of clustering, where the entire data set is clustered into groups based on similarity

so that the commonality of the data in the same cluster can be revealed. However, using R* tree for cluster analysis should be done cautiously as researchers already issued the following warning a decade ago (Ester, Kriegel, & Xu, 1995).

- IF R* tree is used for clustering, then all clusters (i.e., the directory rectangles) have a rectangular shape and these rectangles have to be parallel to the axes of the coordinate system. This restriction does not comply with the objective of cluster analysis.
- An R* tree is a balanced tree. For all nodes in the tree (except for the root), the number of entries is between *m* and *M*, where *M* is node capacity, and *m* is set to a constant in [0, *M*/2]. Assuming that each node is a cluster, the number of members in every cluster is then between *m* and *M*. In reality, however, no bound is set for the number of cluster members.
- The R* tree structure does not allow users to specify the number of clusters; it derives the number of cluster, *k*, indirectly from *n* and from the capacity of a page. This *k* may be inappropriate for a given application and may yield clusterings with a high total distance.

Although there is a mismatch of objectives and behavior between R* tree structure and cluster analysis, the attractive indexing characteristics of R* trees still lends itself for great potential of contribution to clustering tasks. Below we discuss issues related to this aspect by examining how to take advantage of R* tree's indexing feature to get around the problems mentioned above, and present two improved cluster analysis algorithms by incorporating R* tree features.

K-Means Extended with R* Tree

Among the partitioning algorithms, *k*-means is a popular and widely studied clustering method for points in Euclidean space. The algorithm of this method is presented below, where *IC* is the collection of initial centroids of *k* clusters. *S* is the collection of centroids of leaf nodes. *K*Means(*IC*, *Data*, *k*) refers to the algorithm of clustering *Data* into *k* clusters through *k*-means with initial centroids of *IC*, and *k* clusters returned as the result.

KMeans(*IC, Data, k*)

1. *IC*= Φ
2. Pick randomly from *Data k* data items, c_i i=1 ,…, *k*, $IC = IC \cup c_i$
3. Initialize *k* clusters, Cluster(i), i=1, …, *k*, with centroids of c_i respectively.
4. For each data item d_i, determine its nearest centroids c_j, then set $Cluster(j) = Cluster(j) \cup c_j$
5. Set *IC*= Φ
6. Calculate the new centroids, nc_i, i=1, …, *k*, of each cluster. Set $IC = IC \cup nc_i$
7. Go to 4 until no variation in clustering results.

The *k*-means algorithm is simple and fast, and can handle high dimensionality relatively well. However, *k*-means and its variations have a number of limitations. For example, it has difficulty detecting the "natural" clusters, it suffers from the problem of local minima, it requires provision of the total number of clusters, *k*, from the user, its behavior is significantly affected by the initial selection of centroids, and it has a problem when outliers exist. The issue of initial selection of centroids is partially addressed by a variant (referred to as KMeans-S here) where sampling techniques are used to determine optimized centroids in which sampled points are selected and clustered first to determine initial centroids for clustering the whole data set.

KMeans-*S*(*Data, k*)

1. Sample in *Data* and generate a subset of *Data*, *S*
2. *IC*= Φ
3. Apply *K*-Means(*IC*, *S*, *k*) and obtain new *IC*

4. Apply *K*-Means(*IC*, *Data*, *k*)

Using R* trees cannot take care all of the shortcomings of *K*Means or *K*Means-S algorithms. However, it is reasonable to expect at least R* trees can be used to assist better sampling. Note that when *K*Means-S is used, the sampled points are not guaranteed to represent the whole data set well. In our view, when data points are indexed through an R* tree, if we select one point from each leaf node, these selected data points collectively should represent the data in a way better than random sampling. Consequently, the clustering result could be improved. Pushing this observation a step further, we take centroids of data in leaf nodes as a sample instead of using random points, one from each leaf node. Those sampled centroids are then used as initial centroids for clustering. Note that in *K*Means-S, some sampled points may not represent well the distribution of data points; but in our proposed approach, the sampled points should be relatively good representatives of the data set because the collection of leaf nodes is a partition of a data set and can reflect the distribution of the data set. Shown below is our revised algorithm, *K*Means-R.

Algorithm KMeans-*R*(*Data, k*)

1. *IC*=Φ, *S*= Φ
2. Index *Data* by R* tree
3. For each leaf Ni of R* tree, obtain the centroid of its data elements, c_i, then set $S = S \cup c_i$
4. Apply *K*-Means(*IC*, *S*, *k*), obtain new centroids ICN of newly formed clusters, then set *IC*=ICN
5. Apply *K*-Means(*IC*, *Data*, *k*) return the clustering results.

Comparing the three clustering algorithms, since *K*Means-S uses the centroids of sampled data sets as initial centroids instead of randomly picked centroids as done in *k*-means, centroids should capture the spatial distribution of data better than original *K*Means. *K*Means-R further improves initial centroids by using the centroids of leaf nodes. Since the quality of *k*-means clustering can be affected by the selection of initial centroids, *K*Means-R should perform better than *K*Means-S and *k*-means.

Hierarchical Clustering Extended with R* Tree

Unlike the *k*-means method for which a user needs to specify the number of clusters beforehand, hierarchical clustering gives a series of clustering results at each level through the merging process.

For comparison purpose, the basic algorithm of the hierarchical clustering is shown below, where *Data* is the input data with size of *n*. Note that *distMatrix* is a matrix of distance between any two clusters.

Algorithm Hierarchy (*Data*)

1. Assign each point to a cluster, and generate *n* clusters, say, *Cluster*(1), *Cluster*(2), …, *Cluster*(*n*).
2. Start off the merging process as follows.
   2.1. Calculate and form an initial *n*×*n* all-pair distance matrix (*distMatrix*(*n*, *n*))
   2.2. Based on distance matrix, identify a pair of nearest clusters, say, *Cluster*(*i*) and *Cluster*(*j*), then merge them. Set *Cluster*(*i*) = *merge*(*Cluster*(*i*), *Cluster*(*j*)).
3. Recalculate distance matrix
   3.1. Assume *Cluster*(*s*) is the last one in the sequence of clusters; set *Cluster*(*j*) = *Cluster*(*s*) and then delete *Cluster*(*s*)
   3.2. Recalculate the distance of *Cluster*(*i*) and *Cluster*(*j*) to other remaining clusters
   3.3. Based on the above calculation, form an (*s*-1)×(*s*-1) distance matrix *distMatrix*(*s*-1,*s*-1)
4. Go to 2.2 until number of clusters is reduced to 1

If clustering starts off from individual points as done in the original hierarchical clustering method, the number of start-up clusters will be large and thus clustering will be temporal and spatially expensive (O (m^2), where *m* is the total number of objects). R* trees can come for help. Although a leaf node in R* tree does not necessarily represent a cluster, it is reasonable to hypothesize that when the node capacity is low for leaf nodes, the points in an R* tree's leaf node are likely belonging to same cluster. Therefore, rather than start the clustering process from individual points, we can first index those points and then cluster those minimal bounding boxes of leaf nodes. The new algorithm is presented below. Note that the value of *m* cannot be big (otherwise the points in a leaf node could belong to two or more clusters) nor too small (otherwise the algorithm degrades to hierarchical clustering, m=1).

Algorithm Hierarchy-R(*Data*)

1. Index *Data* through R* Tree and generate *m* leaf nodes.
2. Assign the points in each leaf node to a cluster, and generate *m* initial clusters, *Cluster*(1), *Cluster*(2), …, *Cluster*(*m*).
3. Merge clusters as done in hierarchical clustering.

FUTURE TRENDS

As an attractive database primitive, R* trees will continue to play an important role in similarity retrieval and cluster analysis. However, the R* tree is only one of many spatial data structures. Alternative approaches, such as KD trees (Bentley, 1975) and octrees have been proposed (for an experimental study of octrees, see Aronov et al., 2003). Other spatial data structures dealing with high dimensionality includes TV tree (Lin, Jagadish, & Faloutsos, 1994), X-tree (Berchtold, Keim, & Kriegel, 1996), and SR tree (Katayama & Satoh, 1997). In addition, the trend of studying metric indexes for similarity search will continue. Indexing a metric space requires provision of an efficient support for answering similarity queries, such as using M tree (Ciaccia, Patella, & Zezula, 1997) or other metric trees, where a metric function is used to measure the distance between objects, and only considers relative distances of objects to organize and partition the search space.

CONCLUSION

As our work shows, extending similarity retrieval to clustering is not a straightforward process. The general lesson we learned from this study is that since this relationship is a complex issue, it should be studied on a case-by-case basis; for example, in our study, we have exploited features of R* trees. However, this is not to say that there is no commonality on different cases of relationship between retrieval and reasoning. Rather, finding such a commonality is a challenging task that deserves much effort.

REFERENCES

Agrawal, R., Faloutsos, C., & Swami, A. (1993). Efficient similarity search in sequence databases. *Proceedings of the 4th Conference on Foundations of Data Organization and Algorithms* (pp. 69-84).

Aronov, B., Bronnimann, H., Chang, A. Y., & Chiang, Y.-J. (2003). Cost-driven octree construction schemes: An experimental study. *Proceedings of the Nineteenth Annual Symposium on Computational Geometry* (pp. 227-236).

Beckmann, N., Kriegel, H. P., Schneider, R., & Seeger, B. (1990). The R*-tree: An efficient and robust access method for points and rectangles. *Proceedings of SIGMOD Conference* (pp. 322-331).

Bentley, J. L. (1975). Multidimensional binary search trees used for associative searching. *Communications of the ACM, 18*(9), 509-517.

Berchtold, S., Keim, D. A., & Kriegel, H.-P. (1996). The X-tree: An index structure for high-dimensional data. *Proceedings of 22nd VLDM Conference* (pp. 28-39).

Chen, Z. (1996). Generating suggestions through document structure mapping. *Decision Support Systems, 16*(4), 297-314.

Ciaccia, P., Patella, M., & Zezula, P. (1997). M-tree: An efficient access method for similarity search in metric spaces. *VLDB Journal*, 426-435.

Ester, M., Kriegel, H. P., & Xu, X. (1995). Knowledge discovery in large spatial databases: Focusing techniques for efficient class identification. In *Lecture notes in computer science: Proceedings of 4th International Symposium on Large Spatial Databases (SSD'95)* (pp. 67-82). Springer.

Frisch, A. M., & Allen, J. F. (1982). Knowledge retrieval as limited inference. *Proceedings of the 6th Conference on Automated Deduction* (pp. 274-291).

Gaede, V., & Günther, O. (1998). Multidimensional access methods. *ACM Computing Surveys, 30*(2), 170-231.

Guttman, A. (1984). R-trees: A dynamic index structure for spatial searching. *Proceedings of ACM SIGMOD International Conference on Management of Data* (pp. 47-54).

ICDM. (2004). *Foundation of Data Mining Workshop call for papers.* Retrieved from http://biomig.csie.cgu.edu.tw/meeting/2004.07.25.htm

Katayama, N., & Satoh, S. (1997). The SR-tree: An index structure for high-dimensional nearest neighbor queries. *Proceedings of 1997 ACM SIGMOD* (pp. 369-380).

Keogh, E., Chakrabarti, K., Pazzani, M., & Mehrotra, S. (2000). Dimensionality reduction for fast similarity search in large time series databases. *Knowledge and Information Systems, 3*(3), 263-286.

Lin, K.-I., Jagadish, H. V., & Faloutsos, C. (1994). The TV-tree: An index structure for high-dimensional data. *VLDB Journal 3*(4), 517-542.

KEY TERMS

Cluster Analysis: The concept of cluster refers to a collection of data objects where data objects within the same cluster are similar to one another while dissimilar to the objects in other clusters. Cluster analysis is aimed at finding similarities between data according to the characteristics found in the data and grouping similar data objects into clusters.

Data Mining: As a key step in the knowledge discovery from data (KDD) process, it is intended to extract interesting (nontrivial, implicit, previously unknown, and potentially useful) patterns.

Inference: The act or process of deriving a conclusion from stored data or known facts. While cognitive psychology studies human inference, automated inference algorithms have been studied in artificial intelligence and in its subfield machine learning.

Principal Component Analysis (PCA): PCA performs orthogonal linear transformation, which transforms the data to a new coordinate system to reduce multidimensional data sets to lower dimensions for analysis.

Retrieval: The act of finding previously stored data in an information system to answer a query (which represents a user's information need). Retrieval can be done on structured data (such as in database management systems), unstructured data (such as in information retrieval), or multimedia data (such as image retrieval or music retrieval).

R* Trees: R* tree is a variant of R tree for indexing spatial information. Minimization of both coverage and overlap is crucial to the performance of R trees. The R* tree attempts to reduce

both, using a combination of a revised node-split algorithm and the concept of forced reinsertion at node overflow.

Similarity Retrieval: Unlike conventional data retrieval, similarity retrieval is aimed so that, given a particular query, data objects sharing certain commonality with the given query (rather than identical with the given query) will be retrieved. Similarity retrieval is widely used in information retrieval and multimedia retrieval (such as retrieval of images similar to a given image).

Spatial Index: Spatial indexes are used by databases involving spatial data to optimize spatial queries. Indexes used by nonspatial databases cannot effectively handle features such as how far two points differ and whether points fall within a spatial area of interest.

Chapter LIX
Outlying Subspace Detection for High-Dimensional Data

Ji Zhang
CSIRO Tasmanian ICT Centre, Australia

Qigang Gao
Dalhousie University, Canada

Hai Wang
Saint Mary's University, Canada

INTRODUCTION

Knowledge discovery in databases, commonly referred to as data mining, has attracted enormous research efforts from different domains such as databases, statistics, artificial intelligence, data visualization, and so forth in the past decade. Most of the research work in data mining such as clustering, association rules mining, and classification focus on discovering large patterns from databases (Ramaswamy, Rastogi, & Shim, 2000). Yet, it is also important to explore the small patterns in databases that carry valuable information about the interesting abnormalities. Outlier detection is a research problem in small-pattern mining in databases. It aims at finding a specific number of objects that are considerably dissimilar, exceptional, and inconsistent with respect to the majority records in an input database. Numerous research work in outlier detection has been proposed such as the distribution-based methods (Barnett & Lewis, 1994; Hawkins, 1980), the distance-based methods (Angiulli & Pizzuti, 2002; Knorr & Ng, 1998, 1999; Ramaswamy et al.; Wang, Zhang, & Wang, 2005), the density-based methods (Breuning, Kriegel, Ng, & Sander, 2000; Jin, Tung, & Han, 2001; Tang, Chen, Fu, & Cheung, 2002), and the clustering-based methods (Agrawal, Gehrke, Gunopulos, & Raghavan, 1998; Ester, Kriegel, Sander, & Xu, 1996; Hinneburg & Keim, 1998; Ng & Han, 1994; Sheikholeslami, Chatterjee, & Zhang, 1999; J. Zhang, Hsu, & Lee, 2005; T. Zhang, Ramakrishnan, & Livny, 1996).

One important characteristic of outliers in high-dimensional data sets is that they are usually embedded in lower dimensional feature subspaces, and different data points may be considered as outliers in rather different subspaces. To better demonstrate the motivation of exploring outlying

Figure 1. Two-dimensional views of a high-dimensional data space

subspaces, let us consider the example in Figure 1, in which three two-dimensional views of a high-dimensional data space are presented. Note that point *p* exhibits different outlier qualities in these three views. In the leftmost view, *p* is clearly an outlier. However, in the middle view, *p* has a much weaker outlier status and is not an outlier at all in the rightmost view.

The conventional methods of outlier mining, as mentioned above, are mainly designed to detect a certain number of top outliers in a prespecified feature subspace. Consequently, this may render them to miss many outliers hidden in other feature subspaces. It would be computationally prohibitive for them to perform outlier mining in each possible subspace of a high-dimensional feature space. Thus, identifying the subspaces in which each data point is considered as an outlier would be crucial to outlier detection in high-dimensional databases.

This entry focuses on the problem of outlying subspace detection for high-dimensional data. This challenging problem has recently been identified as a subdomain of outlier mining in databases (J. Zhang, Lou, Ling, & Wang, 2004; J. Zhang & Wang, 2006; Zhu, Kitagawa & Faloutsos, 2005). Outlier mining can benefit from outlying subspace detection in the following aspects.

- Outlying subspace detection can enable outlier mining to be performed more accurately. It can allow outliers to be detected from more than one subspace. This is important to many applications. There are abundant examples of high-dimensional data such as spatial data and gene expression (microarray) data. Outlier detection from these data sets can discover potentially useful abnormal patterns. As we have mentioned earlier, outliers in these high-dimensional data sets are usually hidden in some low-dimensional subspaces. The conventional outlier detection methods may not be able to provide satisfactory effectiveness to theses applications. This is because they usually rely on a prespecified feature subspace to detect outliers, which may cause them to miss many potential outliers hidden in other feature subspaces and fail to raise necessary alarms. Therefore, outlying subspace analysis plays a crucial role for outlier mining in these applications to identify the correct feature subspaces in which outliers can be accurately mined. An object is considered as an outlier in these applications if it exhibits abnormality in one or more subspaces.
- Outlying subspace detection can help outlier detection methods to mine outliers in high-dimensional spaces more efficiently. Outlying subspace detection is an important intermediate step in example-based outlier mining in a high-dimensional feature space. Zhu et al. (2005) and Zhu, Kitagawa, Papadimitriou, and Faloutsos (2004) proposed a method to detect outliers based on a set of outlier examples. The basic idea of this method is to find the outlying subspaces of these outlier examples, from which more outliers that have similar outlier characteristics to the given examples can thereby be found in a more efficient manner. This is because outlier detection only needs to be performed in those detected outlying subspaces and the computation cost in other irrelevant subspaces can thus be saved, which leads to a substantial performance gain.
- Outlying subspace detection can contribute to a better characterization of the outliers detected. The characterization of outliers mainly involves presenting the subspaces in which these outliers exist. In a high-dimen-

sional space, it is important to not only mine outliers, but also find the context in which these outliers exist. In conventional outlier detection methods, each detected outlier can only be characterized by the subspace chosen by users that is used for outlier detection. This is not desirable as different outliers may be characterized by different outlying subspaces. Outlying subspace detection makes it possible to give a more informative characterization of the outliers we detect by finding their outlying subspaces.

BACKGROUND

The major task of outlying subspace detection is to find the subspaces (subsets of features) in which each data point exhibits significant deviation from the rest of the population. An outlying subspace for a data point is a subspace in which this data point can be considered an outlier. The problem of outlying subspace detection can be formulated as follows: Given a data point, find the subspaces in which this point is considerably dissimilar, exceptional, or inconsistent with respect to the remaining population in the database (J. Zhang & Wang, 2006; Zhu et al., 2005). The unique feature of outlying subspace detection is that, instead of detecting outliers in specific subspaces as in the classical outlier detection methods, it involves searching from the space lattice for the subspaces in which the given data point exhibits abnormal deviations.

One important step in outlying subspace detection is to devise the metric for measuring the outlier characteristics of each data point in different subspaces and formulate the definition of outlying subspaces.

J. Zhang and Wang (2006) and J. Zhang, Lou, Ling, & Wang (2005) proposed the outlying metric called outlying degree (OD), which is defined as the sum of the distances between a point and its k nearest neighbors in a subspace. The OD of a point p in space s is computed as follows:

$$OD_s(s, p) = \sum_{i=1}^{k} Dist(p, p_i) \mid p_i \in KNNSet(p, s)$$

where $KNNSet(p, s)$ denotes the set of the k nearest neighbors of p in s. OD is applicable to both numeric and nominal data: For numeric data we use Euclidean distance, while for nominal data we use the simple match method. Mathematically, the Euclidean distance between two numeric points p_1 and p_2 is defined as

$$Dist(p_1, p_2) = \sqrt{\sum \left(\frac{p_{1i} - p_{2i}}{Max_i - Min_i}\right)^2},$$

where Max_i and Min_i denote the maximum and minimum data values in the i^{th} dimension. The simple match method measures the distance between two nominal points p_1 and p_2 as $Dist(p_1, p_2) = \Sigma|p_{1i} - p_{2i}|/t$, where $|p_{1i} - p_{2i}|$ is 0 if p_{1i} equals to p_{2i} and is 1 otherwise; t is the total number of attributes.

A distance threshold T is utilized to decide whether or not a data point deviates significantly from its neighboring points in a subspace. A subspace s is regarded as an outlying subspace of data point p if $OD_s(s, p) \geq T$.

Zhu et al. (2005) proposed a different measurement of the outlier quality of data points in different subspaces based on the sparsity coefficient (Aggarwal & Yu, 2001) of the hypercube in the subspace to which the data point belongs. The computation of the sparsity coefficient involves a grid discretization of the data space, with an assumption of normal distribution for the data in each cell of the hypercube. Each attribute of the data is divided into r equidepth ranges. In each range, there is a fraction $f = 1/r$ of the data. Then, a k-dimensional hypercube is made of ranges from k different dimensions. Let N be the number of data points in the data set and $n(C)$ denote the number of objects in a k-dimensional hypercube C. Under the condition that attributes were independent statistically, the sparsity coefficient $S(C)$ of the hypercube C is defined as

$$S(C) = \frac{n(C) - N \cdot f^k}{\sqrt{N \cdot f^k (1 - f^k)}}.$$

The outlier quality of a point p in subspace s is equal to the value of the sparsity coefficient of the hypercube C in s, which p falls into, and a specified number of subspaces having the largest sparsity coefficient are considered as outlying subspaces of p.

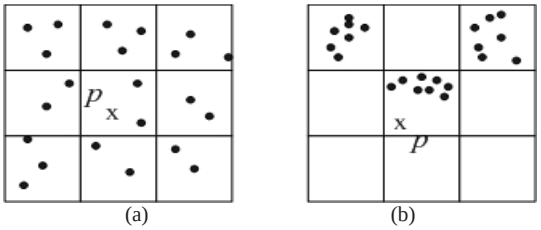
SEARCHING TECHNIQUES FOR OUTLYING SUBSPACE DETECTION

Finding the outlying subspaces for data points is essentially a search problem in high-dimensional feature spaces. Unfortunately, due to the exponential growth of the number of subspaces with respect to the dimension of the data set, the problem of outlying subspace detection is NP by nature. The straightforward exhaustive search is apparently infeasible for this problem, especially for high-dimensional data sets. In response to the inherent difficulty of this problem, state-of-the-art methods were proposed to employ heuristics in speeding up the search process in order to render this problem tractable.

An outlying subspace search algorithm, called HighDoD, is proposed that utilizes a sample-based learning process to efficiently identify the outlying subspaces for the given points (J. Zhang & Wang, 2006; J. Zhang et al., 2004). Two heuristic pruning strategies employing the upward and downward closure property are used to reduce search space:

1. **Downward Pruning Strategy:** *If a point p is not an outlier in a subspace s, then it cannot be an outlier in any subspace that is a subset of s.*
2. **Upward Pruning Strategy:** *If a point p is an outlier in a subspace s, then it will be an outlier in any subspace that is a superset of s.*

Figure 2. A data point with different outlier qualities in two two-dimensional subspaces



HighDoD performs a search in the space lattice for the outlying subspace of the given data point, and the two properties given above can be used to quickly detect the subspaces in which the point is not an outlier or the subspaces in which the point is an outlier. All these subspaces can be removed from further consideration in the later stage of the search process. Instead of taking a top-down or bottom-up search scenario, HighDoD adopts a dynamic search of the space lattice. To do this, a fast, dynamic subspace search algorithm is proposed. The total saving factor (TSF) of each layer of subspaces in the lattice, used to measure the potential advantage in saving computation, is computed in order to select the best layer of space lattice for exploration in each step of the algorithm. TSF is dynamically updated and the search is performed in the layer of lattice that has the highest TSF value in each step. The technique of random sampling is used to speed up the algorithm.

Alternatively, a genetic algorithm is used to quickly find the hidden subspace in which outlier examples stand out greatly (Zhu et al., 2005). The genetic algorithm is a promising approach that is able to, with a reasonable overhead, explore the space lattice for those subspaces that optimize the outlying fitness function. The fitness function is the negative average of all examples' sparsity coefficients, which encourages the algorithm to converge to those subspaces with high sparsity coefficient values with respect to the given outlier examples.

LIMITATIONS OF EXISTING METHODS

The major drawbacks of HighDoD lie in the binary fashion of its result and the difficulty in specifying the distance threshold. Specifically, HighDoD labels each subspace in a binary manner—either an outlying subspace or a nonoutlying one with respect to the given point—and most subspaces are pruned away before their outlying degrees are virtually evaluated in HighDoD. Thus, HighDoD is not able to return a ranked list of the detected outlying subspaces. Apparently, a ranked list will be more informative and useful than an unranked one in many cases. Additionally, a human-defined distance cutoff is used to decide whether or not a subspace is outlying with respect to the given point. This parameter will specify for each data point the outlying front (the boundary between those outlying subspaces and those nonoutlying ones) in the space lattice. Unfortunately, the value of this parameter cannot be easily specified due to the lack of prior knowledge about the underlying distribution of data points in the data set that may be rather complex in the high-dimensional space.

The major limitation of the genetic algorithm is that, for a particular subspace, only calculating the sparsity coefficients of the cubes in this subspace to which the given outlier examples belong is not adequate for giving an accurate measurement. For example, an outlier example p (highlighted by x) fall into the central cube in two two-dimensional subspaces, as shown in Figures 2a and 2b. We suppose that the cubes to which p belong in cases a and b (i.e., the central cube in the figure) are C_a and C_b, respectively. Apparently, the value of the sparsity coefficient of C_a is larger than C_b as the density of C_a is higher than C_b, but the outlier quality of p in a is actually smaller than b since p exhibits a more remarkable deviation from the majority of the population in the data set in b than in a. This inconsistency is caused by the fact that the sparsity coefficient measures the overall sparsity of data points in a hypercube and it is not necessarily able to accurately measure the outlier characteristics of a particular point in the hypercube.

FUTURE RESEARCH DIRECTIONS

More robust and efficient outlying subspace detection techniques need to be developed to well address the limitations of the existing methods. The new techniques are expected to have the following desired characteristics. First, these methods should be able to give a more accurate measurement of outlier characteristics of given data points in different subspaces. Due to the inherent sparsity of data in a high-dimensional space, it is important to take into account the behavior of the majority of data points in the data set when measuring the outlier quality of a particular given data point in a subspace rather than only considering the immediate neighborhood of the data point. Second, these methods should rank the detected outlying subspaces, and the specification of parameter values in the methods should be easy regardless of the data distribution of the data set. Finally, these methods should be able to explore the high-dimensional space efficiently. Heuristics are important to facilitate a speedup of the search process and render the outlying subspace detection problem more tractable.

CONCLUSION

This article formulates the outlying subspace detection problem and provides a survey of the existing methods for solving this problem. In particular, it focuses on the metrics used to measure the outlier quality of given data points in different subspaces and the searching strategies employed by the existing techniques for exploring high-dimensional space lattices. We have also pointed out the major limitations of the existing techniques, and some important issues to be considered in developing a better outlying subspace detection method in future research work.

REFERENCES

Aggarwal, C. C., & Yu, P. S. (2001). Outlier detection in high dimensional data. *Proceedings of the 2001 ACM SIGMOD International Conference on Management of Data (SIGMOD'01)* (pp. 37-46).

Aggrawal, R., Gehrke, J., Gunopulos, D., & Raghavan, P. (1998). Automatic subspace clustering of high dimensional data for data mining applications. *Proceedings of the 1998 ACM SIGMOD International Conference on Management of Data (SIGMOD'98)* (pp. 94-105).

Angiulli, F., & Pizzuti, C. (2002). Fast outlier detection in high dimensional spaces. *Proceedings of the 6th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD'02)* (pp. 15-26).

Barnett, V., & Lewis, T. (1994). *Outliers in statistical data* (3rd ed.). New York: John Wiley.

Boudjeloud, L., & Poulet, F. (2005). Visual interactive evolutionary algorithm for high dimensional data clustering and outlier detection. *Proceedings of the 9th Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining (PAKDD'05)* (pp. 426-431).

Breuning, M., Kriegel, H.-P., Ng, R., & Sander, J. (2000). LOF: Identifying density-based local outliers. *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data (SIGMOD'00)* (pp. 93-104).

Ester, M., Kriegel, H.-P., Sander, J., & Xu, X. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. *Proceedings of the 2nd ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD'96)* (pp. 226-231).

Hawkins, D. (1980). *Identification of outliers*. London: Chapman & Hall.

Hinneburg, A., & Keim, D. A. (1998). An efficient approach to cluster in large multimedia databases with noise. *Proceedings of the 3rd ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD'98)* (pp. 58-65).

Jin, W., Tung, A. K. H., & Han, J. (2001). Finding top *n* local outliers in large database. *Proceedings of the 7th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD'01)* (pp. 293-298).

Knorr, E. M., & Ng, R. T. (1998). Algorithms for mining distance-based outliers in large dataset. *Proceedings of the 24th International Conference on Very Large Data Bases (VLDB'98)* (pp. 392-403).

Knorr, E. M., & Ng, R. T. (1999). Finding intentional knowledge of distance-based outliers. *Proceedings of the 25th International Conference on Very Large Data Bases (VLDB'99)* (pp. 211-222).

Ng, R., & Han, J. (1994). Efficient and effective clustering methods for spatial data mining. *Proceedings of the 20th International Conference on Very Large Data Bases (VLDB'94)* (pp. 144-155).

Ramaswamy, S., Rastogi, R., & Shim, K. (2000). Efficient algorithms for mining outliers from large data sets. *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data (SIGMOD'00)* (pp. 427-438).

Sheikholeslami, G., Chatterjee, S., & Zhang, A. (1999). WaveCluster: A wavelet based clustering approach for spatial data in very large database. *VLDB Journal, 8*(3-4), 289-304.

Tang, J., Chen, Z., Fu, A., & Cheung, D. W. (2002). Enhancing effectiveness of outlier detections for low density patterns. *Proceedings of the 6th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD'02)* (pp. 535-548).

Wang, W., Zhang, J., & Wang, H. (2005). Grid-ODF: Detecting outliers effectively and efficiently in large multi-dimensional databases. *Proceedings of the 2005 International Conference on Computational Intelligence and Security (CIS'05)* (pp. 765-770).

Zhang, J., Hsu, W., & Lee, M. L. (2005). Clustering in dynamic spatial databases. *Journal of Intelligent Information Systems, 24*(1), 5-27.

Zhang, J., Lou, M., Ling, T. W., & Wang, H. (2004). HOS-miner: A system for detecting outlying subspaces of high-dimensional data. *Proceedings of the 30th International Conference on Very Large Data Bases (VLDB'04)* (pp. 1265-1268).

Zhang, J., & Wang, H. (2006). Detecting outlying subspaces for high-dimensional data: The new task, algorithms and performance. *Knowledge and Information Systems: An International Journal (KAIS).*

Zhang, T., Ramakrishnan, R., & Livny, M. (1996). BIRCH: An efficient data clustering method for very large databases. *Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data (SIGMOD'96)* (pp. 103-114).

Zhu, C., Kitagawa, H., & Faloutsos, C. (2005). Example-based robust outlier detection in high dimensional datasets. *Proceedings of the 2005 IEEE International Conference on Data Mining (ICDM'05)* (pp. 829-832).

Zhu, C., Kitagawa, H., Papadimitriou, S., & Faloutsos, C. (2004). OBE: Outlier by example. *Proceedings of the 8th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD'04)* (pp. 222-234).

KEY TERMS

Example-Based Outlier Mining: Given a set of outlier examples, example-based outlier mining finds more outliers from the dataset that exhibit the similar outlier qualities to the given outlier examples.

Genetic Algorithm: A genetic algorithm (abbreviated as GA) is a search technique used in computer science to approximate solutions to optimization and search problems.

Outlier Mining: Outlier mining is a data-mining task aiming to find a specific number of objects that are considerably dissimilar, exceptional, and inconsistent with respect to the majority records in the input databases.

Outlying Subspace: An outlying subspace of a point is a subspace (subset of features) in which this point is considerably dissimilar, exceptional, or inconsistent with respect to the remaining population in the database.

Random Sampling: Random sampling is a sampling technique where we select a group of subjects (a sample) for study from a larger group (a population). Each individual is chosen entirely by chance and each member of the population has a known, but possibly unequal, chance of being included in the sample.

Space Lattice: A space lattice is a lattice that contains all the possible subspaces of a data space. Each subspace in the lattice is represented as a combination of features of that subspace.

Subspace: A subspace is a combination of features or attributes of a database.

Chapter LX
Data Clustering

Yanchang Zhao
University of Technology, Sydney, Australia

Longbing Cao
University of Technology, Sydney, Australia

Huaifeng Zhang
University of Technology, Sydney, Australia

Chengqi Zhang
University of Technology, Sydney, Australia

INTRODUCTION

Clustering is one of the most important techniques in data mining. This chapter presents a survey of popular approaches for data clustering, including well-known clustering techniques, such as partitioning clustering, hierarchical clustering, density-based clustering and grid-based clustering, and recent advances in clustering, such as subspace clustering, text clustering and data stream clustering. The major challenges and future trends of data clustering will also be introduced in this chapter.

The remainder of this chapter is organized as follows. The background of data clustering will be introduced in Section 2, including the definition of clustering, categories of clustering techniques, features of good clustering algorithms, and the validation of clustering. Section 3 will present main approaches for clustering, which range from the classic partitioning and hierarchical clustering to recent approaches of bi-clustering and semi-supervised clustering. Challenges and future trends will be discussed in Section 4, followed by the conclusions in the last section.

BACKGROUND

Data clustering is sourced from pattern recognition (Theodoridis & Koutroumbas, 2006), machine learning (Alpaydin, 2004), statistics (Hill & Lewicki, 2007) and database technology (Date, 2003). *Data clustering* is to partition data into groups, where the data in the same group are similar to one another and the data from different groups are dissimilar (Han & Kamber, 2000). More specifically, it is to segment data into clusters

so that the intra-cluster similarity is maximized and that the inter-cluster similarity is minimized. The groups obtained are a partition of data, which can be used for customer segmentation, document categorization, etc.

Clustering techniques can be "clustered" into groups in multiple ways. In terms of the membership of objects, there are two kinds of clustering, *fuzzy clustering* and *hard clustering*. Fuzzy clustering is also known as *soft clustering*, where an object can be in more than one cluster, but with different membership degrees. In contrast, an object in hard clustering can belong to one cluster only. Generally speaking, clustering is referred to as hard clustering implicitly. In terms of approaches, data clustering techniques can be classified into the following groups: partitioning clustering, hierarchical clustering, density-based clustering, grid-based clustering and model-based clustering. In terms of the type of data, there are spatial data clustering, text clustering, multimedia clustering, time series clustering, data stream clustering and graph clustering.

For a good clustering algorithm, it is supposed to have the following features: 1) the ability to detect clusters with various shapes and different distributions; 2) the capability of finding clusters with considerably different sizes; 3) the ability to work when outliers are present; 4) no or few parameters needed as input; and 5) scalability to both the size and the dimensionality of data.

How to evaluate the results is an important problem for clustering. For the validation of clustering results, there are many different measures, such as *Compactness* (Zait & Messatfa, 1997), *Conditional Entropy (CE)* and *Normalized Mutual Information (NMI)* (Strehl & Ghosh, 2002; Fern & Brodley, 2003). The validation measures can be classified into three categories, 1) *internal validation*, such as *Compactness*, *Dunn's validation index*, *Silhouette index* and *Hubert's correlation with distance matrix*, which is based on calculating the properties of result clusters, 2) *relative validation*, such as *Figure of merit* and *Stability*, which is based on comparisons of partitions, and 3) *external validation*, such as *CE*, *NMI*, *Hubert's correlation*, *Rand statistics*, *Jaccard coefficient*, and *Folkes and Mallows index*, which is based on comparing with a known true partition of data (Halkidi et al., 2001, Brun et al., 2007).

DATA CLUSTERING TECHNIQUES

The popular clustering techniques will be briefly presented in this section. More detailed introduction and comparison of various clustering techniques can be found in books on data mining and survey papers on clustering (Berkhin, 2002; Grabmeier & Rudolph, 2002; Han & Kamber, 2000; Jain, Murty, & Flynn, 1999; Kolatch, 2001; Xu & Wunsch, 2005; Zait & Messatfa, 1997).

Partitioning Clustering

The idea of *partitioning clustering* is to partition the data into *k* groups first and then try to improve the quality of clustering by moving objects from one group to another. A typical method of partitioning clustering is *k-means* (Alsabti, Ranka, & Singh, 1998; Macqueen, 1967), which randomly selects *k* objects as cluster centers and assigns other objects to the nearest cluster centers, and then improves the clustering by iteratively updating the cluster centers and reassigning the objects to the new centers. *k-medoids* (Huang, 1998) is a variation of *k*-means for categorical data, where the medoid (i.e., the object closest to the center), instead of the centroid, is used to represent a cluster. Some other partitioning methods are PAM and CLARA proposed by Kaufman & Rousseeuw (1990) and CLARANS by Ng and Han (1994).

The disadvantage of partitioning clustering is that the result of clustering is dependent on the selection of initial cluster centers and it may result in a local optimum instead of a global one. A simple way to improve the chance of obtaining the global optimum is to run *k*-means

multiple times with different initial centers and then choose the best clustering result as output. Another disadvantage of *k*-means is that it tends to result in sphere-shaped clusters with similar sizes. Moreover, how to choose a value for *k* also remains as a non-trivial question.

Hierarchical Clustering

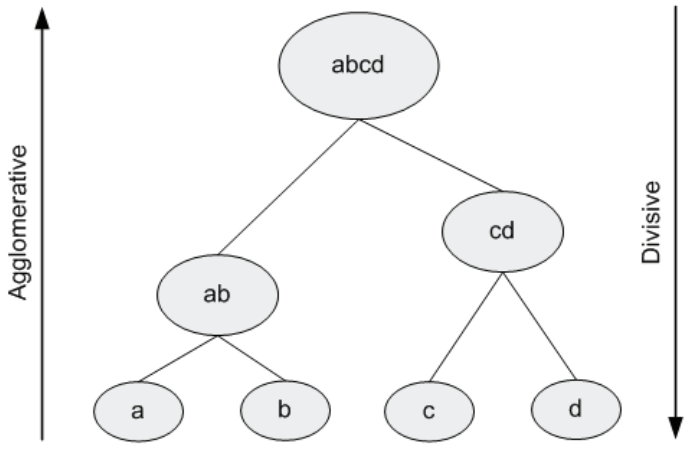
With *hierarchical clustering* approach, a hierarchical decomposition of data is built in either bottom-up (agglomerative) or top-down (divisive) way (see Figure 1). Generally a dendrogram is generated and a user may select to cut it at a certain level to get the clusters. With *agglomerative clustering,* every single object is taken as a cluster and then iteratively the two nearest clusters are merged to build bigger clusters until the expected number of clusters is obtained or when only one cluster is left. AGENS is a typical agglomerative clustering algorithm (Kaufman & Rousseeuw, 1990). *Divisive clustering* works in an opposite way, which puts all objects in a single cluster and then divides the cluster into smaller and smaller ones. An example of divisive clustering is DIANA (Kaufman & Rousseeuw, 1990). Some other popular hierarchical clustering algorithms are BIRCH (Zhang, Ramakrishnan, & Livny, 1996), CURE (Guha, Rastogi, & Shim, 1998), ROCK (Guha, Rastogi, & Shim, 1999) and Chameleon (Karypis, Han, & Kumar, 1999).

In hierarchical clustering, there are four different methods to measure the distance between clusters: centroid distance, average distance, single-link distance and complete-link distance. *Centroid distance* is the distance between the centroids of two clusters. *Average distance* is the average of the distances between every pair of objects from two clusters. *Single-link distance*, also known as *minimum distance*, is the distance between the two nearest objects from two clusters. *Complete-link distance*, also referred to as *maximum distance*, is the distance between the two objects which are the farthest from each other from two clusters.

Density-Based Clustering

The rationale of *density-based clustering* is that a cluster is composed of well-connected dense regions. DBSCAN is a typical density-based clustering algorithm, which works by expanding clusters to their dense neighborhood (Ester, Kriegel, Sander, & Xu, 1996). Although a user is not required to guess the number of clusters before clustering, he has to provide two other

Figure 1. Hierarchical clustering



parameters, the radius of neighborhood and the density threshold, to run DBSCAN. AGRID (Zhao & Song, 2003) is an efficient density-based algorithm in that it uses grid to reduce the complexity of distance computation and cluster merging. By partitioning the data space into cells, only neighboring cells are taken into account when computing density and merging clusters. Some other density-based clustering techniques are OPTICS (Ankerst, Breunig, Kriegel, & Sander, 1999) and DENCLUE (Hinneburg & Keim, 1998). The advantage of density-based clustering is that it can filter out noise and find clusters of arbitrary shapes (as long as they are composed of connected dense regions). However, most density-based approaches utilize the indexing techniques for efficient neighborhood inquiry, such as R-tree and R*-tree, which do not scalable well to high-dimensional space.

Grid-Based Clustering

Grid-based clustering works by partitioning the data space into cells with grid and then merging them to build clusters. Some grid-based methods, such as STING, WaveCluster and CLIQUE, use regular grids to partition data space, while some employ adaptive or irregular grids, such as adaptive grid (Goil, Nagesh, & Choudhary, 1999) and optimal grid (Hinneburg & Keim, 1999).

STING (Wang, Yang, & Muntz, 1997) is a grid-base multi-resolution clustering technique in which the spatial area is divided into rectangular cells and organized into a statistical information cell hierarchy. Statistical information, such as the count, mean, minimum, maximum, standard deviation and distribution, is stored for each cell. Thus, the statistical information of cells is captured and clustering can be performed without recourse to the individual objects. WaveCluster (Sheikholeslami, Chatterjee, & Zhang, 1998) is proposed to look at the multi-dimensional data space from a signal processing perspective. The objects are taken as a *d*-dimensional signal, and the high frequency parts of the signal correspond to the boundary of clusters, where the distribution of objects changes rapidly. The low frequency parts with high amplitude correspond to clusters, where data are concentrated. CLIQUE (Agrawal, Gehrke, Gunopulos, & Raghavan, 1998) works like APRIORI (Agrawal & Srikant, 1994), a famous algorithm for association mining. It partitions each dimension into intervals and computes the dense units in all dimensions. Then the dense units are combined to generate the dense units in higher dimensions.

The advantage of gird-based clustering is that the processing time is very fast, because it is independent on the number of objects, but dependent on the number of cells. Its disadvantage is that the quality of clustering is dependent on the granularity of cells and that the number of cells increases exponentially with the dimensionality of data. Therefore, adaptive grid and optimal grid are designed to tackle the above problems. MAFIA (Goil et al., 1999) uses adaptive grids to partition a dimension depending on the distribution of data in the dimension, in contrast to partition every dimension evenly. OptiGrid (Hinneburg & Keim, 1999) uses contracting projections of data to determine the optimal cutting hyper-planes for data partitioning, where the grid is "arbitrary", as compared with equidistant, axis-parallel grids.

Model-Based Clustering

Model-based clustering assumes that the data are generated by a mixture of probability distributions, and it attempts to learn statistical probability models from data, with each model representing one particular cluster (Zhong & Ghosh, 2003). The model type is often set as Gaussian or hidden Markov models (HMMs), and then the model structure is determined by model selection techniques and parameters are estimated using maximum likelihood algorithms. Some examples of model-based clustering methods are DMBC (Distributed Model-Based Clustering) (Kriegel et

al., 2005), and model-based clustering based on data summarization (bEMADS and gEMADS) (Jin et al., 2005). Another kind of model-based clustering is neural network approaches, such as SOM (self-organizing feature maps) (Kohonen, 1988).

Fuzzy Clustering

Generally speaking, an object is either a member of a cluster or not a member of it. *Fuzzy clustering* is an extension by allowing an object to be a member of more than one cluster. For example, an object is the member of three clusters with membership as 0.5, 0.3 and 0.2, respectively. Fuzzy clustering is also known as *soft clustering*, as compared with *hard clustering* where the membership can be either 1 or 0 only. Two examples of fuzzy clustering are fuzzy k-means (Karayiannis, 1995) and EM (*Expectation Maximization*) algorithm (Dempster, Laird, & Rubin, 1977). EM algorithm generates fuzzy clustering in two steps. The *expectation* (E) step calculates for each object the expectation of probabilities in each cluster, and then the *maximization* (M) step computes the distribution parameters of clusters, i.e., maximizing the likelihood of the distributions given the data. The two steps are repeated to improve the clustering, until the improvement (say, the increase in log-likelihood) becomes negligible. EM algorithms may result in a local maximum instead of the global maximum. Similar to *k*-means, the chance to get the global maximum can be improved by running EM for multiple times with different initial guesses and then choosing the best clustering.

Subspace Clustering

In some real-world applications, the dimensionality of data is in hundreds or even thousands, and more often than not, no meaningful clusters can be found in the full dimensional space. Therefore, *subspace clustering* is proposed for high dimensional data, where two clusters can be in two different subspaces and the subspaces can be of different dimensionalities. Well-known subspace clustering algorithms are CLIQUE (Agrawal et al., 1998), MAFIA (Goil et al., 1999), Random Projection (Fern & Brodley, 2003), Projected clustering (Aggarwal, Wolf, Yu, Procopiuc, & Park, 1999), Monte-Carlo (Procopiuc, Jones, Agarwal, & Murali, 2002), and Projective clustering (E. K. K. Ng, Fu, & Wong, 2005). With most techniques for subspace clustering, a cluster is defined as an axis-parallel hyper-rectangle in a subspace.

A technique for subspace clustering proposed by Fern and Brodley (2003) is to find the subspaces in a random projection and ensemble way. The dataset is first projected into random subspaces, and then EM algorithm is used to discover clusters in the projected dataset. The algorithm generates several groups of clusters with the above method and then combines them into a similarity matrix, from which the final clusters are discovered with an agglomerative clustering method.

MAFIA (Goil et al., 1999) is an efficient algorithm for subspace clustering using a density and grid based approach. It uses adaptive grids to partition a dimension depending on the distribution of data. The bins and cells that have low density of data are pruned to reduce computation. The boundaries of the bins are not rigid, which improves the quality of clustering.

Bi-Clustering

Bi-clustering, also known as *co-clustering* or *two-way clustering*, is to group objects for a subset of attributes by performing simultaneous clustering of both rows and columns (Cheng & Church, 2000; Madeira & Oliveira, 2004). It is a kind of subspace clustering. Bi-clustering is widely used for clustering microarray data to analyze the activities of genes under many different conditions. Microarray data can be viewed as a matrix, where each row represents a gene, each column stands for a condition and each

entry gives the expression level of a gene under a condition. From microarray data, four major types of biclusters are to discover: 1) biclusters with constant values, 2) biclusters with constant values on rows or columns, 3) biclusters with coherent values, and 4) biclusters with coherent evolutions (Madeira & Oliveira, 2004).

Text Clustering

Text clustering, also referred to as *document clustering*, is to group documents based on the similarity in the terms used and is widely used for document categorization, information retrieval and web search engine (Beil, Ester, & Xu, 2002; Zhong & Ghosh, 2005). Most algorithms for text clustering are based on a vector space model, where each document is represented by a vector of frequencies of terms. At first, a bag of words is collected for each document by filtering tags, stemming and pruning. Then the similarity of two documents is measured by how many words they share in common, and the documents can be clustered into groups with one of the clustering algorithms introduced above. Some examples of text clustering algorithms are Suffix Tree Clustering (Zamir & Etzioni, 1998), FTC (Frequent Term-based Clustering) and HFTC (Hierarchical FTC) (Beil, Ester, & Xu, 2002).

Data Stream Clustering

As tradition clustering deals with static data, *data stream clustering* is to cluster data streams where new data arrive continuously, such as click-streams, retail transactions and stock prices (C. C. Aggarwal, J. Han, J. Wang, & P. S. Yu, 2003; Guha, Meyerson, Mishra, Motwani, & O'Callaghan, 2003). For data stream clustering, the clusters are adjusted dynamically according to new data, and emerging clusters are also detected. New data can come in two ways, either as new records, such as transaction data, or as new dimensions, such as stock price data and other time series data. In either way, new data can bring changes in clusters as time elapses. Aggarwal et al. (2003) proposed the concepts of pyramid time frame and micro-clustering to cluster evolving data streams. The statistical information of data is stored as micro-clusters, and the micro-clusters are stored at snapshots in time following a pyramidal pattern. The above are then used in an offline process to explore stream clustering over different horizons.

Semi-Supervised Clustering

Generally speaking, clustering is unsupervised learning. However, sometimes there is a small amount of knowledge which can be used to guide clustering, and such kind of clustering is referred to as *semi-supervised clustering*. The knowledge available is normally not enough for a supervised learning to classify the data. The knowledge can be either pairwise constraints, such as must-link and cannot-link, or class labels for some objects. Some examples of semi-supervised clustering techniques are COP-COBWEB (Constraint-Partitioning COBWEB) (Wagstaff & Cardie, 2000), CCL (Constrained Complete-Link) (Klein et al., 2002), MPC-KMeans (Metric Pairwise Constrained K-Means) (Basu et al., 2003), semi-supervised clustering with user feedback (Cohn et al., 2003), and a probabilistic model for semi-supervised clustering (Basu et al., 2004).

FUTURE TRENDS

Data miming is confronted with larger volume of data, higher dimensionality, more complex data and new types of applications. The above are also challenges for clustering. More scalable algorithms are needed to clustering data in Gigabytes or even in Terabytes and of dimensionality in hundreds and even in thousands. In addition to scalability, the other problem introduced by high dimensionality is the meaningfulness of similar-

ity, the definition of clusters and the meaning of clustering. Another challenge is from new types of data and more complex data, such as multimedia data, semi-structured/unstructured data and stream data. The visualization of clusters and the change/trend analysis of clusters is also a trend of future research. More challenges will also be brought by new applications of clustering, such as bioinformatics, astronomy and meteorology.

CONCLUSION

We have presented a brief survey of popular data clustering approaches, including both classic methods and recent advanced algorithms. The basic ideas of the approaches have been introduced and their characteristics analyzed. The techniques are designed for different applications and for different types of data, such as numerical data, categorical data, spatial data, text data and microarray data. The definitions of clusters in the algorithms are not always the same, and most of them favor certain types of clusters, such as sphere-shaped clusters, convex clusters and axis-parallel clusters. New definitions of clusters and novel techniques for clustering keep emerging as data mining is applied in new applications and in new fields.

REFERENCES

Aggarwal, C. C., Han, J., Wang, J., & Yu, P. S. (2003). *A framework for clustering evolving data streams.* Paper presented at the 29th international conference on Very Large Data Bases, Berlin, Germany.

Aggarwal, C. C., Wolf, J. L., Yu, P. S., Procopiuc, C., & Park, J. S. (1999). *Fast algorithms for projected clustering.* Paper presented at SIGMOD '99: the 1999 ACM SIGMOD international conference on Management of Data, New York, NY, USA.

Agrawal, R., Gehrke, J., Gunopulos, D., & Raghavan, P. (1998). *Automatic subspace clustering of high dimensional data for data mining applications.* Paper presented at SIGMOD '98: the 1998 ACM SIGMOD international conference on Management of Data, New York, NY, USA.

Agrawal, R., & Srikant, R. (1994, September). *Fast Algorithms for Mining Association Rules in Large Databases.* Paper presented at the 20th International Conference on Very Large Data Bases (VLDB), Santiago, Chile.

Alpaydin, E. (2004). *Introduction to Machine Learning*: The MIT Press.

Alsabti, K., Ranka, S., & Singh, V. (1998). *An Efficient K-Means Clustering Algorithm*. Paper presented at the First Workshop on High Performance Data Mining, Orlando, Florida.

Ankerst, M., Breunig, M. M., Kriegel, H.-P., & Sander, J. (1999). *OPTICS: ordering points to identify the clustering structure.* Paper presented at SIGMOD '99: the 1999 ACM SIGMOD international conference on Management of Data, New York, NY, USA.

Basu, S., Bilenko, M., & Mooney, R. (2003). *Comparing and unifying search-based and similarity-based approaches to semi-supervised clustering.* Paper presented at ICML'03: the twentieth International Conference on Machine Learning, 2003.

Basu, S.; Bilenko, M., & Mooney, R. J. (2004). *A probabilistic framework for semi-supervised clustering.* Paper presented at KDD '04: the tenth ACM SIGKDD international conference on Knowledge discovery and data mining, ACM, 2004, 59-68.

Beil, F., Ester, M., & Xu, X. (2002). *Frequent term-based text clustering.* Paper presented at the eighth ACM SIGKDD international conference on Knowledge Discovery and Data Mining.

Berkhin, P. (2002). *Survey of Clustering Data Mining Techniques.* Accrue Software, San Jose, CA, USA. URL: http://citeseer.ist.psu.edu/berkhin02survey.html.

Brun, M., Sima, C., Hua, J., Lowey, J., Carroll, B., Suh, E., & Dougherty, E. R. (2007). *Model-based evaluation of clustering validation measures, 40,* 807-824. Pattern Recognition, Elsevier Science Inc.

Cheng, Y., & Church, G. M. (2000). *Biclustering of Expression Data*. Paper presented at the eighth International Conference on Intelligent Systems for Molecular Biology.

Cohn, D., Caruana, R., & McCallum, A. (2003). *Semi-supervised clustering with user feedback.* Technical Report TR2003-1892, Cornell University, USA.

Date, C. J. (2003). *An Introduction to Database Systems*: Addison-Wesley Longman Publishing Co., Inc.

Dempster, A., Laird, N., & Rubin, D. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B, 39*(1), 1-38.

Ester, M., Kriegel, H.-P., Sander, J. o. r., & Xu, X. (1996). *A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise.* Paper presented at the second International Conference on Knowledge Discovery and Data Mining (KDD'96).

Fern, X., & Brodley, C. (2003). *Random Projection for High Dimensional Data Clustering: A Cluster Ensemble Approach.* Paper presented at the twentieth International Conference on Machine Learning (ICML'03).

Goil, S., Nagesh, H., & Choudhary, A. (1999). *MAFIA: Efficient and scalable subspace clustering for very large data sets.* Technical Report CPDC-TR-9906-010: Northwestern University.

Grabmeier, J., & Rudolph, A. (2002). Techniques of Cluster Algorithms in Data Mining. *Data Mining and Knowledge Discovery, 6*(4), 303-360.

Guha, S., Meyerson, A., Mishra, N., Motwani, R., & O'Callaghan, L. (2003). Clustering Data Streams: Theory and Practice. I*EEE Transactions on Knowledge and Data Engineering, 15*(3), 515-528.

Guha, S., Rastogi, R., & Shim, K. (1998). *CURE: an efficient clustering algorithm for large databases.* Paper presented at SIGMOD '98: the 1998 ACM SIGMOD international conference on Management of data, New York, NY, USA.

Guha, S., Rastogi, R., & Shim, K. (1999). *ROCK: A Robust Clustering Algorithm for Categorical Attributes.* Paper presented at the 15th International Conference on Data Engineering, 23-26 March 1999, Sydney, Australia.

Halkidi, M., Batistakis, Y., & Vazirgiannis, M. (2001). On Clustering Validation Techniques. *Journal of Intelligent Information Systems, 2001, 17,* 107-145.

Han, J., & Kamber, M. (2000). *Data mining: concepts and techniques.* San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.

Hill, T., & Lewicki, P. (2007). *Statistics: Methods and Applications.* Tulsa, OK: StatSoft.

Hinneburg, A., & Keim, D. A. (1998). *An Efficient Approach to Clustering in Large Multimedia Databases with Noise.* Paper presented at the fourth International Conference on Knowledge Discovery and Data Mining (KDD'98).

Hinneburg, A., & Keim, D. A. (1999). *Optimal Grid-Clustering: Towards Breaking the Curse of Dimensionality in High-Dimensional Clustering.* Paper presented at the VLDB'99: the 25th International Conference on Very Large Data Bases, September 7-10, 1999, Edinburgh, Scotland, UK.

Huang, Z. (1998). Extensions to the k-Means Algorithm for Clustering Large Data Sets with Categorical Values. *Data Mining and Knowledge Discovery, 2*(3), 283-304.

Jain, A. K., Murty, M. N., & Flynn, P. J. (1999). Data clustering: a review. *ACM Computing Surveys, 31*(3), 264-323.

Jin, H.; Wong, M., & Leung, K. (2005). Scalable Model-Based Clustering for Large Databases Based on Data Summarization. *IEEE Transactions on Pattern Anal. Mach. Intell, 27,* 1710-1719. IEEE Computer Society.

Karayiannis, N. B. (1995). *Generalized fuzzy k-means algorithms and their application in image compression.* Paper presented at the Society of Photo-Optical Instrumentation Engineers (SPIE) Conference.

Karypis, G., Han, E.-H., & Kumar, V. (1999). Chameleon: hierarchical clustering using dynamic modeling. *Computer, 32*(8), 68-75.

Kaufman, L., & Rousseeuw, P. J. (1990). Finding groups in data. an introduction to cluster analysis: Wiley Series in Probability and Mathematical Statistics. *Applied Probability and Statistics,* New York: Wiley, 1990.

Klein, D., Kamvar, S. D., & Manning, C. D. (2002). *From Instance-level Constraints to Space-Level Constraints: Making the Most of Prior Knowledge in Data Clustering.* Paper presented at ICML '02: the nineteenth International Conference on Machine Learning, Morgan Kaufmann Publishers Inc., 2002, 307-314.

Kohonen, T. (1988). *Self-organized formation of topologically correct feature maps.* Neurocomputing: foundations of research, MIT Press, 1988, (pp. 509-521).

Kolatch, E. (2001). *Clustering Algorithms for Spatial Databases: A Survey.* Unpublished manuscript. Department of Computer Science, University of Maryland, College Park. URL: "http://citeseer.ist.psu.edu/kolatch01clustering.html"

Kriegel, H., Kroger, P., Pryakhin, A., & Schubert, M. (2005). *Effective and Efficient Distributed Model-Based Clustering.* Paper presented at ICDM '05: the Fifth IEEE International Conference on Data Mining, IEEE Computer Society, 2005, 258-265.

Macqueen, J. B. (1967). *Some methods of classification and analysis of multivariate observations.* Paper presented at the the Fifth Berkeley Symposium on Mathematical Statistics and Probability.

Madeira, S. C., & Oliveira, A. L. (2004). Biclustering Algorithms for Biological Data Analysis: A Survey. *IEEE/ACM Transactions Computational Biology and Bioinformatics, 1*(1), 24-45.

Ng, E. K. K., Fu, A. W.-c., & Wong, R. C.-W. (2005). Projective Clustering by Histograms. *IEEE Transactions on Knowledge and Data Engineering, 17*(3), 369-383.

Ng, R. T., & Han, J. (1994). *Efficient and Effective Clustering Methods for Spatial Data Mining.* Paper presented at VLDB'94: the 20th International Conference on Very Large Data Bases, San Francisco, CA, USA.

Procopiuc, C. M., Jones, M., Agarwal, P. K., & Murali, T. M. (2002). *A Monte Carlo algorithm for fast projective clustering.* Paper presented at SIGMOD'02: the 2002 ACM SIGMOD international conference on Management of data, New York, NY, USA.

Sheikholeslami, G., Chatterjee, S., & Zhang, A. (1998). *WaveCluster: A Multi-Resolution Clustering Approach for Very Large Spatial Databases.* Paper presented at VLDB '98: the 24th International Conference on Very Large Data Bases, San Francisco, CA, USA.

Strehl, A., & Ghosh, J. (2002). Cluster ensembles - a knowledge reuse framework for combining multiple partitions. *Machine Learning Research, 3,* 583-417.

Theodoridis, S., & Koutroumbas, K. (2006). *Pattern Recognition,* Third Edition: Academic Press, Inc.

Wagstaff, K., & Cardie, C. (2000). *Clustering with Instance-level Constraints.* Paper presented at ICML'00: the seventeenth International Conference on Machine Learning, Morgan Kaufmann Publishers Inc., 2000, 1103-1110.

Wang, W., Yang, J., & Muntz, R. R. (1997). *STING: A Statistical Information Grid Approach to Spatial Data Mining.* Paper presented at VLDB'97: the 23rd International Conference on Very Large Data Bases, August 25-29, 1997, Athens, Greece.

Xu, R., & Wunsch, D., II. (2005). Survey of clustering algorithms. *IEEE Transactions on Neural Networks, 16*(3), 645-678.

Zait, M., & Messatfa, H. (1997). A comparative study of clustering methods. *Future Generation Computer Systems, 13*(2-3), 149-159.

Zamir, O., & Etzioni, O. (1998). *Web document clustering: a feasibility demonstration.* Paper presented at SIGIR '98: the 21st annual international ACM SIGIR conference on Research and development in information retrieval, ACM, 1998, 46-54.

Zhang, T., Ramakrishnan, R., & Livny, M. (1996). *BIRCH: an efficient data clustering method for very large databases.* Paper presented at SIGMOD'96: the 1996 ACM SIGMOD international conference on Management of data, New York, NY, USA.

Zhao, Y., & Song, J. (2003). *AGRID: An Efficient Algorithm for Clustering Large High-Dimensional Datasets.* Paper presented at the PAKDD'03: 7th Pacific-Asia Conference on Knowledge Discovery and Data Mining, Seoul, Korea.

Zhong, S., & Ghosh, J. (2003). A unified framework for model-based clustering. *Journal of Machine Learning Research, 4,* 1001-1037. MIT Press.

Zhong, S., & Ghosh, J. (2005). Generative model-based document clustering: a comparative study. *Knowledge and Information Systems, 8*(3), 374-384.

KEY TERMS

Bi-Clustering: Also known as co-clustering, it is to group objects for a subset of attributes by performing simultaneous clustering of both rows and columns.

Data Clustering: Data clustering is to partition data into groups, where the data in the same group are similar to one another and the data from different groups are different from one another.

Data Stream Clustering: It is to group continuously arriving data, instead of static data, into groups based on the similarity.

Density-Based Clustering: Density-based clustering takes densely populated regions as clusters, while objects in sparse areas are removed as noises.

Fuzzy Clustering: Also known as Soft Clustering. For fuzzy clustering, an object can be classified with fractional membership into multiple groups, in contrast to Hard Clustering where an object can be classified into one group only.

Grid-Based Clustering: It is to partition the whole space into cells with grids and then merge the cells to build clusters.

Hierarchical Clustering: It is to build a hierarchical decomposition of data in either bottom-up or top-down way. Generally a dendrogram is generated and a user may select to cut it at a certain level to get the clusters.

Model-Based Clustering: Model-based clustering assumes that the data are generated by a mixture of probability distributions, and attempts to learn statistical probability models from data, with each model representing one particular cluster.

Partitioning Clustering: It is a clustering approach which uses centers to represent clusters and then improves the partitioning by moving objects from group to group.

Semi-Supervised Clustering: Semi-supervised clustering is a partly supervised clustering which is guided with a small amount of knowledge, such as pairwise constraints and class labels for some objects.

Subspace Clustering: It is to find clusters in subspaces, where two clusters may exist in two different subspaces and the subspaces may also have different dimensionalities.

Text Clustering: It is to group documents based on the similarity in their topics and text.

Chapter LXI
C-MICRA:
A Tool for Clustering Microarray Data

Emmanuel Udoh
Indiana University–Purdue University, USA

Salim Bhuiyan
Indiana University–Purdue University, USA

INTRODUCTION

In the field of bioinformatics, small to large data sets of genes, proteins, and genomes are analyzed for biological significance. A technology that has been in the forefront of generating large amounts of gene data is the microarray or hybridization technique. It has been instrumental in the success of the human genome project and paved the way for a new era of genetic screening, testing, and diagnostics (Scheena, 2003). The microarray data set can be made of thousands of rows and columns. It often contains missing values, exhibits high-dimensional attributes, and is generally too large for manual management or examination (Tseng & Kao, 2005; Turner, Bailey, Krzanowski, & Hemingway, 2005). Database technology is necessary for the extraction, sorting, and analyzing of microarray data sets.

A plethora of techniques from machine learning, pattern recognition, statistics, and databases has been deployed to address the issue of knowledge discovery from large microarray data sets. Techniques commonly employed in this endeavor are data transformation, scatter plots, principle component analysis, expression maps, pathway analysis, workflow management, support vector machines, artificial neural networks, and cluster analysis (Baxevanis & Ouellette, 2005; Rogers, Girolami, Campbell, & Breitling, 2005). These techniques can be categorized into two groups: supervised or unsupervised methods. The supervised methods, such as support vector machines and linear discriminant analysis, require two sets of measurements. The first set represents the expression measurements from the microarray gene chips run on a set of samples, while the second is the data characterizing the samples under investigation. The goal of this method is to train the model for predictive purposes. Supervised methods tend to determine genes that fit a predetermined pattern. For a broad overview of supervised methods, the reader is referred to Scheena (2003). The second category of approaches is the unsupervised method, and it characterizes components of a data set without a prior input or knowledge

of a training signal. Clustering is an example of an unsupervised method. While these techniques are useful in microarray analysis, the focus of this article will be on clustering.

Clustering is an important unsupervised method in the exploration of the expression patterns in microarray analysis. As a tool of discovery, clustering classifies similar objects into different groups or nonoverlapping clusters so that the data in each group share commonality, often proximity according to some defined distance measure. Clustering algorithms can be partitional or hierarchical. Hierarchical algorithms find successive clusters using previously established clusters, whereas partitional algorithms determine all clusters at once (Bolshakova & Azuaje, 2006). These algorithms can be used to determine what group a particular genetic sample belongs to and the tendency for certain clusters to be associated with certain characteristics. The most widely used clustering techniques are hierarchical, *k*-means, and self-organizing maps. In the literature, other terms used interchangeably for clustering are cluster analysis, automatic classification, numerical taxonomy, botryology, and typological analysis (Fung, Ye, & Zhang, 2003; Glenisson, Mathys, & De Moor, 2003).

BACKGROUND

Numerous data mining studies based on partitioning groups in multidimensional microarray data sets reveal the significance of clustering in biological information extraction (Au, Chan, Wong, & Wang, 2005; Bolshakova & Azuaje, 2006; Lacroix, 2002; Piatetsky-Shapiro & Tamayo, 2003). A broad overview of biostatistical clustering approaches in microarray analysis is given by Scheena (2003). There are large clustering algorithms in the literature, but they are relatively equivalent in performance.

Clustering techniques partition data that are not a priori known to contain any identified subsets and can determine intrinsic grouping in a set of microarray data using distance or conceptual measures. To determine membership in a cluster, clustering algorithms evaluate distance between a point and the cluster centroid. The output is basically a statistical description of the cluster centroid with the number of components in each cluster. It is a data reduction method in that the observations in a group can be viewed as a mean of the observations in that subset. However, domain knowledge is useful to formulate appropriate measures in a clustering algorithm, which may be exclusive, overlapping, hierarchical, or probabilistic (Hanczar, Courtine, Benis, Hennegar, Clement, and Zucker, 2003; Parsons, Haque, & Liu, 2004).

Programs such as Cluster (Eisen, Spellman, Brown, & Botstein, 1998) and Hierarchical Clustering Explorer (HCE; Seo & Schneiderman, 2002) provide useful insights on summarized representations of groups in microarray data. Eisen et al. implemented in Cluster a hierarchical clustering (HC) algorithm based on the average linkage method of Sokal and Michener, which was developed for clustering correlation matrices. Although Cluster implements hierarchical clustering, *k*-means, and self-organizing maps efficiently, it is generally known to be weak in visualization. To strengthen Cluster, an interactive graphical program TreeView was developed to visualize the results of Cluster. It allows tree and image-based browsing of hierarchical clusters. Another program with interactive capability is the HCE. This program implements only hierarchical clustering, but it has more functionalities than Cluster and TreeView. For instance, HCE implements several techniques involving entire data sets, dynamic query controls, coordinated displays, scatter plots for relevance ordering, gene ontology browsing, and profile search with temporal patterns. Based on the method developed by Yeung, Haynor, and Ruzzo (2001), HCE can be described to have high predictive power. The C-MICRA (Clustering Microarray) program developed by the authors has several features available in HCE except scatter plots and gene ontology browsing, and it manages scrolling and cluster comparison better than HCE.

With the advent of information visualization techniques, there is a general revival of interest in the clustering method. Clustering handles efficiently the large data set generated by microarray hybridization experiments and appropriately presents a visual high-level summary of each cluster. It reduces redundancy and cognitive demand and is frequently one of the first techniques deployed to forage information from the cumbersome microarray data set, thus underscoring its importance in biological information systems.

MAIN FOCUS

A current research theme in clustering microarray data focuses on identifying specific informative subgroups using interactive visualization techniques. Interactive clustering approaches are advantageous and effective (Jiang, Pei, & Zhang, 2003; Seo & Schneiderman, 2005; Wu, Ye, & Zhang, 2003). The authors of this article developed a program, C-MICRA, that enables users to determine natural grouping in microarray data. In harnessing the techniques, these steps are commonly deployed: sourcing microarray data, filtering, hierarchical clustering, and visualization.

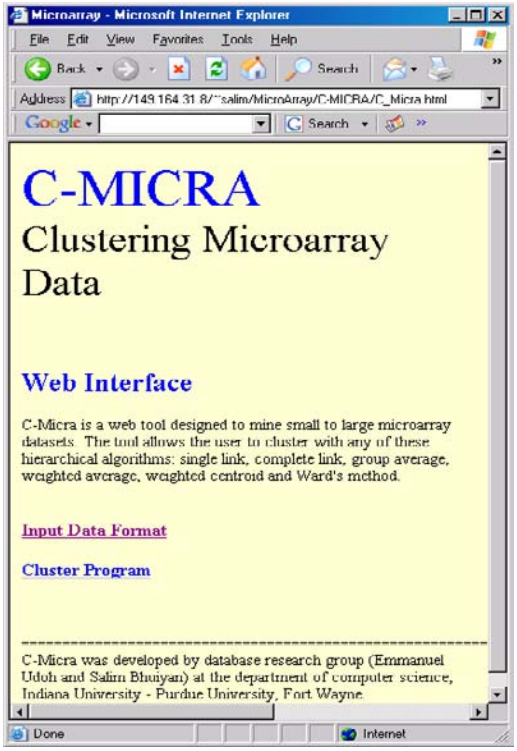
Microarray Data

Microarray data are gene expression values that are usually preprocessed with certain vendors' software before usage because the actual gene expression data are determined from the fluorescence intensity at each microarray spot. Preprocessing has a significant impact on the data quality and is an important area of active research (Baxevanis & Ouellette, 2005; Scheena, 2003). Analyzing microarray data presupposes that preprocessing was well done and that the data quality is good. The preprocessed data are usually stored in a matrix form. ChipWriter is one example of software used to extract expression values from images and scaling algorithms to make expression values comparable across chips (Baxevanis & Ouellette; Scheena). In most occasions, microarray data used for research are preprocessed, with dimensions in hundreds of rows and columns. For the development of the authors' program, a microarray data set with dimension 504x227 with genetic profiles for prostate, liver, breast, colon, lung, kidney, testis, and ovary cancer was used as the base data.

Filtering and Normalization

Genes need be expressed to be used. The filtering technique is a data reduction approach used to remove genes that are not well expressed or invariant across sample types. It is reductive in that the rows corresponding to the genes are removed from the matrix, thus making the data set smaller. Some gene chips contain variables that show if the gene is expressed, not expressed, or indeterminate. An approach deployed in filtering is using the threshold of the gene variance (Baxevanis & Ouellette,

Figure 1. Web interface to the cluster program C-MICRA



2005; Scheena, 2003). Filtering enhances feature selection, but care must be exercised to preserve important genes when removing the least differentially expressed genes.

Furthermore, it is essential to standardize data before usage in microarray analysis, using techniques like variance stabilization and normalization. Normalization is a form of standardization in that the data are transformed into a format needed for meaningful statistical analysis. Normalization of sampled genes is achieved by subtracting from each expression level the mean of the expression levels for that gene and then dividing by the standard deviation of that gene (Scheena, 2003). The clustering approach applied here is distance based and also sensitive to differences in the absolute values of the gene expression values, thus requiring normalization.

Hierarchical Clustering

Hierarchical clustering algorithms are commonly used to analyze microarray data by using the distance measures between the rows of the data matrices. HC can be divisive (top-down) or agglomerative (bottom-up). The agglomerative method is common, while the divisive approach is rarely used. The agglomerative approach begins with each element as a separate cluster and merges them in successively larger or super clusters until one group is left. The merge sequence generates a hierarchy based on similarity. The algorithm initially places each element into a group by itself. If there is more than one group, the algorithm searches for similarities and merges two similar groups. However, it must be emphasized that hierarchical clustering has no probabilistic foundation and cannot statistically test where to cut the cluster diagram.

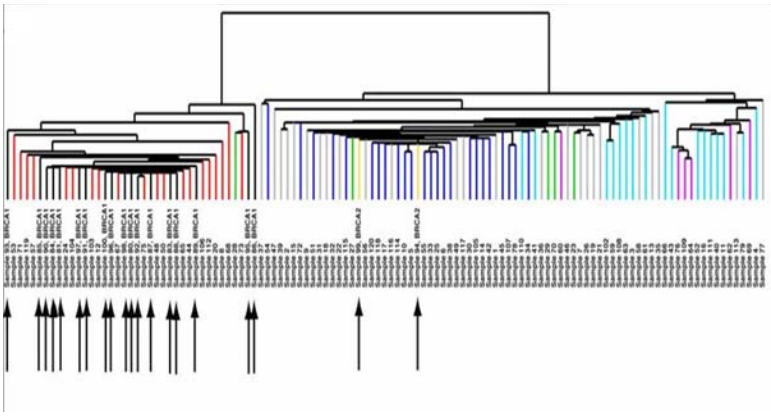
As regards program implementation, the contemporary computing landscape in biological clustering information systems is typically Web based with architectures that are two tiered or three tiered. In a two-tier or client-server computing mode, the system client requests services directly from the server and visualizes the data to the user, while the server typically acts as the data store. Using private communication protocols, it is relatively less daunting to configure the system but demanding to upgrade and extend. On the other hand, the three-tier architecture, which is a client-server mode with middleware, outperforms the two-tier mode because the system client's requests to the database server are coordinated by the intermediate server. The handling of the business rules and data access functions at this intermediate level provides added efficiency and has helped to popularize the three-tiered architecture in biological information systems. The three-tiered architecture was used to design the clustering program C-MICRA.

Figure 1 is the Web interface to C-MICRA, a hierarchical clustering program developed by the authors. It was developed in a PHP/MySQL environment and hosted on a Tomcat server that provides global access to the system. The program partitions microarray data into intuitive groups and allows users to explore and compare the different algorithms. There are three program modules in the system, formatting, clustering, and visualization, in addition to a unified interface to streamline the workflow of microarray analysis. The formatting module enables the upload of data in Excel format and the eventual reformatting of data in the MySQL database en route to the clustering component. The visualization module uses the dendrogram and ghostscript programs to create an intuitive graphical interface. The clustering module implements different mathematical methods to partition a sample.

The implemented methods, which are described below, are single link, complete link, group average, weighted average, weighted centroid, and Ward's method:

1. **Single link (nearest neighbor):** The distance used in this method is the one between the two nearest or closest neighbors in the different groups, producing often elongated clusters.
2. **Complete link (furthest neighbor):** This method uses the greatest or furthest distances

Figure 2. A dendrogram of microarray data from C-MICRA (arrow indicates abnormal sample)



between the different clusters as the distance measure. It is more associated with clusters that look bundled together as opposed to elongated ones.

3. **Group average:** In this approach, the average distance between all pairs of objects in the two different groups represents the distance between two clusters.
4. **Weighted average:** It is similar to the group-average method except that the number of elements in each cluster is used as a weight for the computations. This method is efficient in situations where cluster sizes make a difference.
5. **Weighted centroid (median):** The difference between the centers of the clusters (known as the centroid) is used to denote the distance measure for the unweighted centroid method. When weight is introduced, it is known as the weighted centroid approach. The weighting process allows the number of objects that determines the cluster size to be considered in situations where size difference is significant.
6. **Ward's method:** This is one of the most popular hierarchical agglomerating clustering algorithms, proposed by the statistician Ward (Baxevanis & Ouellette, 2005). This approach uses the analysis of variance (ANOVA) to determine the distances between clusters. It essentially minimizes the sum of squares between two partitions produced at each stage of the cluster computation.

Visualization and Dendrograms

A microarray data set is intrinsically noisy, requiring robust visualization techniques to detect patterns and outliers in the system. Due to its extensive numerical range, many microarray visualization techniques use a cutoff value to specify the display limit. This allows the observation of the variation around the zero value, which may obscure variation of the outliers. At a minimum, this cutoff value should be dynamically controlled by the user so that there is the ability to see both types of variation (Hibbs, Dirksen, Li, & Troyanskaya, 2005). A rank-based visualization method, such as the rank-based Spearman correlation coefficient, has been proposed to serve as a complement to these noise-robust distance metrics (Hibbs et al.; Seo & Schneiderman, 2005). This will complement the ability to change cutoff values, increase dynamic range, and decrease the effects of noise in visualization.

Generally, it is important to visualize the output of microarray analysis in an understandable way. C-MICRA harnesses the rich visualization features in PHP/MySQL to implement a hierarchical clustering program as depicted in Figure 2. The HC result is generally represented as a

binary tree called a dendrogram, whose subtrees are clusters. Users can see the overall cluster result in a single screen or zoom in to see more details. The lower the subtree, the more similar the items in the subtrees are. Users can control the level of details by using a cutoff bar. When the number of data samples is quite large, the dendrogram visualization suffers from slow display and scrolling. It may not be able to show the overview of the entire dendrogram window. In some instances, the heat map visualization can be used to see a color mosaic, which also may not show a strong relation between the visualizations (Hibbs et al., 2005). As can be gleaned from Figure 2, the dendrogram can produce categorized representation of microarray data that can be associated with diseases.

FUTURE TRENDS

Microarray data constitute important genetic data sets and will continue to attract improved data mining techniques in the future. Techniques that are based on human-computer interaction such as interactive hierarchical clustering will be extensively developed in the near future. Furthermore, visualization techniques coupled with query systems will be improved. There are also attempts to develop a mathematical framework to integrate the results of multiple cluster analyses into a single one. This method will put two genes in the same gene cluster in the final analysis if the genes are similar. Thus, conceptually, the different cluster methods will group similar genes into the same cluster and dissimilar ones into different subgroups.

CONCLUSION

Microarray data are often large and cumbersome to handle manually. Several algorithms have been developed to harness these data sets. We provided a brief description of hierarchical and nonhierarchical clustering methods, in addition to an implemented program C-MICRA. The authors are of the view that microarray data can be handled more efficiently using distance-based methods as opposed to parametric ones since the assumptions of parametric approaches have currently weak support. Finally, clustering techniques are data reduction methods and can be effective in detecting relevant genetic markers in microarray data.

REFERENCES

Au, W., Chan, K. C., Wong, A., & Wang, Y. (2005). Attribute of grouping, selection, and classification of gene expression data. *IEEE/ACM Transactions on Computational Biology and Bioinformatics, 2*(2), 83-101.

Baxevanis, A. D., & Ouellette, F. (2005). *Bioinformatics: A practical guide to the analysis of genes and proteins.* New York: Wiley-Interscience.

Bolshakova, N., & Azuaje, F. (2006). Estimating the number of clusters in DNA microarray data. *Methods of Information in Medicine, 45*(2), 153-157.

Eisen, M. B., Spellman, P. T., Brown, P. O., & Botstein, D. (1998). Cluster analysis and display of genome-wide expression patterns. *Proceedings of the National Academy of Science, 95*, 14863-14868.

Fung, B. Y. M., Ye, Y., & Zhang, L. (2003). Classification of heterogeneous gene expression data. *ACM-SIGKDD Explorations, 5*(2), 69-78.

Glenisson, P., Mathys, J., & De Moor, B. (2003). Meta-clustering of gene expression data and literature based information. *SIGKDD Explorations, 5*(2), 101-112.

Hanczar, B., Courtine, M., Benis, A., Hennegar, C., Clement, K., & Zucker, J.-D. (2003). Improving classification of microarray data using prototype-based feature selection. *ACM-SIGKDD Explorations, 5*(2), 23-30.

Hibbs, M. A., Dirksen, N. C., Li, K., & Troyanskaya, O. G. (2005). Visualization methods for

statistical analysis of microarray clusters. *BMC Bioinformatics, 6*(115), 10.

Jiang, D., Pei, J., & Zhang, A. (2003). Toward interactive exploration of gene expression patterns. *ACM-SIGKDD Explorations, 5*(2), 79-90.

Lacroix, Z. (2002). Biological data integration: Wrapping data and tools. *IEEE Transactions on Information Technology in Biomedicine, 6*(2), 123-128.

Parsons, L., Haque, E., & Liu, H. (2004). Subspace clustering for high dimensional data: A review. *ACM-SIGKDD Explorations Newsletter, 6*(1), 90-105.

Piatetsky-Shapiro, G., & Tamayo, P. (2003). Microarray data mining: Facing the challenges.

ACM-SIGKDD Explorations, 5(2), 1-5.

Rogers, S., Girolami, M., Campbell, C., & Breitling, R. (2005). The latent process decomposition of cDAN microarray data sets. *IEEE/ACM Transactions on Computational Biology and Bioinformatics, 2*(2), 143-156.

Scheena, M. (2003). *Microarray analysis.* NJ: Wiley.

Seo, J., & Schneiderman, B. (2002). Interactively exploring hierarchical clustering results. *IEEE Computer, 35*(7), 80-86.

Seo, J., & Schneiderman, B. (2005). A rank-by-feature framework for interactive exploration of multidimensional data. *Information Visualization, 4*(2), 99-113.

Tseng, V. S., & Kao, C.-P. (2005). Efficiently mining gene expression data via a novel parameterless clustering method. *IEEE/ACM Transactions on Computational Biology and Bioinformatics, 2*(4), 355-365.

Turner, H. L., Bailey, T.C., Krzanowski, W. J., & Hemingway, C. A. (2005). Biclustering models for structured microarray data. *IEEE/ACM Transactions on Computational Biology and Bioinformatics, 2*(4), 316-329.

Wu, X., Ye, Y., & Zhang, L. (2003). Graphical modeling based gene interaction analysis for microarray data. *SIGKDD Explorations Newsletter, 5*(2), 91-100.

Yeung, K. Y., Haynor, D. R., & Ruzzo, W. L. (2001). Validating clustering for gene expression data. *Bioinformatics, 17*, 309-318.

KEY TERMS

Bioinformatics: A broad term used to describe the collection, organization, and analysis of biological data using computer technology and information science. Some bioinformatics activities include the modeling of protein structure and creation of extensive electronic databases on genes, genomes, and protein sequences.

Centroid: The centroid of a cluster is the average point in the multidimensional space defined by the dimensions.

Dendrogram: A two-dimensional diagram illustrating the fusions or divisions made at each successive stage of hierarchical clustering.

Filtering: The process of removing genes that are not expressed or do not vary across sample types.

Hierarchical Clustering: This involves the recursive clustering of data points, which may be agglomerative or divisive. An agglomerative clustering method starts with each case in a separate cluster and then combines the clusters sequentially, reducing the number of clusters at each step until only one cluster is left. The divisive hierarchical clustering starts with all objects in one cluster and then subdivides them into smaller pieces.

***K*-Means Clustering:** Objects are grouped into a fixed number (k) of partitions so that the partitions are dissimilar to each other.

Microarray Analysis: It involves five-step exploratory approach to analyzing microarray data. The first step involves posing the biological

question the experiment should address. Sample preparation is the next step, which includes experiments to be performed such as DNA/ RNA isolation, probe amplification, and microarray manufacture. The next step is the biochemical reactions between the target and probe molecules, for example, hybridization. The fourth step detects captured images from the microarray using scanners or imaging instruments. Finally, the captured images are analyzed and modeled.

Microarray Data: This is a data set in matrix form containing gene expression values of samples. The row entry represents a gene while the column contains the chip.

Self-Organizing Maps (SOMs): A neural-network method that reduces the dimensions of data while preserving the topological properties of the input data. SOM is suitable for visualizing high-dimensional data such as microarray data.

Chapter LXII
Deep Web:
Databases on the Web

Denis Shestakov
Turku Centre of Computer Science, Finland

INTRODUCTION

Finding information on the Web using a web search engine is one of the primary activities of today's web users. For a majority of users results returned by conventional search engines are an essentially complete set of links to all pages on the Web relevant to their queries. However, current-day searchers do not crawl and index a significant portion of the Web and, hence, web users relying on search engines only are unable to discover and access a large amount of information from the non-indexable part of the Web. Specifically, dynamic pages generated based on parameters provided by a user via web search forms are not indexed by search engines and cannot be found in searchers' results. Such search interfaces provide web users with an online access to myriads of databases on the Web. In order to obtain some information from a web database of interest, a user issues his/her query by specifying query terms in a search form and receives the query results, a set of dynamic pages which embed required information from a database. At the same time, issuing a query via an arbitrary search interface is an extremely complex task for any kind of automatic agents including web crawlers, which, at least up to the present day, do not even attempt to pass through web forms on a large scale.

Content provided by many web databases is often of very high quality and can be extremely valuable to many users. For example, the PubMed database (http://www.pubmed.gov) allows a user to search through millions of high-quality peer-reviewed papers on biomedical research, while the AutoTrader car classifieds database at http://autotrader.com is highly useful for anyone wishing to buy or sell a car. In general, since each

searchable database is a collection of data in a specific domain it can often provide more specific and detailed information that is not available or hard to find in the indexable Web. The following section provides background information on the non-indexable Web and web databases.

BACKGROUND

Conventional web search engines index only a portion of the Web, called the publicly indexable Web, which consists of publicly available web pages reachable by following hyperlinks.

Figure 1. Indexable and non-indexable portions of the Web and deep Web

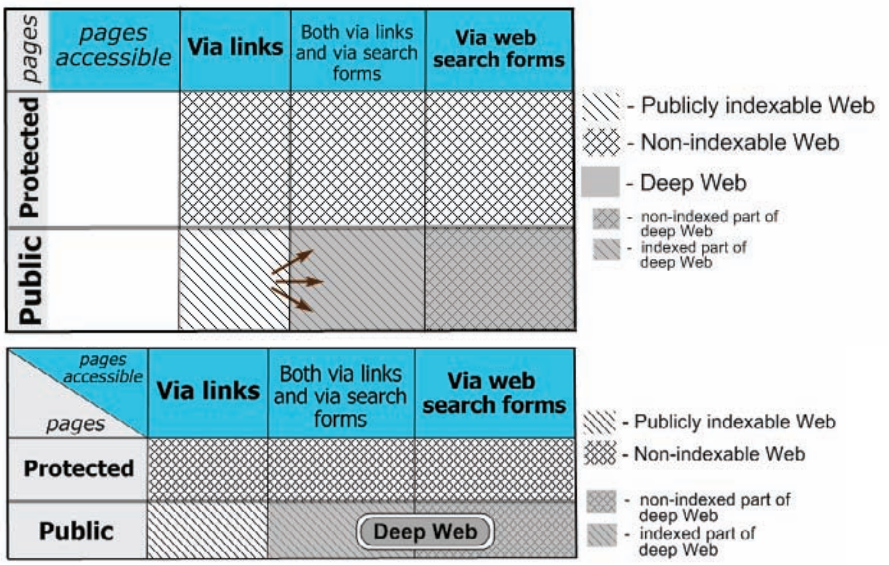
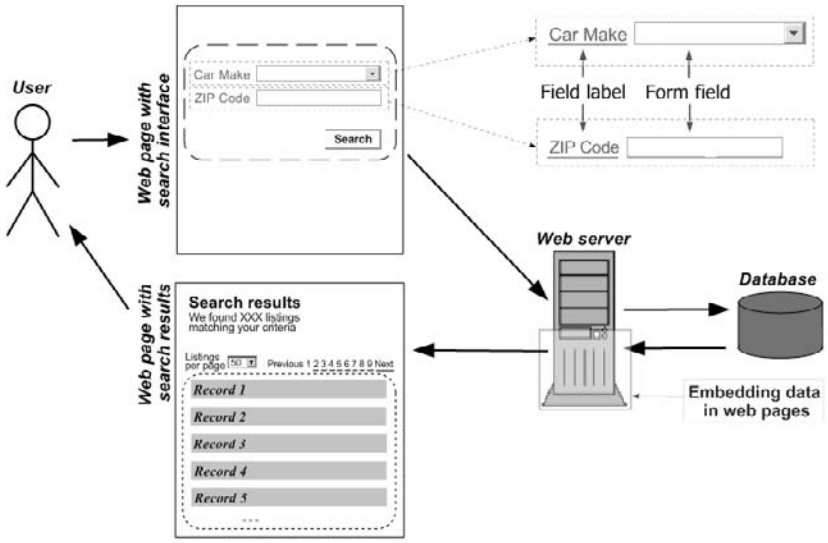


Figure 2. User interaction with web database



The rest of the Web known as the non-indexable Web can be roughly divided into two large parts. The first is the part consisting of protected web pages, i.e., pages that are password-protected, marked as non-indexable by webmasters using the Robots Exclusion Protocol, or only available when visited from certain networks (i.e., corporate intranets). Web pages accessible via web search forms (or search interfaces) comprise the second part of non-indexable Web known as the deep Web (Bergman, 2001) or the hidden Web (Florescu, Levy & Mendelzon, 1998). The deep Web is not completely unknown to search engines: some web databases can be accessed not only through web forms but also through link-based navigational interfaces (e.g., a typical shopping site usually allows browsing products in some subject hierarchy). Thus, a certain part of the deep Web is, in fact, indexable as web searchers can technically reach content of those databases with browse interfaces. Separation into indexable and non-indexable portions of the Web is summarized in Figure 1. The deep Web shown in gray is formed by those public pages that are accessible via web search forms.

A user interaction with a web database is depicted schematically in Figure 2. A web user queries a database via a search interface located on a web page. First, search conditions are specified in a form and then submitted to a web server. Middleware (often called a server-side script) running on a web server processes a user's query by transforming it into a proper format and passing to an underlying database. Second, a server-side script generates a resulting web page by embedding results returned by a database into page template and, finally, a web server sends the generated page with results back to a user. Frequently, results do not fit on one page and, therefore, the page returned to a user contains some sort of navigation through the other pages with results. The resulting pages are often termed data-rich or data-intensive (Crescenzi, Mecca & Merialdo, 2001).

While unstructured content is dominating on the publicly available Web the deep Web is a huge source of structured data. Web databases can be broadly classified into two types: 1) structured web databases, that provide data objects as structured relational-like records with attribute-value pairs, and 2) unstructured web databases, that provide data objects as unstructured media (e.g., text, images, audio, video) (Chang et al., 2004). For example, being a collection of research paper abstracts the PubMed database mentioned earlier is considered to be an unstructured database, while autotrader.com has a structured database for car classifieds, which returns structured car descriptions (e.g., make = "Toyota", model = "Corolla", price = $5000). According to the recent survey of Chang et al. (2004), structured databases dominate in the deep Web – approximately four in five web databases are structured. The distinction between unstructured and structured databases is meaningful since the unstructured content provided by web databases of the first type can be adequately well handled by web searchers. The structured content is much different in this respect as to be fully utilized it should be first extracted from data-intensive pages and then processed in a way similar to instances of a traditional database.

CHALLENGES OF DEEP WEB

Deep Web Characterization

Though the term deep Web was coined in 2000 (Bergman, 2001), which is sufficiently long ago for any web-related concept/technology, many important characteristics of the deep Web are still unknown. For instance, the total size of the deep Web (i.e., how much information is stored in web databases) may only be guessed. At the same time, the existing estimates for the total number of deep web sites are considered to be reliable and consistent. It has been estimated that 43,000 - 96,000 deep web sites existed in March

2000 (Bergman, 2001). A 3-7 times increase in the four years has been reported in the survey (Chang et al., 2004), where the total numbers for deep web sites and web databases in April 2004 have been estimated as around 310,000 and 450,000 correspondingly. Moreover, even national- or domain-specific subsets of deep web resources are, in fact, very large collections. For example, in summer 2005 there were around 7,300 deep web sites on the Russian part of the Web only (Shestakov & Salakoski, 2007). Or the bioinformatics community alone has been maintaining almost one thousand molecular biology databases freely available on the Web (Galperin, 2007).

As mentioned earlier, the deep Web is partially indexed by web search engines because the content of some web databases is accessible both via hyperlinks and search interfaces. According to the rough estimates obtained based on the analysis of 80 databases in eight domains (Chang et al., 2004), around 25% of deep web data is covered by Google (http://google.com). At the same time, most deep web data indexed by Google is out-of-date, i.e., pages returned by Google contain out-of-date information – only each fifth indexed page is fresh.

Querying Web Databases

The Hidden Web Exposer (HiWE) (Raghavan & Garcia-Molina, 2001) is an example of the deep web crawler, i.e., a system that provides a web crawler the ability to automatically fill out search interfaces. Each time when the HiWE finds a web page with a search interface, it performs three steps as follows:

1. Constructs a logical representation of a search form by associating a field label (descriptive text that helps a user to understand the semantics of a form field) to each form field. For example, as shown in Figure 2, *"Car Make"* is a field label for the upper form field.
2. Takes values from a pre-existing value set (initially provided by a user) and submits the form by assigning the most relevant values to form fields.
3. Analyzes the response and, if found valid, retrieves the resulting pages and stores them in the repository (pages in which can be then indexed to support user queries, just like a conventional search engine).

One of the major challenges for deep web crawlers such as the HiWE as well as for any other automatic agents that query web databases is understanding semantics of search interface. Web search forms are created and designed for human beings, but not for computer applications. Consequently, such simple task (from the position of a user) as filling out a search form turns out to be a computationally hard problem. Recognition of field labels that describe fields of a search interface (see Figure 2) is a challenging task since there are myriads of web coding practices and, thus, no common rules regarding the relative position of two elements, that declare a field label and its corresponding field, in the HTML code of a web page. The HiWE introduces the Layout-based Information Extraction Technique (LITE) (Raghavan & Garcia-Molina, 2001), where in addition to the code of a page, the physical layout of a page is also used to aid in extraction. LITE, using the layout engine, identifies the pieces of text, if any, that are physically closest to the form field, in the horizontal and vertical directions. These pieces of text (potential field labels) are then ranked based on several heuristics and the highest ranked candidate is considered to be a field label associated with the form field.

The different approach to automatically issuing queries to textual (unstructured) database was presented by Callan and Connell (2001). The method called query-based sampling exploits result from a first query to extract new query terms, which are then submitted, and so on iteratively. The approach has been applied with some success to siphon data hidden behind keyword-based search interfaces (Barbosa & Freire, 2004). Ntoulas et

al. (2005) have proposed an adaptive algorithm to select the best keyword queries, namely keyword queries that can return a large fraction of database content. Particularly, it has been shown that with only 83 queries, it is possible to download almost 80% of all documents stored in the PubMed database. Query selection for crawling of structured web databases was studied by Wu et al. (2006), who modeled a structured database as an attribute-value graph. Then, the query-based web database crawling is transformed into a graph traversal activity, which is parallel to the traditional problem of crawling the publicly indexable Web following hyperlinks.

Although the HiWE provides a very effective approach to crawl the deep Web, it does not extract data from the resulting pages. The DeLa (Data Extraction and Label Assignment) system (Wang & Lochovsky, 2003) sends queries via search forms, automatically extracts data objects from the resulting pages, fits the extracted data into a table and assigns labels to the attributes of data objects, i.e., columns of the table. The DeLa adopts the HiWE to detect the field labels of search interfaces and to fill out search interfaces. The data extraction and label assignment approaches are based on two main observations. First, data objects embedded in the resulting pages share a common tag structure and they are listed continuously in the web pages. Thus, regular expression wrappers can be automatically generated to extract such data objects and fit them into a table. Second, a search interface, through which web users issue their queries, gives a sketch of (part of) the underlying database. Based on this, extracted field labels are matched to the columns of the table, thereby annotating the attributes of the extracted data.

The HiWE system (and hence the DeLa system) works with relatively simple search interfaces. Besides, it does not consider the navigational complexity of many deep web sites. Particularly, resulting pages may contain links to other web pages with relevant information and consequently it is necessary to navigate these links for evaluation of their relevancies. Additionally, obtaining pages with results sometimes requires two or even more successful form submissions (as a rule, the second form is returned to refine/modify search conditions provided in the previous form). The DEQUE (Deep Web Query System) (Shestakov, Bhowmick & Lim, 2005) addressed these challenges and proposed a web form query language called DEQUEL that allows a user to assign more values at a time to the search interface's fields than it is possible when the interface is manually filled out. For example, data from relational tables or data obtained from querying other web databases can be used as input values. The data extraction from resulting pages in the DEQUE is based on a modified version of the RoadRunner technique (Crescenzi, Mecca & Merialdo, 2001).

Among research efforts solely devoted to information extraction from resulting pages, Caverlee et al. (2005) presented the Thor framework for sampling, locating, and partitioning the query-related content regions (called Query-Answer Pagelets or QA-Pagelets) from the deep Web. The Thor is designed as a suite of algorithms to support the entire scope of the deep Web information platform. The Thor data extraction and preparation subsystem supports a robust approach to the sampling, locating, and partitioning of the QA-Pagelets in four phases: 1) the first phase probes web databases to sample a set of resulting pages that covers all possible structurally diverse resulting pages, such as pages with multiple records, single record, and no record; 2) the second phase uses a page clustering algorithm to segment resulting pages according to their control-flow dependent similarities, aiming at separating pages that contain query matches from pages that contain no matches; 3) in the third phase, pages from top-ranked clusters are examined at a subtree level to locate and rank the query-related regions of the pages; and 4) the final phase uses a set of partitioning algorithms to separate and locate itemized objects in each QA-Pagelet.

Searching for Web Databases

Previous subsection discussed how to query web databases assuming that search interfaces to web databases of interest were already discovered. Surprisingly, finding of search interfaces to web databases is a challenging problem in itself. Indeed, since several hundred thousands of databases are available on the Web (Chang et al., 2004) one cannot be aware that he/she knows the most relevant databases even in a very specific domain. Realizing that, people have started to manually create web database collections such as the Molecular Biology Database Collection (Galperin, 2007). However, manual approaches are not practical as there are hundreds of thousands databases. Besides, since new databases are constantly being added, the freshness of a manually maintained collection is highly compromised.

Barbosa and Freire (2005) proposed a new crawling strategy to automatically locate web databases. They built a form-focused crawler that uses three classifiers: page classifier (classifies pages as belonging to topics in taxonomy), link classifier (identifies links that are likely to lead to the pages with search interfaces in one or more steps), and form classifier. The form classifier is a domain-independent binary classifier that uses a decision tree to determine whether a web form is searchable or non-searchable (e.g., forms for login, registration, leaving comments, discussion group interfaces, mailing list subscriptions, purchase forms, etc.).

FUTURE TRENDS

Typically, search interface and results of search are located on two different web pages (as depicted in Figure 2). However, due to advances of web technologies, a web server may send back only query results (in a specific format) rather than a whole generated page with results (Garrett, 2005). In this case, a page is populated with results on the client-side: specifically, a client-side script embedded within a page with search form receives the data from a web server and modifies the page content by inserting the received data. Ever-growing use of client-side scripts will pose a technical challenge (Alvarez et al., 2004) since such scripts can modify or constrain the behavior of a search form and, moreover, can be responsible for generation of resulting pages. Another technical challenge is dealing with non-HTML search interfaces such as interfaces implemented as Java applets or in Flash (Adobe Flash). It is worth to mention here that all approaches described in the previous section work with HTML-forms only and do not take into account client-side scripts.

Web search forms are user-friendly interfaces to web databases rather than program-friendly. However, more and more web databases begin to provide APIs (i.e., program-friendly interfaces) to their content. The prevalence of APIs will solve most of the problems related to the querying of databases and, at the same time, will make the problem of integrating deep Web data a very active area of research. For instance, one of the challenges we will face is how to select only a few databases, which content is the most appropriate for a given user's query, among a large pool of databases. Nevertheless, one can expect merely a gradual migration towards access web databases via APIs and, thus, at least for recent years, many web databases will still be accessible only through user-friendly web forms.

CONCLUSION

The existence and continued growth of the deep Web creates new challenges for web search engines, which are attempting to make all content on the Web easily accessible by all users (Madhavan et al., 2006). Though much research has emerged in recent years on querying web databases, there is still a great deal of work to be done. The deep Web will require more effective access techniques

since the traditional crawl-and-index techniques, which have been quite successful for unstructured web pages in the publicly indexable Web, may not be appropriate for mostly structured data in the deep Web. Thus, a new field of research combining methods and research efforts of database and information retrieval communities may be created.

REFERENCES

Alvarez, M., Pan, A., Raposo, J., & Vina, J. (2004). Client-side deep web data extraction. *IEEE Conference on e-Commerce Technology for Dynamic E-Business (CEC-East'04),* 158-161.

Barbosa, L., & Freire, J. (2004). Siphoning hidden-web data through keyword-based interfaces. *19th Brazilian Symposium on Data Base (SBBD'04),* 309-321.

Barbosa, L., & Freire, J. (2005). Searching for hidden-web databases. *8th International Workshop on the Web and Databases (WebDB'05),* 1-6.

Bergman, M. (2001). The deep Web: surfacing hidden value. *Journal of Electronic Publishing, 7*(1).

Callan, J., & Connell, M. (2001). Query-based sampling of text databases. *ACM Transactions on Information Systems (TOIS), 19*(2), 97-130.

Caverlee, J., & Liu, L. (2005). QA-Pagelet: data preparation techniques for large-scale data analysis of the deep Web. *IEEE Transactions on Knowledge and Data Engineering, 17*(9), 1247-1262.

Chang, K., He, B., Li, C., Patel, M., & Zhang, Z. (2004). Structured databases on the Web: observations and implications. *SIGMOD Rec., 33*(3), 61-70.

Crescenzi, V., Mecca, G., & Merialdo, P. (2001). RoadRunner: towards automatic data extraction from large web sites. *27th International Conference on Very Large Data Bases (VLDB'01),* 109-118.

Florescu, D., Levy, A., & Mendelzon, A. (1998). Database techniques for the World-Wide Web: a survey. *SIGMOD Rec., 27*(3), 59-74.

Galperin, M. (2007). The molecular biology database collection: 2007 update. *Nucleic Acids Research, 35*(Database-Issue), 3-4.

Garrett, J. (2005). Ajax: a new approach to web applications. AdaptivePath. Retrieved October 8, 2007, from *http://www.adaptivepath.com/publications/essays/archives/000385.php*

Madhavan, J., Halevy, A., Cohen, S., Dong, X., Jeffery, S., Ko, D., & Yu, C. (2006). Structured data meets the Web: a few observations. *IEEE Date Engineering Bulletin, 29*(4), 19-26.

Ntoulas, A., Zerfos, P., & Cho, J. (2005). Downloading textual hidden web content through keyword queries. *5th ACM/IEEE-CS Joint Conference on Digital Libraries (JCDL'05),* 100-109.

Raghavan, S., & Garcia-Molina, H. (2001). Crawling the hidden Web. *27th International Conference on Very Large Data Bases (VLDB'01),* 129-138.

Shestakov, D., Bhowmick, S., & Lim, E.-P. (2005). DEQUE: querying the deep Web. *Data&Knowledge Engineering, 52*(3), 273-311.

Shestakov, D., & Salakoski, T. (2007). On estimating the scale of national deep Web. *18th International Conference on Database and Expert Systems Applications (DEXA'07),* 780-789.

Wang, J., & Lochovsky, F. (2003). Data extraction and label assignment for web databases. *12th International Conference on* World Wide Web *(WWW'03),* 187-196.

Wu, P., Wen, J.-R., Liu, H., & Ma, W.-Y. (2006). Query selection techniques for efficient crawling of structured web sources. 22nd International Conference on Data Engineering (ICDE'06), 47.

Adobe Flash, Retrieved October 8, 2007, from *http://www.adobe.com/products/flash/*

KEY TERMS

Deep Web (also hidden Web): The part of the Web that consists of all web pages accessible via search interfaces.

Deep Web Site: A web site that provides an access via search interface to one or more back-end web databases.

Non-Indexable Web (also invisible Web): The part of the Web that is not indexed by the general-purpose web search engines.

Publicly Indexable Web: The part of the Web that is indexed by the major web search engines.

Search Interface (also search form): A user-friendly interface located on a web site that allows a user to issue queries to a database.

Web Database: A database accessible online via at least one search interface.

Web Crawler: An automated program used by search engines to collect web pages to be indexed.

Chapter LXIII
Learning Classifiers from Distributed Data Sources

Doina Caragea
Kansas State University, USA

Vasant Honavar
Iowa State University, USA

INTRODUCTION

Recent development of high throughput data acquisition technologies in a number of domains (e.g., biological sciences, atmospheric sciences, space sciences, commerce) together with advances in digital storage, computing, and communications technologies have resulted in the proliferation of a multitude of physically distributed data repositories created and maintained by autonomous entities (e.g., scientists, organizations). The resulting increasingly data-rich domains offer unprecedented opportunities in computer assisted data-driven knowledge acquisition in a number of applications, including, in particular, data-driven scientific discovery, data-driven decision-making in business and commerce, monitoring and control of complex systems, and security informatics.

Machine learning (Duda, Hart & Stork, 2000; Mitchell, 1997) offers one of the most cost-effective approaches to analyzing, exploring, and extracting knowledge (i.e., features, correlations, and other complex relationships and hypotheses that describe potentially interesting regularities) from data. However, the applicability of current machine learning approaches in emerging data-rich applications is severely limited by a number of factors:

a. Data repositories are large in size, dynamic, and physically distributed. Consequently, it is neither desirable nor feasible to gather all of the data in a centralized location for analysis. Hence, there is a need for efficient

algorithms for analyzing and exploring multiple distributed data sources without transmitting large amounts of data.

b. Autonomously developed and operated data sources often differ in their structures and organizations (e.g., relational databases, flat files, etc.) and the operations that can be performed on the data sources (e.g., types of queries—relational queries, statistical queries, keyword matches). Hence, there is a need for theoretically well-founded strategies for efficiently obtaining the information needed for analysis within the operational constraints imposed by the data sources.

The purpose of this entry is to precisely define the problem of learning classifiers from distributed data and summarize recent advances that have led to a solution to this problem (Caragea, Silvescu & Honavar, 2004; Caragea, Zhang, Bao, Pathak & Honavar, 2005).

BACKGROUND: PROBLEM SPECIFICATION

Given a dataset D, a hypothesis class H, and a performance criterion P, an algorithm L for learning (from centralized data D) outputs a hypothesis $h \in H$ that optimizes P. In pattern classification applications, h is a classifier (e.g., a decision tree, a support vector machine, etc.) (see Figure 1). Data D typically consist of a set of training examples. Each training example is an ordered tuple of attribute values where one of the attributes corresponds to a class label and the remaining attributes represent inputs to the classifier. The goal of learning is to produce a hypothesis that optimizes the performance criterion (e.g., minimizes classification error on the training data) and the complexity of the hypothesis.

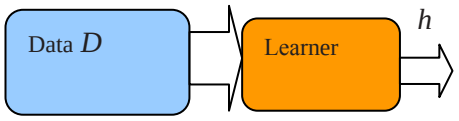
In a distributed setting, a dataset D is distributed among the sites 1,...,n containing the dataset fragments D_1,...,D_n. Two common types of data fragmentation are *horizontal fragmentation* and *vertical fragmentation*. In the case of horizontal fragmentation, each site contains a subset of data tuples that make up D, *for example,*

$$(D = \bigcup_{i=1}^{n} D_i).$$

In the case of vertical fragmentation, each site stores the subtuples of data tuples (corresponding to a subset of the attributes used to define data tuples in D). In this case, D can be constructed by taking the *join* of the individual datasets D_1,...,D_n (assuming a unique identifier for each data tuple is stored with the corresponding subtuples). More generally, the data may be fragmented into a set of relations, as in the case of tables of a relational database, but distributed across multiple sites (i.e., $D = \otimes_{i=1}^{n} D_i$), where $\otimes$ denotes the *join* operation (Friedman, Getoor, Koller & Pfeffer, 1999; Ozsu & Valduriez, 1999). If a dataset D is distributed among the sites 1,…,n containing dataset fragments D_1,..., D_n, we assume that the individual datasets D_1,...,D_n collectively contain (in principle) all the information needed to construct dataset D.

The distributed setting typically imposes a set of constraints Z on the learner (absent in the centralized setting). For example, the constraints Z may prohibit the transfer of raw data from each of the sites to a central location, while allowing the learner to obtain certain types of statistics from the individual sites (e.g., counts of instances that have specified values for some subset of attributes). In some applications of data mining (e.g., knowledge discovery from clinical records), Z might include constraints designed to preserve privacy.

Figure 1. Learning from centralized data



The problem of learning from distributed data can be stated as follows (Caragea et al., 2004; 2005): Given the fragments $D_1,\ldots, D_n$ of a dataset D distributed across the sites $1,\ldots,n$, a set of constraints Z, a hypothesis class H, and a performance criterion P, the task of a distributed learner L_d is to output a hypothesis that optimizes P using only operations allowed by Z. Clearly, the problem of learning from a centralized dataset D is a special case of learning from distributed data where n=1 and Z=∅.

Having defined the problem of learning from distributed data, we proceed to define some criteria that can be used to evaluate the quality of the hypothesis produced by an algorithm L_d for learning from distributed data relative to its centralized counterpart. We say that an algorithm L_d for learning from distributed data sets $D_1, \ldots, D_n$ is *exact* relative to its centralized counterpart L if the hypothesis produced by L_d is identical to that obtained by L from dataset D obtained by appropriately combining datasets $D_1,\ldots, D_n$.

The proof of exactness of an algorithm for learning from distributed data relative to its centralized counterpart ensures that a large collection of existing theoretical (e.g., sample complexity, error bounds) as well as empirical results obtained in the centralized setting apply in the distributed setting.

MAIN THRUST: STRATEGY FOR LEARNING FROM DISTRIBUTED DATA

Decomposition of the Learning from Data Task

A general strategy for designing algorithms for learning from distributed data that are provably exact with respect to their centralized counterpart follows from the observation that most of the learning algorithms use only certain statistics computed from data D in the process of generating the hypotheses that they output. (Recall that a statistic is simply a function of the data; examples of statistics include mean value of an attribute, counts of instances that have specified values for some subset of attributes, the most frequent value of an attribute, etc.). This yields a natural decomposition of a learning algorithm into two components (see Figure 2):

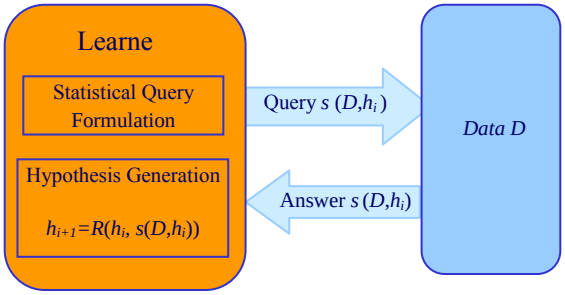
a. An information extraction component that formulates and sends a statistical query to a data source
b. A hypothesis generation component that uses the resulting statistic to modify a partially constructed hypothesis (and it may further invoke the information extraction component as needed).

Sufficient Statistics for Learning

A statistic $s(D)$ is called a sufficient statistic for a parameter θ if $s(D)$, loosely speaking, provides all the information needed for estimating the parameter from data D (Casella & Berger, 2001). Thus, sample mean is a sufficient statistic for the mean of a Gaussian distribution.

This notion of a sufficient statistic for a parameter θ can be generalized to yield a sufficient statistic $s_{L,h}(D)$ for learning a hypothesis h using a learning algorithm L applied to a dataset D (Caragea et al., 2004). Trivially, data D is a sufficient

Figure 2. Learning = Statistical Query Answering & Hypothesis Generation



statistic for learning h using L. However, we are typically interested in statistics that are minimal or, at the very least, substantially smaller in size (in terms of the number of bits needed for encoding) than dataset D. In some simple cases, it is possible to extract a sufficient statistic $s_{L,h}(D)$ for constructing a hypothesis h in one step (e.g., by querying the data source for a set of conditional probability estimates when L is the standard algorithm for learning a Naive Bayes classifier). In such a case, we say that $s_{L,h}(D)$ is a sufficient statistic for learning h using the learning algorithm L if there exists an algorithm that accepts $s_{L,h}(D)$ as input and outputs $h=L(D)$. In a more general setting, h is constructed by L by interleaving information extraction and hypothesis generation operations. Thus, a decision tree learning algorithm would start with an empty initial hypothesis h_0, obtain the sufficient statistics (expected information concerning the class membership of an instance associated with each of the attributes) for the root of the decision tree (a partial hypothesis h_1), and recursively generate queries for additional statistics needed to iteratively refine h_1 to obtain a succession of partial hypotheses $h_1,h_2,\ldots$ culminating in h (see Figure 2).

We say that $s(D, h_i)$ is a sufficient statistic for the *refinement* of a hypothesis h_i into h_{i+1} (denoted by $s_{h_i \to h_{i+1}}$) if there exists an algorithm R that accepts h_i and $s(D, h_i)$ as inputs and outputs h_{i+1}. We say that $s_h(D, h_i,...h_m)$ is a sufficient statistic for the *composition* of the hypotheses $(h_i,...h_m)$ into h (denoted by $s_{(h_1,...h_m)\to h}$) if there exists an algorithm C that accepts as inputs $h_i,...h_m$ and $s_h(D, h_i,...h_m)$ and outputs the hypothesis h. We say that $s_{h_i \to h_{i+k}}$ (where $i \geq 0$ and $k > 0$ are positive integers) is a sufficient statistic for iteratively refining a hypothesis h_i into h_{i+k} if h_{i+k} can be obtained through a sequence of refinements starting with h_i. We say that $s_{(h_1,...h_m)\to h}$ is a sufficient statistic for obtaining hypothesis h starting with hypotheses $h_1,...h_m$ if h can be obtained from $h_1,...h_m$ through some sequence of applications of composition and refinement operations.

Assuming that the relevant sufficient statistics (and the procedures for computing them) can be defined, the application of a learning algorithm L to a dataset D can be reduced to the computation of $s_{(h_0,...h_m)\to h}$ through some sequence of applications of hypothesis refinement and composition operations starting with the hypothesis h (see Figure 2). In this model, the only interaction of the learner with the data repository is through queries for the relevant statistics.

Information Extraction from Distributed Data

Based on the decomposition of the learning task into information extraction and hypothesis generation, learning from distributed data reduces to information extraction from distributed data. The information extraction from distributed data entails decomposing each statistical query q posed by the information extraction component of the learner into subqueries $q_1,...,q_n$ that can be answered by the individual data sources $D_1,...,D_n$ respectively, and a procedure for combining the answers to the subqueries into an answer for the original query q (see Figure 3).

It is important to note that this general strategy for learning classifiers from distributed data is applicable to the entire class of algorithms for learning classifiers from data. This follows from the fact that the output h of any learning algorithm is, in fact, a function of data D, and hence by definition, a *statistic*. Consequently, we can devise a strategy for computing h from data D through some combination of refinement and composition operations starting with an initial hypothesis (or an initial set of hypotheses). When data D is stored in one or more relational database(s), this approach is able to effectively use the database support for aggregate queries for efficient data mining. When the learner's access to data sources is subject to constraints Z, the resulting plan for information extraction has to be executable without violating the constraints Z. The *exactness*

of the algorithm L_d for learning from distributed data relative to its centralized counterpart, which requires access to the complete dataset D, follows from the correctness (soundness) of the query decomposition and answer composition procedure. The separation of concerns between hypothesis construction and extraction of sufficient statistics from data makes it possible to explore the use of sophisticated techniques for query optimization. These techniques yield optimal plans for gathering sufficient statistics from distributed data sources under a specified set of constraints. The constraints describe the query capabilities of the data sources; operations permitted by the data sources (e.g., execution of user-supplied procedures); and available computation, bandwidth, and memory resources.

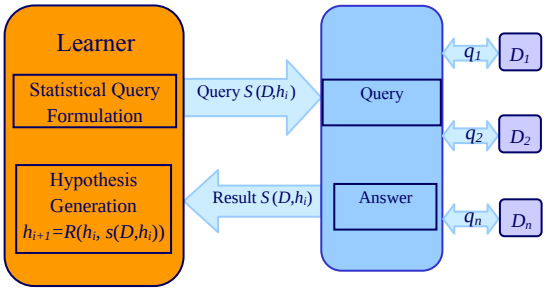
Related Work

Srivastava, Han, Kumar, and Singh (1999) propose methods for distributing a large centralized dataset to multiple processors to exploit parallel processing to speed up learning. Grossman and Gou (2001), and Provost and Kolluri (1999) survey several methods that exploit parallel processing for scaling up data mining algorithms to work with large datasets. In contrast, the focus of the proposed research is on learning classifiers from a set of autonomous distributed data sources. The autonomous nature of the data sources implies that the learner has little control over the manner in which the data are distributed among the different sources.

Distributed data mining has received considerable attention in the literature (Park & Kargupta, 2002). Domingos (1997) and Prodromidis, Chan, and Stolfo (2000) propose an *ensemble of classifiers* approach to learning from horizontally fragmented distributed data, which essentially involves learning separate classifiers from each dataset and combining them, typically using a weighted voting scheme. A potential drawback of the ensemble of classifiers approach to learning from distributed data is that the resulting ensemble of classifiers is typically much harder to comprehend than a single classifier. Another important limitation of the ensemble classifier approach to learning from distributed data is the lack of guarantees concerning generalization accuracy of the resulting hypothesis relative to the hypothesis obtained in the centralized setting.

The approach described here offers a general framework for the design of algorithms for learning from distributed data, which is provably exact with respect to its centralized counterpart. Central to this approach is a clear separation of concerns between hypothesis construction and extraction of sufficient statistics from data, making it possible to explore the use of sophisticated techniques for query optimization.

Figure 3. Learning from distributed data = statistical query answering + hypothesis generation



FUTURE TRENDS

It is inevitable that autonomously developed data sources are semantically heterogeneous. The ontological commitments associated with a data source (and hence its implied semantics) are typically determined by the data source designers, based on their understanding of the intended use of the data. Very often, data sources that are created for use in one context or application find use in other contexts or applications, and therefore, seman-

tic differences among autonomously designed, owned, and operated data repositories are simply unavoidable. Effective use of multiple sources of data in a given context requires reconciliation of semantic differences. The approach described here lends itself to adaptation to settings where the ontologies associated with the individual data sources differ from each other and to settings that require learning predictive models from partially specified data (Caragea et al., 2005).

CONCLUSION

With the proliferation of large, autonomous, distributed databases in many application domains, the problem of mining distributed data is an extremely important topic with relevance to many application domains where databases play a central role. Effective solutions to this problem are critical to advances in cyberinfrastructure for e-science (Hey & Trefethen, 2005). In this entry, we have precisely formulated the problem of learning classifiers from distributed data and described a general strategy for transforming standard machine learning algorithms that assume centralized access to data in a single location into algorithms for learning from distributed data. The resulting algorithms are *provably exact* in that the hypothesis constructed from distributed data is identical to that obtained by the corresponding algorithm when it is used in the centralized setting. This ensures that the entire body of theoretical (e.g., sample complexity, error bounds) and empirical results obtained in the centralized setting carry over to the distributed setting. The approach described can be extended to scenarios where the distributed data sources are semantically heterogeneous (Caragea et al., 2005).

ACKNOWLEDGMENT

This work is supported by the National Science Foundation Grants IIS 0219699 and IIS 0711396.

REFERENCES

Caragea, D., Silvescu, A., & Honavar, V. (2004). A framework for learning from distributed data using sufficient statistics and its application to learning decision trees. *International Journal of Hybrid Intelligent Systems, 1,* 80–89.

Caragea, D., Zhang, J., Bao, J., Pathak, J., & Honavar, V. (2005). *Algorithms and software for collaborative discovery from autonomous, semantically heterogeneous, distributed information sources.* Proceedings of the Conference on Algorithmic Learning Theory, LNCS, 3734, 13–44.

Casella, G., & Berger, R.L. (2001). *Statistical inference*. Belmont, CA: Duxbury Press.

Domingos, P. (1997). Knowledge acquisition from examples via multiple models. Proceedings of the Fourteenth International Conference on Machine Learning, Nashville, Tennessee, 98–106.

Duda, R., Hart, E., & Stork, D. (2000). *Pattern recognition*. New York: Wiley.

Friedman, N., Getoor, L., Koller, D., & Pfeffer, A. (1999). *Learning probabilistic relational models.* Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence, Orlando, Florida, 1300–1309.

Grossman, L.R., & Gou, Y. (2001). Parallel methods for scaling data mining algorithms to large data sets. In J.M. Zytkow (Ed.), *Handbook on data mining and knowledge discovery.* Oxford University Press.

Hey, T., & Trefethen, A.E. (2005). Cyberinfrastructure for e-science. *Science, 308*(5723), 817–821.

Mitchell, T. (1997). *Machine learning.* New York: Addison-Wesley.

Ozsu, M.T., & Valduriez, P. (1999). *Principles of distributed database systems* (2nd Edition). Prentice Hall, Inc.

Park, B., & Kargupta, H. (2002). Distributed data mining: Algorithms, systems, and applications. In Nong Ye (Ed.), *Data mining handbook* (pp. 341–358). IEA.

Prodromidis, A.L., Chan, P., & Stolfo, S.J. (2000). Meta-learning in distributed data mining systems: Issues and approaches. In H. Kargupta, & P. Chan (Eds.), *Advances of distributed data mining.* AAAI Press.

Provost, F.J., & Kolluri, V. (1999). A survey of methods for scaling up inductive algorithms. *Data Mining and Knowledge Discovery, 3*(2), 131–169.

Srivastava, A., Han, E., Kumar, V., & Singh, V. (1999). Parallel formulations of decision-tree classification algorithms. *Data Mining and Knowledge Discovery, 3*(3), 237–261.

KEY TERMS

Classification Task: A task for which the learner is given experience in the form of labeled examples and is supposed to learn to classify new unlabeled examples. In a classification task, the output of the learning algorithm is called hypothesis or classifier (e.g., a decision tree, a support vector machine, etc.).

Distributed Data Sources: In a distributed setting, the data are distributed across several data sources. Each data source contains only a fragment of the data. This leads to a fragmentation of a data. Two common types of data fragmentation are *horizontal fragmentation*, wherein (possibly overlapping) subsets of data tuples are stored at different sites; and *vertical fragmentation*, wherein (possibly overlapping) subtuples of data tuples are stored at different sites. More generally, the data may be fragmented into a set of relations (tables of a relational database, distributed across multiple sites).

Learning from Data: Given a dataset, a hypothesis class (e.g., decision trees), and a performance criterion (e.g., accuracy), a learning algorithm outputs a hypothesis that optimizes the performance criterion.

Learning from Distributed Data: Given several fragments of a dataset (distributed across several sites), a set of constraints (referring to privacy concerns, storage issues, operations allowed, etc.), a hypothesis class, and a performance criterion, the task of the distributed learner is to output a hypothesis that optimizes the performance criterion without violating the set of constraints. We say that an algorithm for learning from distributed data is *exact* if the hypothesis that it outputs is identical to that produced by the centralized algorithm for learning from the complete dataset, obtained by appropriately combining the distributed data fragments.

Learning Task Decomposition: A learning algorithm can be decomposed in two components: (a) an *information extraction* component that formulates and sends a statistical query to a data source; and (b) a *hypothesis generation* component that uses the resulting statistic to modify a partially constructed algorithm output (and further invokes the information extraction component, if needed, to generate the final algorithm output).

Machine Learning: A key objective of Machine Learning is to design and analyze algorithms that are able to improve the performance at some task through experience. A machine learning system is specified by several components: (a)

Learner – an algorithm or a computer program that is able to use the experience to improve its performance; (b) *Task* – a description of the task that the learner is trying to accomplish (e.g., learn a concept, a function, etc.); (c) *Experience* – specification of the information that the learner uses to perform the learning; (d) *Background knowledge* – the information that the learner has about the task before the learning process (e.g., "simple" answers are preferable over "complex" answers); (e) *Performance Criteria* – measure the quality of the learning output in terms of accuracy, simplicity, efficiency, and so forth.

Sufficient Statistics: A statistic is called a sufficient statistic for a parameter if the statistic captures all the information about the parameter, contained in the data. More generally, a statistic is called a sufficient statistic for learning a hypothesis using a particular learning algorithm applied to a given dataset if there exists an algorithm that takes as input the statistic and outputs the desired hypothesis. A query that returns a statistic is called a statistical query.

Chapter LXIV
Differential Learning Expert System in Data Management

Manjunath R.
Bangalore University, India

INTRODUCTION

Expert systems have been applied to many areas of research to handle problems effectively. Designing and implementing an expert system is a difficult job, and it usually takes experimentation and experience to achieve high performance. The important feature of an expert system is that it should be easy to modify. They evolve gradually. This evolutionary or incremental development technique has to be noticed as the dominant methodology in the expert-system area. The simple evolutionary model of an expert system is provided in B. Tomic, J. Jovanovic, & V. Devedzic, 2006.

Knowledge acquisition for expert systems poses many problems. Expert systems depend on a human expert to formulate knowledge in symbolic rules. The user can handle the expert systems by updating the rules through user interfaces (J. Jovanovic, D. Gasevic, V. Devedzic, 2004). However, it is almost impossible for an expert to describe knowledge entirely in the form of rules. An expert system may therefore not be able to diagnose a case that the expert is able to. The question is how to extract experience from a set of examples for the use of expert systems.

Machine-learning algorithms such as "learning from example" claim that they are able to extract knowledge from experience. Symbolic systems as, for example, ID3 (Quinlan, 1983) and version-space (Mitchell, 1982) are capable of learning from examples. Connectionist systems claim to have advantages over these systems in generalization and in handling noisy and incomplete data. For every data set, the rule-based systems have to find a definite diagnosis. Inconsistent data can force symbolic systems into an indefinite state. In connectionist networks, a distributed representation of concepts is used. The interference of different concepts allows networks to generalize A network computes for every input the best output.

Due to this, connectionist networks perform well in handling noisy and incomplete data. They are also able to make a plausible statement about missing components. A system that uses a rule-based expert system with an integrated connectionist network could benefit from the described advantages of connectionist systems. Machine-learning helps towards that end.

BACKGROUND

Maintenance of databases in medium-size and large size organizations is quite involved in terms of dynamic reconfiguration, security, and the changing demands of its applications. Here, compact architecture making use of expert systems is explored to crisply update the database. An architecture with a unique combination of digital signal processing/information theory and database technology is tried. Neuro-fuzzy systems are introduced to learn "if-then-else" rules of expert systems.

Kuo, Wu, and Wang (2000) developed a fuzzy neural network with linguistic teaching signals. The novel feature of the expert system is that it makes use of a large number of previous outputs to generate the present output. Such a system is found to be adaptive and reconfigures fast. The expert system makes use of a learning algorithm based on differential feedback.

The differentially fed learning algorithm (Manjunath & Gurumurthy, 2002) is introduced for learning. The learning error is found to be minimal with differential feedback. Here, a portion of the output is fed back to the input to improve the performance. The differential feedback technique is tried at the system level, making the system behave with the same set of learning properties. Thus, control of an expert system controls the entire system

KNOWLEDGE EXTRACTION FROM DIFFERENTIALLY FED NEURAL NETWORKS

The expert systems are organized in a hierarchical fashion. Each level controls a unique set of databases. Finally, the different expert systems themselves are controlled by a larger expert system. This ensures security of databases and selective permissions to their access, (i.e., some of the data needs to be public and the rest has to be private and protected; a concept borrowed from object-oriented programming). Thus, the master expert system can have access to public information. The neural networks are integrated with a rule-based expert system. The system realizes the automatic acquisition of knowledge out of a set of examples. It enhances the reasoning capabilities of classical expert systems with the ability to generalize and the handle incomplete cases. It uses neural nets with differential feedback algorithms to extract regularities out of case data. A symbolic-rule generator transforms these regularities into rules governing the expert system. The generated rules and the trained neural nets are embedded into the expert system as knowledge bases. In the system diagnosis phase it is possible to use these knowledge bases together with human experts' knowledge bases in order to diagnose an unknown case. Furthermore, the system is able to diagnose and to complete inconsistent data using the trained neural nets exploiting their ability to generalize.

It is required to describe a possible approach for the optimization of the job scheduling in large distributed systems, based on self-organizing neural networks. This dynamic scheduling system should be seen as adaptive middle-layer software, aware of current available resources and making the scheduling decisions using past experience. It aims to optimize job-specific parameters as well as resource utilization. The scheduling system is able to dynamically learn and cluster information in a large dimensional parameter space and

at the same time to explore new regions in the parameter's space. This self-organizing scheduling system may offer a possible solution for providing an effective use of resources for the off-line data-processing jobs.

Finding and optimizing efficient job-scheduling policies in large distributed systems, which evolve dynamically, is a challenging task. It requires the analysis of a large number of parameters describing the jobs and the time-dependent state of the system. In one approach, the job-scheduling task in distributed architectures is based on self-organizing neural networks. The use of these networks enables the scheduling system to dynamically learn and cluster information in a high-dimensional parameter space. This approach may be applied to the problem of distributing off-line data processing. These jobs need random access to very large amounts of data, which are assumed to be organized and managed by distributed federations of OODB (object-oriented database) systems. Such a scheduling system may also help manage the way data are distributed among regional centers as a function of time, making it capable of providing useful information for the establishment and execution of data replication policies. A hybrid combination of neural networks and expert systems was tried by Apolloni, Zamponi, and Zanaboni (2000). Fdez-Riverola and Corchado (2003) used unsupervised learning for prediction of parameters during the learning process.

NEED OF DIFFERENTIAL FEEDBACK

The importance of connectionist model is provided in (Stephen.I.Gallant, 1993) . It provides the exact model of the way the human brain process the information.

In the learning of an expert system, one common issue but a powerful paradigm involves chaining of IF-THEN rules to form a line of reasoning. If the chaining starts from a set of conditions and moves toward some conclusion, the method is called forward chaining. If the conclusion or the goal to be achieved is known in advance, but the path to that conclusion is not known, then reasoning backwards is called for and the method is backward chaining. These problem-solving methods are built into program modules called inference engines or inference procedures that manipulate and use the knowledge in the knowledge base to form a line of reasoning.

However both these processes are relatively slow. A feed back may be used in the problem solving model. The feedback makes the model mixed chaining problem solving model. Here the chaining starts from a set of conditions and moves towards partial or intermediate conclusions. These conclusions are compared with the known conclusions. The error or difference could be an additional condition to proceed further. Higher ordered errors (corresponding to higher order differentials) may be used as in feedback neural network. These errors are generated as the differences of differences. The problem solving may be made very fast as the feed back imparts fast learning in to the system. Other benefits of the feedback would follow.

SYSTEM DESCRIPTION

Case data that are presented to an expert system are usually stored in a case database. A data-transformation module encodes such cases in a suitable way to be learnt by neuronal networks. This module performs as follows:

First, it transforms or preprocesses the data so that the components with equal scopes or a hierarchy is formed. It has been shown that discretisation of data (i.e., preprocessing of data in to several subdivisions) makes the neural network converge faster (Abu Bakar et al., 2003). At the same time, hierarchy in the data can be maintained.

Second, the transformation module encodes the data into a binary input pattern because some neural networks, as, for example, the competitive learning model (Rumelhart & Zipser, 1985) processes only binary inputs. To do this, the intervals of the components are subdivided into different ranges. These ranges are adapted according to the distribution of the components. So, every component of a vector is represented by the range its value belongs to. Depending on the kind of representation, the ranges could be encoded locally.

With the transformed data, different neuronal networks with unsupervised learning algorithms, such as competitive learning (Rumelhart & Zipser, 1985), ART, and Kohonen, are trained. These networks have the ability to adapt their internal structures (i.e., weight matrix) to the structure of the data. In a rule-generation module, the structures learned by the neuronal networks are detected, examined, and transformed into expert systems rules. These rules can be inspected by a human expert and added to an expert system. When a case is presented to the expert system, the system first tries to reason with the rules that have been acquired from the human expert to produce a suitable diagnosis. If this fails to produce a diagnosis, the new rules produced by the process described can be used. If the case can be handled in such a way, all steps of the reasoning process may be inspected by and explained to a user of the system.

If the system, however, is not able to produce a suitable diagnosis in this way, be it, that data is missing, or the input is erroneous or no rule fits the data, since such a case has not been considered while building the knowledge base, the expert system can turn the case over to the networks. The networks, with their ability to associate and generalize, search for a most suiting case that has been learned before. The diagnosis that has been associated with that case is then returned as a possible diagnosis.

PROPOSED ARCHITECTURE

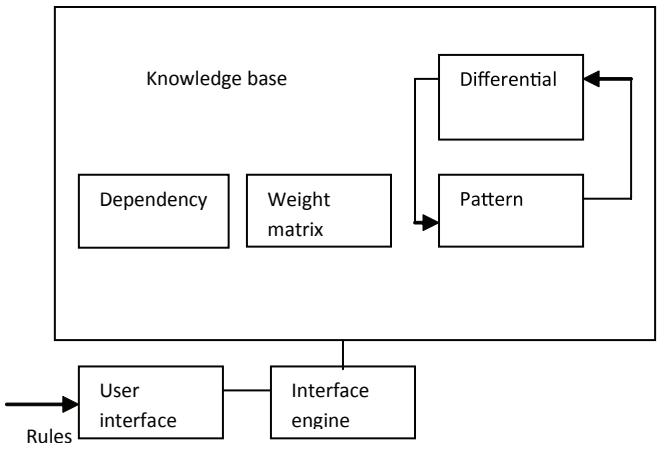
In recent years, neural networks have been extensively used to simulate human behavior in areas such as vision, speech, and pattern recognition. In large scales, they perform the recognition and classification tasks better than human beings. Neural nets can find a relation by making a match between known inputs and outputs in a pool of a large data. The performance of these networks can be improved by providing a differential feedback from the output to the input Expert systems have been used in conjunction with neural network technology to eliminate the time-consuming part of constructing and debugging the knowledge base. Neural networks can be used to recognize patterns, such as financial or sensory data, which are then acted on by the rules of an expert system. An expert system trains the neural network that in turn generates rules for an expert system. The use of an artificial neural network greatly simplifies the knowledge-engineering task because it allows the system itself to construct the knowledge base from a set of training examples. Because the correct output of the expert system is known for the given set of inputs, a supervised learning algorithm is used to train the system.

Compared to rule-based expert systems, connectionist expert systems give a better model of reasoning. These models can be applied to any type of decision problems, especially when the creation of if then rules is not possible, or the information is contradictory.

A connectionist expert system usually contains three major parts: a knowledge base (database), an inference engine, and the user interface. The knowledge base is a problem-dependent part that contains expert knowledge.

The connectionist expert system database is composed of neurons and connections among them. The inference engine is a problem-independent driver program which is responsible for reasoning. The user interface is a link between the inference engine and the external user. The

Figure 1. An expert system with DANN-based learning



architecture of a connectionist expert system is shown in Figure 1. The weight matrix stores the rules that are translated in to the input–output dependency in the neural network through the dependency matrix and the pattern vector. The differential feedback connects a part of the output to the input and reduces the training period as well as the errors in pattern matching.

The training, weight matrix updating and learning can also happen remotely through the internet. The web based learning and diagnosis is discussed in (Chenn-Jung Huang, Ming-Chou Liu, San-Shine Chu , Chih-Lun Cheng, 2007)

DATA PROCESSING

The Kohonen network, consisting of two layers, is used here. The input layer has *n* units representing the n components of a data vector. The output layer is a two dimensional array of units arranged on a grid. The number of the output units is determined experimentally. Each unit in the input layer is connected to every unit in the output layer, with a weight associated. The weights are initialized randomly, taking the smallest and the greatest value of each component of all vectors as boundaries. They are adjusted according to Kohonen's learning rule (Kohonen, 1984). The applied rule uses the Euclidean distance and a simulated Mexican-hat function to realize lateral inhibition. In the output layer, neighboring units form regions, which correspond to similar input vectors. These neighborhoods form disjoint regions, thus classifying the input vectors.

The automatic detection of this classification is difficult because the Kohonen algorithm converges to an equal distribution of the units in the output layer. Therefore, a special algorithm, the so-called U-matrix method, is used to detect classes that are in the data (Ultsch & Siemon, 1990).

In summary, by using this method, structure in the data can be detected as classes. These classes represent sets of data that have something in common.

RULE EXTRACTION

As a first approach to generate rules from the classified data, a well–known, machine-learning algorithm, ID3, is used (Ultsch & Panda, 1991). Although able to generate rules, this algorithm has a serious problem: It uses a minimization criterion that seems to be unnatural for a human expert. Rules are generated that use only a minimal set of decisions to come to a conclusion A large number

of parameters, most of them time dependent, must be used for the job scheduling in large distributed systems. The problem is even more difficult when not all of these parameters are correctly identified, or when the knowledge about the state of the distributed system is incomplete or is known to have a certain delay in the past.

SCHEDULING DECISION

A "classical" scheme to perform job scheduling is based on a set of rules using part of the parameters and a list of empirical constraints based on experience. It may be implemented as a long set of hard coded comparisons to achieve a scheduling decision for each job.

In general, it can be represented as a function, which may depend on large numbers of parameters describing the state of the systems and the jobs. After a job is executed based on this decision, a performance evaluation can be done to quantify the same.

The decision for future jobs should be based on identifying the clusters in the total parameter space, which are close to the hyperplane defined in this space by the subset of parameters describing the job and the state of the system (i.e., parameters known before the job is submitted). In this way, the decision can be made evaluating the typical performances of this list of close clusters and choose a decision set that meets the expected– available performance–resources and cost. However, the self-organizing network has, in the beginning, only a limited "knowledge" in the parameter space, and exploring efficiently other regions is quite a difficult task.

In this approach for scheduling, the difficult part is not learning from previous experience, but making decisions when the system does not know (or never tried) all possible options for a certain state of the system. Even more difficult is to bring the system into a certain load state, which may require a long sequence of successive decisions, such as in a strategic game problem. The result of each decision is seen only after the job is finished, which also adds to the complexity of quantifying the effect of each decision. For this reason, a relatively short history of the previous decisions made by the system is also used in the learning process. A relatively long sequence of the decision (a policy) can be described with a set of a few points of decision history, which partially overlap. This means that a trajectory can be built in the decision space by small segments that partially overlap.

FUTURE TRENDS

The present-day expert systems deal with domains of narrow specialization. For such systems to perform competently over a broad range of tasks, they will have to be given a wealth of knowledge. The next generation expert systems require large knowledge bases. This calls for crisp learning algorithms based on connectionist networks with higher order differential feedback. Improvements in the learning rate and stability of these algorithms to organize large data structures provide a good topic for research.

CONCLUSION

The implementation of the system demonstrates the usefulness of the combination of a rule-based expert system with neural networks (Ultsch & Panda, 1991). Unsupervised-learning neural networks are capable of extracting regularities from data. Due to the distributed sub symbolic representation, neural networks are typically not able to explain inferences. The proposed system avoids this disadvantage by extracting symbolic rules out of the network. The acquired rules can be used like the expert's rules. In particular, it is therefore possible to explain the inferences of the connectionist system.

Such a system is useful in two ways. First, the system is able to learn from examples with a known diagnosis. With this extracted knowledge it is possible to diagnose new unknown examples. Second, the system has the ability to handle a large data set for which a classification or diagnosis is unknown. For such a data set, classification rules are proposed to an expert. The integration of a connectionist module realizes "learning from examples." Furthermore, the system is able to handle noisy and incomplete data. First results show that the combination of a rule-based expert system with a connectionist module is not only feasible but also useful. The system considered is one of the possible ways to combine the advantages of the symbolic and sub symbolic paradigms. It is an example to equip a rule based expert system with the ability to learn from experience using a neural network.

REFERENCES

Abu Bakar, A., et al. (2003). Rough discretisation approach in probability analysis for neural classifier. *ITSIM*.

Apolloni, B., Zamponi, G., & Zanaboni, A. M. (2000). An integrated symbolic/connectionist architecture for parsing Italian sentences containing pp-attachment ambiguities. *Applied Artificial Intelligence, 14*(3), 271-308.

Fdez-Riverola, F., & Corchado, J. M. (2003). Forecasting system for red tides: A hybrid autonomous AI model. *Applied Artificial Intelligence, 17*(10), 955–982.

Gallant, S. I. (1993). *Neural Network Learning and Expert Systems.* MIT press

Huang, C.-J., Liu, M.-C., Chu, S.-S., & Cheng, C.-L. (2007). An intelligent learning diagnosis system for Web-based thematic learning platform. *Computers & Education, 48*(4), 658-679

Jovanovic, J., Gasevic, D., & Devedzic, V. (2004). A GUI for Jess. *Expert Systems With Applications, 26*(4), 625-637

Kohonen. (1984). *Self-organization and associative memory.* Berlin, Germany: Springer-Verlag.

Kuo, R. J., Wu, P. C., & Wang, C. P. (2000). Fuzzy neural networks for learning fuzzy if-then rules. *Applied Artificial Intelligence, 14*(6).

Manjunath, R., & Gurumurthy, K. S. (2002). Information geometry of differentially fed artificial neural networks. *TENCON'02.*

Mitchell, T. M. (1982). Generalization as search. *Artificial Intelligence, 18,* 203-226.

Quinlan, J. R. (1983). Learning efficient classification procedures and their application to chess end-games. In *Machine learning: An artificial intelligence approach* (pp. 463-482).

Rumelhart, D. E., & Zipser, D. (1985). Feature discovery by competitive learning. *Cognitive Science, 9,* 75–112.

Tomic, B., Jovanovic, J., & Devedzic, V. (2006). JavaDON: An Open-Source Expert System Shell. *Expert Systems With Applications, 31*(3), 595-606

Ultsch, A., & Siemon, H. P. (1990). Kohonen's self organizing feature maps for exploratory data analysis. In *Proc. Intern. Neural Networks* (pp. 305-308). Kluwer Academic Press.

Ultsch, A., & Panda, P. G. (1991). Die Kopplung konnektionistischer Modelle mit wissensbasierten Systemen. In *Tagungsband Experteny stemtage Dortmund* (pp. 74-94). VDI Verlag.

KEY TERMS

Artificial Intelligence (AI): A research discipline whose aim is to make computers able to

simulate human abilities, especially the ability to learn. AI is separated as neural net theory, expert systems, robotics, fuzzy control systems, game theory, and so forth.

Connectionist Expert System: Expert systems that use artificial neural networks to develop their knowledge bases and to make inferences are called connectionist expert systems. A classical expert system is defined with IF-THEN rules, explicitly. In a connectionist expert system, training examples are used by employing the generalization capability of a neural network, in which the network is coded in the rules of an expert system. The neural network models depend on the processing elements that are connected through weighted connections. The knowledge in these systems is represented by these weights. The topology of the connections are explicit representations of the rules.

Expert System: An expert system is a computer program that simulates the judgment and behavior of a human or an organization that has expert knowledge and experience in a particular field. Typically, such a system contains a knowledge base containing accumulated experience and a set of rules for applying the knowledge base to each particular situation that is described to the program.

Kohonen Feature Map: It is basically a feed forward/feedback type neural net. Built of an input layer(i.e., the neuron of one layer is connected with each neuron of another layer), called "feature map." The feature map can be one or two dimensional, and each of its neurons is connected to all other neurons on the map. It is mainly used for classification.

Neural Network: A member of a class of software that is "trained" by presenting it with examples of input and the corresponding desired output. Training might be conducted using synthetic data, iterating on the examples until satisfactory depth estimates are obtained. Neural networks are general-purpose programs, which have applications outside potential fields, including almost any problem that can be regarded as pattern recognition in some form.

Supervised Learning: This is performed with feed forward nets where training patterns are composed of an input vector and an output vector that are associated with the input and output nodes, respectively. An input vector is presented at the inputs together with a set of desired responses, one for each node. A forward pass is done and the errors or discrepancies, between the desired and actual response for each node in the output layer, are found. These are then used to determine weight changes in the net according to the prevailing learning rule.

Unsupervised Learning: A specific type of a learning algorithm, especially for self-organizing neural nets such as the Kohonen feature map.

Chapter LXV
Machine Learning as a Commonsense Reasoning Process

Xenia Naidenova
Military Medical Academy, Russia

INTRODUCTION

One of the most important tasks in database technology is to combine the following activities: data mining or inferring knowledge from data and query processing or reasoning on acquired knowledge. The solution of this task requires a logical language with unified syntax and semantics for integrating deductive (using knowledge) and inductive (acquiring knowledge) reasoning.

In this paper, we propose a unified model of commonsense reasoning. We also demonstrate that a large class of inductive machine learning (ML) algorithms can be transformed into the commonsense reasoning processes based on well-known deduction and induction logical rules. The concept of a good classification (diagnostic) test (Naidenova & Polegaeva, 1986) is the basis of our approach to combining deductive and inductive reasoning.

The unique role of the good test's concept is explained by the equivalence of the following relationships (Cosmadakis et al., 1986):

- Functional/implicative dependencies between attributes/values of attributes;
- Partition dependencies between classifications generated by attributes (attributes' values) on a set of objects descriptions.

The task of inferring good diagnostic tests is formulated as the search for the best approximations of a given classification (a partitioning) on a given set of objects' examples. It is this task that some well known ML problems can be reduced to (Naidenova, 1996): finding keys and

functional dependencies in database relations, finding implicative dependencies and association rules, inferring logical rules (if-then rules, rough sets, and "ripple down" rules) and decision tree from examples, learning by discovering concept hierarchies, and some others.

The analysis of ML algorithms in the framework of good tests inferring and their decomposition to subtasks and elementary operations made it possible to see that they are the processes of interconnected deductive and inductive commonsense reasoning.

BACKGROUND

There is not an exact definition of commonsense reasoning. This area of research covers a wide range of topics: default reasoning (Sakama, 2005), active agent's reasoning (Thomason, 2007) and some others, for instance, qualitative reasoning, everyday thought about physical systems, spatial reasoning (Mueller, 2006).

Traditionally, commonsense reasoning is considered only as deduction (using knowledge). Induction of new knowledge from observations is considered in the framework of ML problems. That's why many efforts are made in order to combine inductive and deductive reasoning. Two basic ways for this goal exist: 1) to aggregate into a whole system some well-known models of ML and deductive reasoning (Lavrac & Flash, 2000); 2) to enlarge the logic programming language to support both types of inference in a single formalism (Aragão, & Fernandes, 2004), (Sakama, 2005), (Galitsky et al., 2005), (Lisi, 2006). These approaches are very promising. However, the theoretical basis of these works is first-order predicate calculus with predicate as the main element of knowledge description while the main element of commonsense reasoning is concept.

We propose a model of commonsense reasoning based on processes of classification. Knowledge in this model is a system of coordinated links: objects $\leftrightarrow$ classes of objects, classes of objects $\leftrightarrow$ properties, objects $\leftrightarrow$ properties. For instance, "all squares are rhombs", "square is a rhomb", "all the angles of rectangle are right", "square is a rhomb all the angles of which is right", "if the sun is in the sky and not raining, then the weather is good", "conifers are pine-tree, fir-tree, cedar". These connections have causal nature and can be formally expressed with the aid of implications. By commonsense reasoning we understand constructing and using the coordinated classification connections between objects, properties and classes. This understanding goes back to the work of Jean Piaget & Bärvel Inhelder (1959).

The use of these connections is based on the application of syllogisms as deductive reasoning rules. These are rules of everyday reasoning or commonsense reasoning. The construction of these connections is a field of the application of ML algorithms. Reducing these algorithms to the approximations of an assigned classification (partitioning) of a given set of objects' examples gives the possibility to transform them into a model of reasoning in which inductive inference entails applying deductive commonsense reasoning rules.

TOWARDS AN INTERACTIVE MODEL OF COMMONSENSE REASONING

Commonsense Reasoning Rules

The following types of rules are used for commonsense reasoning (Naidenova, 2007):

INSTANCES (evidences) really observed. Instances serve as a source for inductive inference of generalized rules or implicative assertions.

IMPLICATIVE ASSERTIONS describe regular relationships connecting together objects, properties and classes of objects. We consider the following forms of assertions: **implication** (a, b, c $\rightarrow$ d), **forbidden rule (**a, b, c $\rightarrow$ false (never),

diagnostic rule (x, d → a; x, b → not a; d, b → false), **rule of alternatives** (a or b → true (always); a, b → false), **compatibility** (a, b, c → VA, where VA is the occurrence's frequency of rule).

COMMONSENSE REASONING RULES (CRRs) are rules with the help of which implicative assertions are used, updated and inferred from instances.

The deductive CRRs infer consequences from observed facts with the use of implicative assertions. These rules are the following ones:

modus ponens: "if A, then B"; A; hence B;
modus ponendo tollens: "either A or B" (A, B – alternatives); A; hence not B;
modus tollendo ponens: "either A or B" (A, B – alternatives); not A; hence B;
modus tollens: "if A, then B"; not B; hence not A;
generating hypothesis: "if A, then B"; B; A is possible.

The inductive CRRs are the canons formulated by John Stuart Mill (1900): method of only similarity, method of only distinction, joint method of similarity-distinction, method of concomitant changes, and method of residuum. These methods are not rules but they are the processes in which implicative assertions are generated and used immediately.

The Concept of a Good Diagnostic Test

Let $S = \{1, 2, \ldots, N\}$ be the set of objects' indices (objects, for short) and $T = \{A_1, A_2, \ldots, A_j, \ldots, A_m\}$ be the set of attributes' values (values, for short). Each object is described by a collection of values from T. The definition of good tests is based on correspondences G of Galois on $S \times T$ and two relations $S \to T$, $T \to S$ (Ore, 1944). Let $s \subseteq S$, $t \subseteq T$. Denote by t_i, $t_i \subseteq T$, $i = 1, \ldots, N$ an object. We define the relations as follows: $S \to T$: $t = val(s) = \{$intersection of all t_i: $t_i \subseteq T$, $i \in s\}$ and $T \to S$: $s = obj(t) = \{i: i \in S, t \subseteq t_i\}$. Of course, we have $obj(t) = \{$intersection of all $obj(A)$: $A \in T\}$.

Let $S(+)$ and $S(-) = S/S(+)$ be the sets of positive and negative objects respectively.

A diagnostic test for $S(+)$ is a pair (s, t) such that $t \subseteq T$ ($s = obj(t) \neq \varnothing$), $s \subseteq S(+)$ and $t \not\subset t'$ for $\forall\, t' \in S(-)$.

If (s, t) is a test for a class $S(k)$ of objects, where k is the name of this class, then the implication 'if t, then k' is satisfied.

A diagnostic test (s, t), $t \subseteq T$ ($s = obj(t) \neq \varnothing$) is **good** for $S(+)$ if and only if any extension $s' = s \cup i$, $i \notin s$, $i \in S(+)$ implies that $(s', val(s'))$ is not a test for $S(+)$.

A good test (s, t), $t \subseteq T$ ($s = obj(t) \neq \varnothing$) for $S(+)$ is **irredundant** (GIRT) if any narrowing $t' = t/A$, $A \in t$ implies that $(obj(t'), t')$ is not a test for $S(+)$.

A good test for $S(+)$ is **maximally redundant** (GMRT) if any extension of $t' = t \cup A$, $A \notin t$, $A \in T$ implies that $(obj(t \cup A), t')$ is not a good test for $S(+)$.

Generating all types of tests is based on inferring the chains of pairs (s, t) ordered by the inclusion relation:

(1) $s_0 \subseteq \ldots \subseteq s_i \subseteq s_{i+1} \subseteq \ldots \subseteq s_m$ $(val(s_0) \supseteq val(s_1) \supseteq \ldots \supseteq val(s_i) \supseteq val(s_{i+1}) \supseteq \ldots \supseteq val(s_m))$,

(2) $t_0 \subseteq \ldots \subseteq t_i \subseteq t_{i+1} \subseteq \ldots \subseteq t_m$ $(obj(t_0) \supseteq obj(t_1) \supseteq \ldots \supseteq obj(t_i) \supseteq obj(t_{i+1}) \supseteq \ldots \supseteq obj(t_m))$.

Inductive Inference for Constructing Good Diagnostic Tests

The following inductive transitions from one element of a chain to its nearest element are used: (i) from s_q to s_{q+1}, (ii) from t_q to t_{q+1}, (iii) from s_q to s_{q-1}, (iv) from t_q to t_{q-1}, where q, q+1, q-1 are the cardinalities of enumerated subsets.

Transitions can be smooth or boundary. Upon smooth transition, a certain assigned property of generated pairs does not change. Upon boundary transition, a certain assigned property of gener-

ated pairs changes to the opposite one. We control the following properties: "to be a test for $S(+)$", "to be an irredundant collection of values", "not to be a test for $S(+)$", "to be a good test for $S(+)$" and some others.

Special rules are necessary for realizing these inductive transitions.

The generalization rule is used to get all the extensions s_{q+1} of a collection s_q such that (s_q, $val(s_q)$) and (s_{q+1}, $val(s_{q+1})$) are tests for $S(+)$. A chain of generalizations is terminated if any extension does not correspond to a test. This rule is used for inferring GMRTs (Naidenova, 2006).

The specification rule is used to get all the extensions t_{q+1} of a collection t_q such that t_q and t_{q+1} are irredundant collections of values and ($obj(t_q)$, t_q), ($obj(t_{q+1})$, t_{q+1}) are not tests for $S(+)$. A chain of specifications is terminated if any extension is either a redundant collection of values or it corresponds to a test for $S(+)$. This rule is used for inferring GIRTs (Megretskaya, 1988).

The dual generalization (specification) rules relate to narrowing collections of values (objects).

All inductive transitions take their interpretations in human mental acts. The extending of a set of objects with checking the satisfaction of a given condition is a typical method of inductive reasoning. In pattern recognition, the process of inferring hypotheses about the unknown values of some attributes is reduced to the maximal expansion of a collection of the known values of some attributes in such a way that none of the forbidden pairs of values would belong to this expansion. The contraction of a collection of values is used, for instance, in order to delete from it redundant or non-informative values. The contraction of a collection of objects is used, for instance, in order to isolate a certain cluster in a class of objects. Thus, we separate lemons in the citrus fruits.

The boundary inductive transitions are used to get:

(1) all the collections t_q from a collection t_{q-1} such that (obj(t_{q-1}), t_{q-1}) is not a test but (obj(t_q), t_q) is a test, for a given set of objects;
(2) all the collections t_{q-1} from a collection t_q such that (obj(t_q), t_q) is a test, but (obj(t_{q-1}), t_{q-1}) is not a test for a given set of objects;
(3) all the collections s_{q-1} from a collection s_q such that (s_q, $val(s_q)$) is not a test, but (s_{q-1}, $val(s_{q-1})$) is a test for a given set of objects;
(4) all the collections of s_q from a collection s_{q-1} such that (s_{q-1}, $val(s_{q-1})$) is a test, but (s_q,$val(s_q)$) is not a test for a given set of objects.

All the boundary transitions are interpreted as human reasoning operations. Transition (1) is used for distinguishing two diseases with similar symptoms. Transition (2) can be interpreted as including a certain class of objects into a more general one. For instance, square can be named parallelogram, all whose sides are equal (without the property "all its angles are right"). In some intellectual psychological texts, a task is given to remove the "superfluous" (inappropriate) object from a certain group of objects (rose, butterfly, phlox, and dahlia) (transition (3)). Transition (4) can be interpreted as the search for a refuting example.

Note that reasoning begins with using a mechanism for restricting the space of the search for tests: 1) for each collection of values (objects), to avoid constructing all its subsets, 2) for each step of reasoning, to choose a collection of values (objects) without which good tests can not be constructed. For this goal, admissible and essential values (objects) are determined. The search for the admissible or essential values (objects) is based on special **inductive diagnostic rules**.

With the search for tests, implicative assertions are generated and used immediately. For instance, the assertions $(s \cup i) \rightarrow false$, $(t \cup A) \rightarrow false$ are generated and used for realizing the inductive diagnostic rules.

The Decomposition of Inferring Good Diagnostic Tests into Subtasks

To transform the search for good tests into an incremental process we introduce two kinds of subtasks: for a given sets $S(+)$ and $S(-)$:

- Given a positive object t, find all GMRTs (GIRTS) contained in t: $\{\forall t'$: $t' \subset t$, $(obj(t'), t')$ is a good test for $S(+)\}$ (**the subtask of the first kind**);
- Given a non-empty collection of values X such that it is not a test for $S(+)$, find all GMRTs (GIRTs) containing X: $\{\forall t$: $X \subset t$, $(obj(t), t)$ is a good test for $S(+)\}$ (**the subtask of the second kind**).

Special operations are necessary for realizing the decomposition of the search for tests into these subtasks: choosing an object (value) for a subtask, forming a subtask, removing values (objects) from a subtask and some rules controlling the process of inferring good tests.

Only essential objects and values are selected for the formation of subtasks.

There is a powerful rule for removing values without loss of desired solutions. Let X be a GMRT for $S(+)$ and $A \in T$. If $obj(A) \subseteq obj(X)$, then A can be removed from consideration because it will not belong to any GMRT for $S(+)$ different from X.

Examples of the use of two kinds of subtasks can be found in (Naidenova, 2006).

Incremental learning is necessary when a new portion of examples becomes available over time. Suppose that a new example t comes with the indication of its class membership. The following actions are necessary with its arrival:

- Deduction or using already known tests for determining the class membership of t and increasing the inductive power of tests that recognize it correctly;
- Inferring new tests contained in t (the subtask of the first kind);
- Correcting tests for negative classes that accept t (the subtask of the second kind).

It is clearly that incremental learning realizes the intercommunication between data and knowledge via reasoning process.

FUTURE TRENDS

The model of commonsense reasoning described above is in the stage of its formation. In the future, we plan to develop this work in the following directions.

The decomposition of good tests inferring into two kinds of subtasks leads naturally to algorithms realizing parallel calculations.

The algorithmic equivalence of inferring functional and implicative dependencies allows merging these two processes into a whole one with the result of obtaining the main structures of conceptual knowledge: (1) hierarchical structures of objects' classes; (2) implications describing regular links between objects' properties and classes.

It is very important to enlarge our model of commonsense reasoning in order to include in it inferring and using approximate implications with some justification measure.

CONCLUSION

This paper proposes an approach to ML problems based on the search for the best approximations of a given classification on a given set of objects' examples. This approach allows transforming the ML tasks into the commonsense reasoning processes combining deductive reasoning based on four forms of syllogisms and inductive reasoning based on the canons of induction of J.S. Mill.

REFERENCES

Aragão, M., & Fernandes A. (2004). Seamlessly supporting combined knowledge discovery and query answering: A case study. In E. Suzuki & S. Arikava (Eds.), *Proceedings of the 7th International Conference "Discovery Science*. Lecture Notes in Computer Science, *3245*, 403-411. Berlin: Springer.

Cosmadakis, S., Kanellakis, P. C., & Spyratos, N. (1986). Partition semantics for relations. *Journal of Computer and System Sciences, 33*(2), 203-233.

Galitsky, B. A., Kuznetsov, S. O., & Vinogradov, D. V. (2005). *JASMINE: A hybrid reasoning tool for discovering causal links in biological data*. Retrieved from http://www.dcs.bbk.ac.uk/~galitsky/Jasmine.

Lavraĉ, N., & Flash, P. (2000). *An extended transformation approach to Inductive Logic Programming.* (Tech. Rep. No. 00-002). UK: University of Bristol.

Lisi, F. A. (2006). Practice of inductive reasoning on the semantic Web: A system for semantic Web mining. In J. J. Alferes, J. Bailey, W. May, & U. Schwertel (Eds.), *Lecture Notes in Computer Science, 4187*, 242-256. Berlin: Springer.

Megretskaya, I. A. (1988). Construction of natural classification tests for knowledge base generation. In Y. Pecherskij (Ed.), *Problems of Expert System Application in the National Economy* (pp. 89-93). Kishinev: Institute mathematics of Moldavia's Academy of Sciences.

Mill, J. S. (1900). *The system of logic*. Moscow, Russia: Russian Publishing Company "Book Affair".

Mueller, E. T. (2006). *Common Sense Reasoning.* UK: Elsevier.

Naidenova, X. A. (1996). Reducing machine learning tasks to the approximation of a given classification on a given set of examples. *In Proceedings of the 5th National Conference on Artificial Intelligence, 1*, 275-279. Tatarstan: Kazan University Press.

Naidenova, X. A. (2006). An incremental learning algorithm for inferring logical rules from examples in the framework of common reasoning process. In E. Triantaphyllou, & G. Felici (Eds.), *Data mining and knowledge discovery approaches based on rule induction techniques* (pp. 89-146). USA: Springer.

Naidenova, X. A. (2007). Reducing a class of machine learning algorithms to logical commonsense reasoning operations. In G. Felici, & C. Vercellis (Eds.), *Mathematical Methods for Knowledge Discovery and Data Mining* (pp. 41-64). Hershey – New York: ISR Press.

Naidenova, X. A., & Polegaeva, X. A. (1986). An algorithm of finding best diagnostic tests. In G. E. Mintz, & P. P. Lorents (Eds.), *The 4th All Union Conference on Application of Mathematical Logic Methods* (pp. 63-67). Tallinn, Estonia: Institute of Cybernetics, National Academy of Sciences of Estonia.

Ore, O. (1944). Galois connexions. *Trans. Amer. Math. Society, 55*(1), 493-513.

Piaget, J., & Inhelder, B. (1959). *La genèse des structure logiques élémentaires: classifications et sériations.* Neuchâtel: Delachaux & Niestlé.

Sakama, C. (2005). Ordering default theories and non-monotonic logic programs. *Theoretical Computer Science, 339*, 127-152.

Thomason, R. H. (2007). Conditional and Action Logics. *In Logical Formalizations of Commonsense Reasoning: Papers from the AAAI Spring Symposium*. (AAAI Technical Report No SS-07-05), Menlo Park, California, USA: AAAI Press.

KEY TERMS

Commonsense Reasoning Rules (CRRs): These are rules with the help of which implicative assertions are used, updated and inferred from examples. The deductive CRRs are based on the use of syllogisms: modus ponens, modus ponendo tollens, modus tollendo ponens, and modus tollens. The inductive CRRs are the canons formulated by J. S. Mill (1900). Commonsense reasoning is based on using the CCRs.

Diagnostic or Classification Test: Assume that we have two sets of objects' examples called positive and negative examples respectively. A test for a subset of positive examples is a collection of attributes' values describing this subset of examples if it is unique, i. e. none of the negative examples is described by it.

Good Classification Test: A classification test for a given set of positive examples is good if this set is maximal in the sense that if we add to it any positive example, then the collection of attributes' values describing the obtained set will describe at least one negative example.

Good Irredundant Test (GIRT): A good test is irredundant if deleting any attribute's value from it changes its property "to be test" into the property "not to be a test".

Good Maximally Redundant Test (GMRT): A good test is a maximally redundant if extending it by any attribute's value not belonging to it changes its property "to be a good test" into the property "to be test but not a good one".

Implicative Assertions: Implications describe the regularities mutually connecting objects, properties and classes of objects. They can be given explicitly by an expert or derived automatically from examples via some learning process.

Inductive Transitions: These are the processes of extending or narrowing collections of values (objects). They can be smooth and boundary. Upon smooth transition, a certain assigned property of the generated collections does not change. Upon boundary transition, a certain assigned property of the generated collections changes to the opposite one.

The Subtask of the First Kind: Assume that we have two sets of positive and negative examples and a positive example. The subtask of the first kind is to find all the collections of attributes' values that are included in the description of this example and correspond to the good tests (GNRTs or GIRTs) for the set of positive examples.

The Subtask of the Second Kind: For a given set of positive and negative examples and a non-empty collection of attributes' values such that it is not a test for the set of positive examples, find all GMRTs (GIRTs) containing it.

Chapter LXVI
Machine Learning and Data Mining in Bioinformatics

George Tzanis
Aristotle University of Thessaloniki, Greece

Christos Berberidis
Aristotle University of Thessaloniki, Greece

Ioannis Vlahavas
Aristotle University of Thessaloniki, Greece

INTRODUCTION

Machine learning is one of the oldest subfields of artificial intelligence and is concerned with the design and development of computational systems that can adapt themselves and learn. The most common machine learning algorithms can be either supervised or unsupervised. Supervised learning algorithms generate a function that maps inputs to desired outputs, based on a set of examples with known output (labeled examples). Unsupervised learning algorithms find patterns and relationships over a given set of inputs (unlabeled examples). Other categories of machine learning are semi-supervised learning, where an algorithm uses both labeled and unlabeled examples, and reinforcement learning, where an algorithm learns a policy of how to act given an observation of the world.

Data mining is a more recently emerged field than machine learning is. Traditional data analysis techniques often fail to process large amounts of -often noisy- data efficiently. The scope of data mining is the knowledge discovery from large data amounts with the help of computers. It is an interdisciplinary area of research, that has its

roots in databases, machine learning, and statistics and has contributions from many other areas such as information retrieval, pattern recognition, visualization, parallel and distributed computing. The main difference between machine learning and data mining is that machine learning algorithms focus on their effectiveness, whereas data mining algorithms focus on their efficiency and scalability.

Recently, the collection of biological data has been increasing at explosive rates due to improvements of existing technologies as well as the introduction of new ones that made possible the conduction of many large scale experiments. An important example is the Human Genome Project, that was founded in 1990 by the U.S. Department of Energy and the U.S. National Institutes of Health (NIH) and was completed in 2003. A representative example of the rapid biological data accumulation is the exponential growth of GenBank (Figure 1), the U.S. NIH genetic sequence database (www.ncbi.nlm.nih.gov). The explosive growth in the amount of biological data demands the use of computers for the organization, the maintenance and the analysis of these data. This led to the evolution of bioinformatics, an interdisciplinary field at the intersection of biology, computer science, and information technology. Luscombe et al. (2001) identify the aims of bioinformatics as follows:

- The organization of data in a way that allows researchers to access existing information and to submit new entries as they are produced.
- The development of tools that help in the analysis of data.
- The use of these tools to analyze the individual systems in detail, in order to gain new biological insights.

There is a strong interest in methods of knowledge discovery and data mining to generate models of biological systems. In order to build knowledge discovery systems that contribute to our understanding of biological systems, biological research requires efficient and scalable data mining systems.

BACKGROUND

One of the basic characteristics of life is diversity, which can be noticed by the great differences among living creatures. Despite this diversity, the molecular details underlying living organisms are

Figure 1. Growth of GenBank (1982-2005)

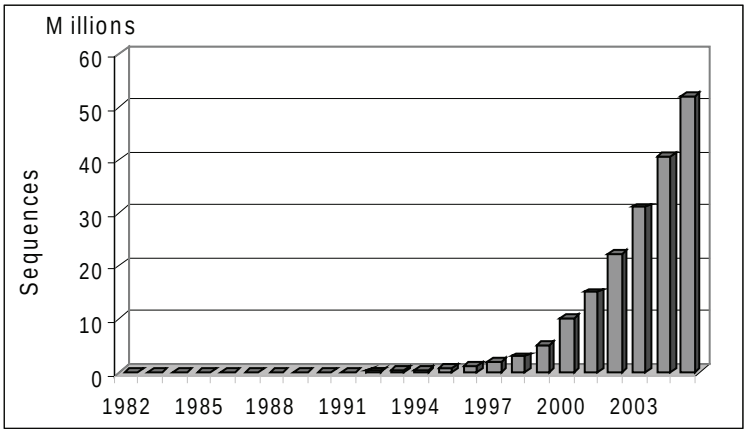


Table 1. The four levels of protein conformation

Conformation Level	Description
Primary structure	The sequence of amino acids, forming a chain called polypeptide.
Secondary structure	The structure that forms a polypeptide after folding.
Tertiary structure	The stable 3D structure that forms a polypeptide.
Quaternary structure	The 3D structure formed by the conjugation of two or more polypeptides.

almost universal. Every living organism depends on the activities of a complex family of molecules called *proteins.* Proteins are the main structural and functional units of an organism's *cell.* A typical example of proteins is enzymes, which catalyze (accelerate) chemical reactions. There are four levels of protein structural arrangement (conformation) as listed in Table 1. The statement about unity among organisms is strengthened by the observation that similar protein sets, having similar functions, are found in very different organisms. Another common characteristic of all organisms is the presence of a second family of molecules, the *nucleic acids.* Their role is to carry the information that "codes" life. The force that created both the unity and the diversity of living things is evolution (Hunter, 2004).

Proteins and nucleic acids are both called *macromolecules,* due to their large size compared to other molecules. Important efforts towards understanding life are made by studying the structure and function of biological macromolecules. The branch of biology concerned in this study is *molecular biology.*

Both proteins and nucleic acids are linear *polymers* of smaller molecules called *monomers.* The term sequence is used to refer to the order of monomers that constitute a macromolecule. A sequence can be represented as a string of different symbols, one for each monomer. There are twenty protein monomers called *amino acids.* There exist two nucleic acids, *deoxyribonucleic acid* (*DNA*) and *ribonucleic acid* (*RNA*). The DNA and RNA monomers are *nucleotides.* There are five different nucleotides depending on the nitrogen base they contain, namely *adenine* (*A*), *cytosine* (*C*), *guanine* (*G*), *thymine* (*T*), and *uracil* (*U*). DNA does not contain U, whereas RNA contains U instead of T. DNA is the genetic material of almost every living organism. RNA is the genetic material for some viruses such as HIV, but has also many functions inside a cell and plays an important role in protein synthesis (Table 2).

Organisms are classified in *eukaryotes* and *prokaryotes.* Eukaryotic cells contain a nucleus, whereas prokaryotic cells lack this structure. Eukaryotes include many organisms like animals, plants, fungi, and protists, whereas prokaryotes include bacteria and archaea. Although some prokaryotes are multicellular organisms, most of them contain a single cell.

The genetic material of an organism is organized in long double stranded DNA molecules called *chromosomes.* An organism may contain

Figure 2. Gene-chromosome relationship

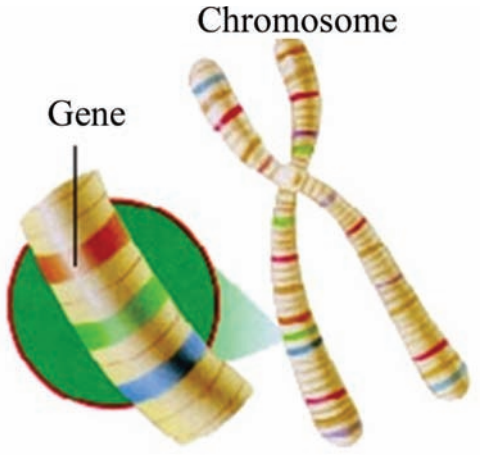


Table 2. Some of the basic types of RNA

RNA Type	Description
Messenger RNA (mRNA)	Carries information from DNA to protein.
Ribosomal RNA (rRNA)	The main constituent of ribosomes, the cellular components where the protein synthesis takes place.
Transfer RNA (tRNA)	Transfers amino acids to ribosomes during protein synthesis.
Small nuclear RNA (snRNA)	It is found in the nucleus and is important in a number of processes including RNA splicing.

one or more chromosomes. A *gene* is a DNA sequence located in a particular chromosome and encodes the information for the synthesis of a protein or RNA molecule. The relationship between a gene and a chromosome is depicted in Figure 2. All the genetic material of a particular organism constitutes its *genome*.

The central dogma of molecular biology, as coined and re-stated by Francis Crick (1958; 1970), describes the flow of the biological sequence information (Figure 3). The general transfers that appear in most organisms are described by the filled arrows. In particular, DNA is transcribed into RNA and then RNA is translated into protein. The circular arrow around DNA denotes its replication ability. Moreover, there are some more specific transfers. In retroviruses RNA is reverse transcribed into DNA. Also, some viruses can replicate their RNA. Finally, in the laboratory it is possible to directly translate DNA into a protein. These more special transfers are denoted by the unfilled arrows of Figure 3.

BIOLOGICAL DATA ANALYSIS

Data mining deals with the discovery of useful knowledge from databases. It is the main step in the process known as Knowledge Discovery in Databases (KDD) (Fayyad et al., 1996), although the two terms are often used interchangeably. Other steps of the KDD process are the selection, cleaning, and transformation of the data and the visualization and evaluation of the extracted knowledge. Some of the most popular tasks are classification, regression, clustering, associa-

Figure 3. The flow of biological sequence information

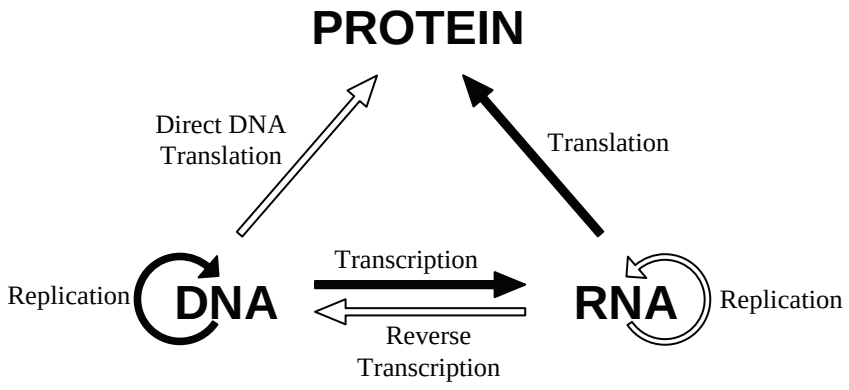


Table 3. Common data mining tasks

Predictive	Descriptive
Classification. Maps data into predefined classes.	**Association Analysis.** The production of rules that describe relationships among data.
	Sequence Analysis. Same as association, but sequence of events is also considered.
Regression. Maps data into a real valued prediction variable.	**Clustering.** Groups similar input patterns together.

tion analysis, and sequence analysis. Depending on the nature of the data as well as the desired knowledge there is a large number of algorithms for each task. All of these algorithms try to fit a model to the data. Such a model can be either *predictive* or *descriptive*. A predictive model makes a prediction about data using known examples, while a descriptive model identifies patterns or relationships in data. Table 3 presents the most common data mining tasks.

The algorithms for the predictive data mining tasks presented in Table 3 correspond to the supervised machine learning algorithms. Respectively, the algorithms for the descriptive data mining tasks correspond to the unsupervised machine learning algorithms. Depending on the purpose of the analysis one can select either a machine learning (interested in effectiveness) or a data mining (interested in efficiency and scalability or dealing with noisy data) algorithm.

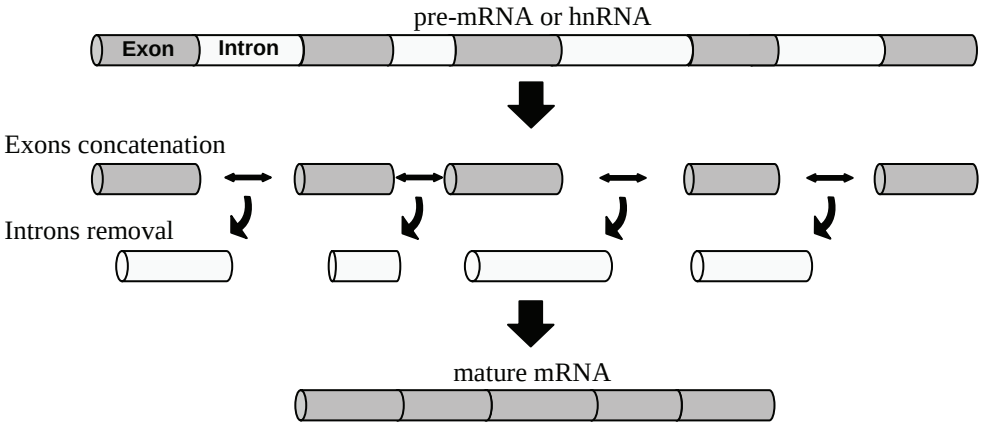
Biological Sequence Analysis

As many genome sequencing projects have been completing, there is the need to annotate those genomes. The observed paradigm shift from static structural genomics to dynamic functional genomics (Houle et al., 2000) and the assignment of functional information to known sequences is deemed particularly important. Gene prediction is concerned with the identification of stretches of DNA that are biologically functional. It is the next step after the sequencing and is particularly important for the annotation and understanding of a genome. Gene prediction is not a straightforward task, especially for eukaryotic genomes, that are more complex. For example, eukaryotic genes consist of coding parts (*exons*) that are separated by intervening non-coding sequences called *introns*. Introns are removed from the transcribed RNA by the process of *splicing* (Figure 4). One of the most important biological problems that demand the construction of predictive data mining or machine learning models is the recognition of the splice sites, namely the boundaries between adjacent exons and introns. The problem becomes even more challenging, if one considers the possibility of alternative splicing, namely the production of different mature mRNA molecules, depending on the number of the exons that are finally concatenated. The phenomenon of alternative splicing is one of the most interesting findings of the last years and guided to the conclusion that a gene may code more than one protein products. Splicing does not take place in prokaryotes, since their genes lack introns. Another reason that gene prediction is easier in prokaryotic genomes is the presence of known conserved patterns around various signals, like promoters, transcription start sites, and translation initiation sites of prokaryotic sequences.

The most important sequence analysis tasks that exploit machine learning and data mining algorithms are the following:

- Prediction of regulatory regions (i.e. *promoters* and *enhancers*), which are segments of DNA where regulatory proteins bind pref-

Figure 4. The process of RNA splicing



erentially and thus control gene expression and consequently protein expression.

- Prediction of the transcription start site, where the process of transcription starts.
- Prediction of the translation initiation site, where the process of translation initiates.
- Prediction of the splice sites for determining exons and introns.
- Prediction of polyadenylation sites where a polyA (multiple adenines) tail is added at the mRNA sequence. Alternative polyadenylation makes the problem more challenging.
- Prediction of coding region in Expressed Sequence Tags (ESTs) that are short and partial sequences of transcribed spliced mRNAs.
- Comparison of a sequence with a database of known sequences for finding possible homologies (e.g. close evolutionary relationships) and grouping of structurally related sequences.

Many machine learning and data mining techniques have been utilized to deal with the above problems. Usually most of these techniques have to be modified and adapted so that can be efficiently and effectively applied to this kind of problems. The most common include neural networks, Bayesian classifiers, decision trees, and support vector machines, (Ma & Wang, 1999; Hirsh & Noordewier, 1994; Zien et al., 2000), as well as multiple classifier systems (Tzanis et al., 2007).

Gene Expression Analysis

Each organism contains a number of genes that code the synthesis of an mRNA or protein molecules. Every cell in an organism -with only few exceptions- has the same set of chromosomes and genes. However, two cells may have very different properties and functions. This is due to the differences in abundance of proteins. The abundance of a protein is partly determined by the levels of mRNA which in turn are determined by the expression levels of the corresponding gene. The process of conversion of the information encoded in a gene, first into mRNA and then to a protein structures and functions of a cell is called *gene expression*. A tool for analyzing *gene expression* is *microarray* or *gene chip*. A microarray experiment measures the relative mRNA levels of typically thousands of genes, providing the ability to compare the expression levels of different biological samples. These samples may

correlate with different time points taken during a biological process or with different tissue types such as normal cells and cancer cells (Aas, 2001). Another method for measuring gene expression is Serial Analysis of Gene Expression (SAGE), which allows the quantitative profiling of a large number of mRNA transcripts (Velculescu et al., 1995). Although this method is more expensive than microarrays, it has the advantage that the experimenter does not have to select the mRNA sequences that will be studied.

The greatest challenge posed by gene expression data is that they contain a small number of samples (less than a hundred), and a very large number of features (genes), that is typically in thousands. Many feature selection approaches have been utilized for reducing the dimensionality of the data by selecting a small number of genes (see Xing et al., 2001). Moreover, a large number of genes are usually irrelevant and uninformative for the classification. The danger of overshadowing the contribution of relevant genes is reduced when gene selection is applied.

Clustering is the far most used method in gene expression analysis. Clustering methods can be used to cluster genes with similar behavior or samples with similar gene expressions together. A category of clustering algorithms, called subspace clustering or bi-clustering algorithms, are used to simultaneously cluster genes and samples. Hierarchical clustering is another frequently applied method in gene expression analysis. An important issue concerning the application of clustering methods in microarray data is the assessment of cluster quality. Many techniques such as bootstrap, repeated measurements, mixture model-based approaches, sub-sampling and others have been proposed to deal with the cluster reliability assessment (Yeung et al., 2003; Smolkin & Ghosh, 2003). External criteria have also been used for validating clusters quality based on predefined partitions of the data (Tzanis & Vlahavas, 2007).

Data Mining in Structural Bioinformatics

Structural bioinformatics is the subfield of bioinformatics which is related to the analysis and prediction of the three-dimensional structure of biological macromolecules, especially for proteins. The application of machine learning and data mining in structural bioinformatics is quite challenging, since structural data are not linear. Moreover, the search space for most structural problems is continuous, namely infinite and demands highly efficient and heuristic algorithms.

Many machine learning and data mining algorithms have been utilized for the prediction of various protein properties such as active sites, modification sites, localization, stability, globularity, shape, protein domains, and interactions (Whishart, 2002). The most popular methods for this task are neural networks, nearest neighbor classifiers and hierarchical clustering algorithms.

Machine learning and data mining methods are also applied for protein secondary structure prediction. This problem has been studied for over than 35 years and many techniques have been developed. Initially, statistical approaches were adopted to deal with this problem. Later, more accurate techniques based on information theory, Bayes theory, nearest neighbors, and neural networks were developed. Combined methods such as integrated multiple sequence alignments with neural network or nearest neighbor approaches improve prediction accuracy.

Other important problems of structural bioinformatics that utilize machine learning and data mining methods are the RNA secondary structure prediction, the inference of a protein's function from its structure, the identification of protein-protein interactions and the efficient design of drugs, based on structural knowledge of their target.

Bioinformatics Text Mining

Text mining in molecular biology, defined as the automatic extraction of information about genes, proteins and their functional relationships from text documents (Krallinger and Valencia, 2005), has emerged as a hybrid discipline on the edges of the fields of information science, bioinformatics and computational linguistics.

It is critically important for biology researchers to have access to the most up to date information on their field of research. Current research practice involves on-line search for gene related information utilizing the latest technologies in Information Retrieval, Semantic Web and Text Mining. A new term lately used by bioinformatics experts to describe the text body where they can extract information such as ontology, interaction and function between biological entities is the *textome*. Generally, it can include all parseable and computable scientific text body.

The rapid progress in biomedical research has led to a dramatic increase in the amount of available information, in terms of published articles, journals, books and conference proceedings. Today, PubMed alone provides access to more than 14 million citations from MEDLINE and additional life sciences journals. In total, more than 4,800 journals are currently indexed by PubMed. Although PubMed is by far the richest database of abstracts, citations and full text articles, there is a plethora of such sources of scientific publications on Biology such as NCBI BookShelf for e-books and a large number of on-line resources. The researchers' need to exploit this enormous volume of available information, along with the avail of high performance and efficiency data mining, natural language processing and information retrieval tools has given birth to a new field of research and application, called Bioinformatics Text Mining (BTM). Other terms for BTM are Bio(logical) Text Mining and Biomedical Text Mining.

From a data miner's point of view, biomedical literature has certain characteristics that require special attention, such as heavy use of domain-specific terminology, polysemic words (word sense disambiguation), low frequency words (data sparseness), creation of new names and terms and different writing styles.

Several studies have categorized the tasks of BTM from different points of view. Cohen and Hersh (2005) provide a high-level categorization, identifying the main tasks to be the following:

- Named Entity Recognition (NER).
- Text classification.
- Synonym and abbreviation extraction.
- Relationship extraction.
- Hypothesis generation.

FUTURE TRENDS

Because of the special characteristics of biological data, the variety of new problems and the extremely high importance of bioinformatics research, a large number of critical issues is still open and demands active and collaborative research by the academia as well as the industry. Moreover, new technologies that permit the faster and cheaper conduction of large scale experiments led to a constantly increasing number of new questions on new data. Examples of hot problems in bioinformatics are the accurate prediction of protein structure and gene expression analysis. The recent discovery of the role of a small RNA molecule, called *microRNA* or *miRNA*, has posed new questions. MicroRNA interferes with gene expression by either splitting an mRNA into tiny useless pieces, or preventing it from translating into proteins. In both cases the gene coding for this mRNA is not expressed. Among the most challenging applications of bioinformatics is molecular medicine. Disease prognosis, early disease diagnosis, and therapy response are the current trends. The analysis of a patient's genetic profile

makes possible the personalized medicine, which guides clinicians to select the most appropriate medication. Machine learning and data mining have been developing and improving in order to deal efficiently and effectively with the problems posed by the data explosion in biology. As Houle et al. (2000) mention, these improvements will enable the prediction of protein function in the context of higher order processes such as the regulation of gene expression, metabolic pathways and signaling cascades. The final objective of such analysis will be the illumination of the way conveying from genotype to phenotype and the specification of the molecular and cellular details that govern life.

CONCLUSION

The recent technological advances, have led to an exponential growth of biological data. New questions on these data have been generated. Scientists often have to use exploratory methods instead of confirming already suspected hypotheses. Machine learning and data mining are two relative research areas that aim to provide the analysts with novel, effective and efficient computational tools to overcome the obstacles and constraints posed by the traditional statistical methods. Feature selection, normalization of the data, visualization of the results and evaluation of the produced knowledge are equally important steps in the knowledge discovery process. The mission of bioinformatics as a new and critical research domain is to provide the tools and use them to extract accurate and reliable information in order to gain new biological insights.

REFERENCES

Aas, K. (2001). *Microarray Data Mining: A Survey.* NR Note, SAMBA, Norwegian Computing Center.

Cohen, A. M., & Hersh, W. R. (2005). A survey of current work in biomedical text mining. *Briefings in Bioinformatics, 6*(1), 57–71.

Crick, F. H. C. (1958). On protein synthesis. *Symposium of the Society for Experimental Biology XII,* 139-163.

Crick, F. H. C. (1970). Central Dogma of Molecular Biology. *Nature, 227,* 561-563.

Fayyad, U. M., Piatetsky-Shapiro, G., Smyth, P., & Uthurusamy, R. (1996). *Advances in knowledge discovery and data mining.* AAAI Press/MIT Press, Menlo Park, California, USA.

Hirsh, H., & Noordewier, M. (1994). Using Background Knowledge to Improve Inductive Learning of DNA Sequences. *Proceedings of the 10th IEEE Conference on Artificial Intelligence for Applications,* (pp. 351-357).

Houle, J. L., Cadigan, W., Henry, S., Pinnamaneni, A., & Lundahl, S. (2004. March 10). *Database Mining in the Human Genome Initiative. White-paper,* Bio-databases.com, Amita Corporation. Available: http://www.biodatabases.com/ white-paper.html

Hunter, L. (2004). Life and Its Molecules: A Brief Introduction. *AI Magazine, 25*(1), 9-22.

Krallinger, M., & Valencia, A. (2005). Text-mining and information-retrieval services for molecular biology. *Genome Biology, 6*(224).

Luscombe, N. M., Greenbaum, D., & Gerstein, M. (2001). What is Bioinformatics? A Proposed Definition and Overview of the Field. *Methods of Information in Medicine, 40*(4), 346-358.

Ma, Q., & Wang, J. T. L. (1999). Biological Data Mining Using Bayesian Neural Networks: A Case Study. *International Journal on Artificial Intelligence Tools, Special Issue on Biocomputing, 8*(4), 433-451.

Smolkin, M., & Ghosh, D. (2003). Cluster Stability Scores for Microarray Data in Cancer Studies. *BMC Bioinformatics, 4,* 36.

Tzanis, G., Berberidis, C., & Vlahavas, I. (2007). MANTIS: A Data Mining Methodology for Effective Translation Initiation Site Prediction. *Proceedings of the 29th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, IEEE, Lyon, France.

Tzanis, G., & Vlahavas, I. (2007). Mining High Quality Clusters of SAGE Data. *Proceedings of the 2nd VLDB Workshop on Data Mining in Bioinformatics,* Vienna, Austria.

Velculescu, V. E., Zhang, L., Vogelstein, B., & Kinzler, K. W. (1995). Serial Analysis of Gene Expression. *Science, 270*(5235), 484-487.

Whishart, D.S. (2002). Tools for Protein Technologies. In C. W. Sensen, (Ed.), *Biotechnology, 5b, Genomics and Bioinformatics,* 325-344. Wiley-VCH.

Xing, E. P., Jordan, M. I., & Karp, R. M. (2001). Feature selection for high-dimensional genomic microarray data. *Proceedings of the 18th International Conference on Machine Learning,* (pp. 601-608).

Yeung, Y. K., Medvedovic, M., & Bumgarner, R. E. (2003). Clustering Gene-Expression Data with Repeated Measurements. *Genome Biology, 4*(5), R34.

Zien, A., Ratsch, G., Mika, S., Scholkopf, B., Lengauer, T., & Muller, R.-K. (2000). Engineering Support Vector Machine Kernels that Recognize Translation Initiation Sites. *Bioinformatics, 16*(9), 799-807.

KEY TERMS

Genotype: The exact genetic makeup of an organism.

Gene Mapping: The process of creating a genetic map assigning DNA fragments (genes) to chromosomes.

Gene Regulation: The cellular control of the time that a gene will be activated and the amount of gene product that will be produced.

Phenotype: The physical appearance characteristics of an organism.

Sequence Alignment: The process to test for similarities between a sequence of an unknown target protein, and a single (or a family of) known protein(s).

Sequencing: The process of determining the order of nucleotides in a DNA or RNA molecule or the order of amino acids in a protein.

Transcript: A sequence of messenger RNA.

Chapter LXVII
Sequential Pattern Mining from Sequential Data

Shigeaki Sakurai
Corporate Research & Development Center, Toshiba Corporation, Japan

INTRODUCTION

Owing to the progress of computer and network environments, it is easy to collect data with time information such as daily business reports, weblog data, and physiological information. This is the context in which methods of analyzing data with time information have been studied. This chapter focuses on a sequential pattern discovery method from discrete sequential data. The methods proposed by Pei et al. (2001), Srikant & Agrawal (1996), and Zaki (2001) efficiently discover the frequent patterns as characteristic patterns. However, the discovered patterns do not always correspond to the interests of analysts, because the patterns are common and are not a source of new knowledge for the analysts.

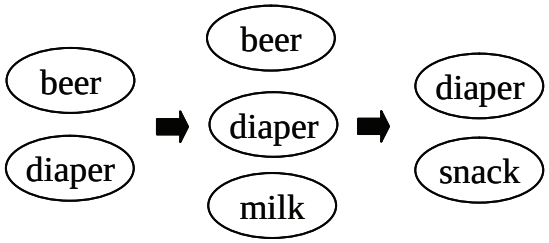
The problem has been pointed out in connection with the discovery of associative rules. Blanchard et al. (2005), Brin et al. (1997), Silberschatz et al. (1996), and Suzuki et al. (2005) propose other criteria in order to discover other kinds of characteristic patterns. The patterns discovered by the criteria are not always frequent but are characteristic of viewpoints. The criteria may be applicable to discovery methods of sequential patterns. However, these criteria do not satisfy the Apriori property. It is difficult for the methods based on the criteria to efficiently discover the patterns. On the other hand, methods that use the background knowledge of analysts have been proposed in order to discover sequential patterns corresponding to the interests of analysts (Garofalakis et al., 1999; Pei et al., 2002; Sakurai et al., 2008b; Yen, 2005).

This chapter focuses on sequential interestingness, which is an evaluation criterion of sequential patterns (Sakurai et al., 2008c). Also, this chapter focuses on 7 types of time constraints that are the background knowledge corresponding to the interests of analysts (Sakurai et al., 2008a). Lastly, this chapter introduces a discovery method based on the sequential interestingness and the time constraints.

BACKGROUND

This chapter explains basic terminology related to the discovery of sequential patterns. Sequential data is rows of item sets and a sequential pattern is a characteristic subrow extracted from the sequential data. Here, an item is an object, an action, or its evaluation in the analysis target. For example, "beer", "diaper", "milk", and "snack" are items in retail business. Each item set has some items that occur at the same time, but each item set does not have multiple identical items. Formally, a sequential pattern s_x is described as $(l_{x1}, l_{x2}, \cdots, l_{xn_x})$, where l_{xi} is an item set and n_x is the number of the item sets included in the sequential pattern. The number n_x is called length and the sequential pattern is called n-th sequential pattern. Also, each l_{xi} is described as $(v_{xi1}, v_{xi2}, \cdots, v_{xim_i})$, where v_{xij} is an item that satisfies the following conditions: $v_{xik_1} \neq v_{xik_2}$ and $k_1 \neq k_2$, and m_i is the number of the items included in the item set l_{xi}. For example, ({"beer", "diaper"}, {"beer", "milk", "snack"}, {"diaper", "snack"}) is an example of the sequential pattern ($s_{example}$) in the retail business. The pattern is a third sequential pattern and is composed of three item sets: {"beer", "diaper"}, {"beer", "milk", "snack"}, and {"diaper", "snack"}. The pattern shows that a person buys "beer" and "diaper" on the first day, buys "beer", "milk", and "diaper" on the second day, and buys "diaper" and "snack" on the third day. The sequential pattern is depicted in Figure 1. In this figure, each circle shows an item, items separated by arrow lines show item sets, and this figure shows that an item set at the left side occurs before an item set at the right side.

Figure 1.



It is necessary to define the frequency of sequential patterns in order to discover the sequential patterns. In advance of this definition, this chapter explains the inclusion of sequential patterns. Let two sequential patterns $s_1 (= (l_{11}, l_{12}, \cdots, l_{1n_1}))$ and $s_2 (= (l_{21}, l_{22}, \cdots, l_{2n_2}))$ be given. s_2 is included in s_1, if s_1 and s_2 satisfy the following conditions: $\exists y \{y_1, y_2, \cdots, y_{n_2}\}$ satisfying the conditions $y_1 < y_2 < \cdots < y_{n_2}$, and $l_{21} \subseteq l_{1y_1}$, $l_{22} \subseteq l_{1y_2}, \cdots$, and $l_{21} \subseteq l_{1y_{n_2}}$. The inclusion is described as $s_2 \subseteq s_1$. For example, a sequential pattern ({"beer", "diaper"}, {"diaper", "snack"}) is included in $s_{example}$, because {"beer", "diaper"} corresponds to the first item set of $s_{example}$ and {"diaper", "snack"} corresponds to the third item set of $s_{example}$. Also, another pattern ({"diaper"}, {"milk"}) is included in $s_{example}$, because {"diaper"} is included in the first item set of $s_{example}$ and {"milk"} is included in the second item set of $s_{example}$. On the other hand, ({"diaper", "milk"}, {"beer"}) is not included in $s_{example}$, because {"diaper", "milk"} is included in the second item set of $s_{example}$ but {"beer"} is not included in the item set after the second item set.

Each sequential pattern is evaluated to determine whether it is included in each row of sequential data. The number of rows including the sequential pattern is regarded as the frequency of the sequential pattern. For example, sequential data is given as shown in Table 1. The frequency of ({"beer", "diaper"}, {"diaper", "snack"}) is 3, because the sequential pattern is included in D1, D3, and D4. Also, the frequency of ({"diaper", "milk"}, {"beer"}) is 2, because the pattern is included in D1 and D2.

Next, this chapter explains the Apriori property, which is the most important property related to the discovery of sequential patterns. The property requires that if a sequential pattern and its sequential subpattern are given, a value of an evaluation criterion of the sequential pattern is smaller than or equal to a value of an

Table 1.

D1	({“beer, “diaper””, “milk”}, {“beer”, “diaper”, “snack”})
D2	({“diaper”, “milk”}, {“beer”, “diaper”}, {“snack”})
D3	({“beer”, “diaper”}, {“beer”}, {“diaper”, “snack”})
D4	({“beer”, “snack”}, {“beer”, “diaper”}, {“diaper”, “snack”})
D5	({“beer”, “milk”}, {“beer”, “diaper”}, {“diaper”, “milk”})

evaluation criterion of the sequential subpattern. For example, if a sequential pattern ({“beer”, “milk”}, {“diaper”}) and its subpattern ({“beer”}, {“diaper”}} are given, an evaluation value of the patterns is smaller than or equal to that of the subpattern. The property can reduce the number of candidate sequential patterns generated by sequential pattern discovery methods.

Using these basic concepts, the following sections introduce a method that discovers sequential patterns corresponding to the interests of analysts.

DISCOVERY OF SEQUENTIAL PATTERNS

Measures for Evaluating Sequential Patterns

In order to discover sequential patterns, it is necessary to evaluate whether the patterns are notable. At first, this section introduces two evaluation criteria for the sequential patterns: the support and the confidence (Agrawal & Srikant, 1995). The support evaluates frequencies of sequential patterns and the confidence evaluates ratios of sequential patterns in the case that sequential subpatterns are given. These criteria are defined by Formula (1) and Formula (2), respectively.

$$supp(s) = \frac{f(s)}{N} \quad (1)$$

$$conf(s \mid s_p) = \frac{f(s)}{f(s_p)} \quad (2)$$

Here, s is a sequential pattern, s_p is a sequential subpattern of the pattern s, $f(s)$ is frequency of the pattern s, and N is the total number of rows of sequential data.

The support satisfies the Apriori property. It is possible for the property to judge whether values of the support of sequential patterns are frequent by evaluating only sequential patterns composed of frequent sequential subpatterns. We can compose an efficient sequential pattern discovery method based on the support. However, the sequential patterns do not always correspond to the interests of analysts.

On the other hand, the confidence can evaluate the relationships between sequential patterns and their sequential subpatterns. The confidence can discover sequential patterns whose frequencies are close to the frequencies of their sequential subpatterns. The analysts are interested in the patterns, because the analysts can use the sequential patterns as probable inference rules. That is, the analysts can predict remaining item sets with high probability by referring to the sequential patterns when their sequential subpatterns are given. The confidence is a better criterion than the support. However, the confidence does not satisfy the Apriori property. We cannot efficiently discover sequential patterns with high confidence.

This section introduces sequential interestingness as a criterion that efficiently discovers sequential patterns corresponding to the interests of analysts. Its definition and its properties are introduced. The criterion notes that a sequential pattern includes sequential subpatterns whose frequencies are not always high and

are close to the frequency of the pattern. The sequential pattern is tied to the sequential subpatterns with high probability, despite the fact that the frequency of the subpattern is not high. We can use the pattern as a probable inference rule. The analysts would be interested in such patterns. The sequential interestingness is defined as a new criterion that discovers the pattern by Formula (3), where $\alpha \geq 0$ is a parameter that represents how important the frequency of the pattern is and the parameter α is called the confidence priority. By using the criterion, sequential patterns that are bigger than or equal to the minimum sequential interestingness given by analysts are discovered. Each discovered pattern is called the interesting sequential pattern. In particular, the pattern is called the interesting item set when the length of the patterns is 1, and the interesting item set is called the interesting item when the item set is composed of an item.

$$inst(s) = \min_{s_p \subseteq s}\{(\frac{1}{f(s_p)})^{\alpha}\} \cdot \frac{\{f(s)\}^{(1+\alpha)}}{N} \qquad (3)$$

The formula corresponds to the definition of the support in the case that α is equal to 0. The criterion satisfies the Apriori property. On the other hand, the transformation of Formula (3) leads to Formula (4). The sequential interestingness is regarded as the support adjusted by the minimum value of the confidence of sequential subpatterns included in a sequential pattern.

$$inst(s) = \min_{s_p \subseteq s}\{(conf(s \mid s_p))^{\alpha}\} \cdot \sup p(s) \qquad (4)$$

Restriction of Sequential Patterns Based on Background Knowledge

This section explains 7 types of time constraints (Sakurai et al., 2008b) in order to incorporate background knowledge given by analysts into the discovery of sequential patterns. These constraints are introduced based on the analysis of some kinds of sequential data such as daily business reports from a sales force automation system and medical examination data of employees.

1. Time constraint between the first item and the last item defines the minimum and the maximum time interval from the first item to the last item. The constraint can evaluate the whole time interval of sequential patterns.
2. Time constraint between the first item and a specified item defines the minimum and the maximum time interval from the first item to a specified item. The constraint can evaluate the relative time from the first item if the specified item occurs in sequential patterns.
3. Time constraint between a specified item and the last item defines the minimum and the maximum time interval from a specified item to the last item.
4. Time constraint between neighboring items defines the minimum and the maximum time interval from an item and its prior neighboring item. The constraint is similar to the gap constraints proposed in (Srikant & Agrawal, 1996). This constraint can evaluate the relative time of all neighboring items in sequential patterns.
5. Time constraint between specified items defines the minimum and the maximum time interval from a specified item to another specified item. The constraint can reflect cyclic relationships and evaluate the relative time for two types of specified items in sequential patterns.
6. Time constraint between a specified item and its prior item defines the minimum and the maximum time interval from an item that occurs just before a specified item to the specified item. The constraint can evaluate the relative time if an item occurs just before the specified item in sequential patterns
7. Time constraint between a specified item and its subsequent item defines the minimum and the maximum time interval from

when a specified item occurs to an item that occurs just after the specified item. The constraint can evaluate the relative time if an item occurs just after the specified item in sequential patterns.

Figure 2 shows these constraints. In this figure, two types of dotted circles are two different specified items. These constraints are checked for all combinations of items of each candidate sequential pattern in the sequential pattern discovery method.

For example, in the analysis of daily business reports, Formula (5) shows that the time interval from the start of business actions for a product to its end is within 180 days and an order is accepted within 90 days from the customer after negative reputation is pointed out for the product. Here, v_F is an item included in the first item set of a sequential pattern, v_L is an item included in its last item set, and *time*(*v*) is a time stamp corresponding to an item *v*. "negative reputation" and "acceptance of order" are items extracted from daily business reports. By using these time constraints, the discovery method can discover daily business reports that find a way of overcoming a negative reputation (Box 1).

Efficient Sequential Pattern Discovery Methods

This section introduces a sequential patterns discovery method based on the combination of Srikant & Agrawal (1996) and the sequential interestingness. Naturally, the sequential interestingness is able to be combined with other techniques for discovering frequent sequential patterns such as Pei et al. (2001) and Zaki (2001). The discovery method is composed of three processes: interesting item discovery, interesting item set discovery, and interesting sequential pattern discovery.

At first, the interesting item discovery process picks up an item from a row of sequential data. The process calculates the number of rows of sequential data including the item as the frequency of the item. The process can calculate the sequential interestingness of the item by dividing the frequency with the total number of rows of the sequential data. The process judges the item to be an interesting item by comparing the sequential interestingness with the minimum sequential interestingness. The process stores interesting items and their frequencies. The process can discover all interesting items by evaluating all items included in the rows of the sequential data.

Next, the interesting item set discovery process generates a candidate item set with i+1 items $V_{i+1}(=\{v_1, v_2 \cdots, v_{i-1}, v_i, v_{i+1}\})$ from two interesting item sets with i items $V_{i,1}(=\{v_1, v_2 \cdots, v_{i-1}, v_i\})$ and $V_{i,2}(=\{v_1, v_2 \cdots, v_{i-1}, v_{i+1}\})$ as shown in Figure 3.

The interesting item set discovery process evaluates whether the candidate item set is interesting. Then, it is necessary for the process to calculate the frequency $f(V_{i+1})$ and the maximum frequency of items in the set $\max_{v_k \in V_{i+1}}\{f(v_k)\}$. The process is repeated until all interesting item sets are discovered. The discovered interesting items and the discovered interesting item sets are regarded as the first interesting sequential patterns.

Next, the interesting sequential pattern discovery process generates a (k+1)-th candidate sequential pattern from two k-th interesting sequential patterns as shown in Figure 4.

Box 1.

$$\left.\begin{array}{l} 0 \le time(v_F) - time(v_L) \le 180 \\ 0 \le time(\text{``negative reputation''}) - time(\text{``acceptance of an order''}) \le 90 \end{array}\right\} \quad (5)$$

Figure 2.

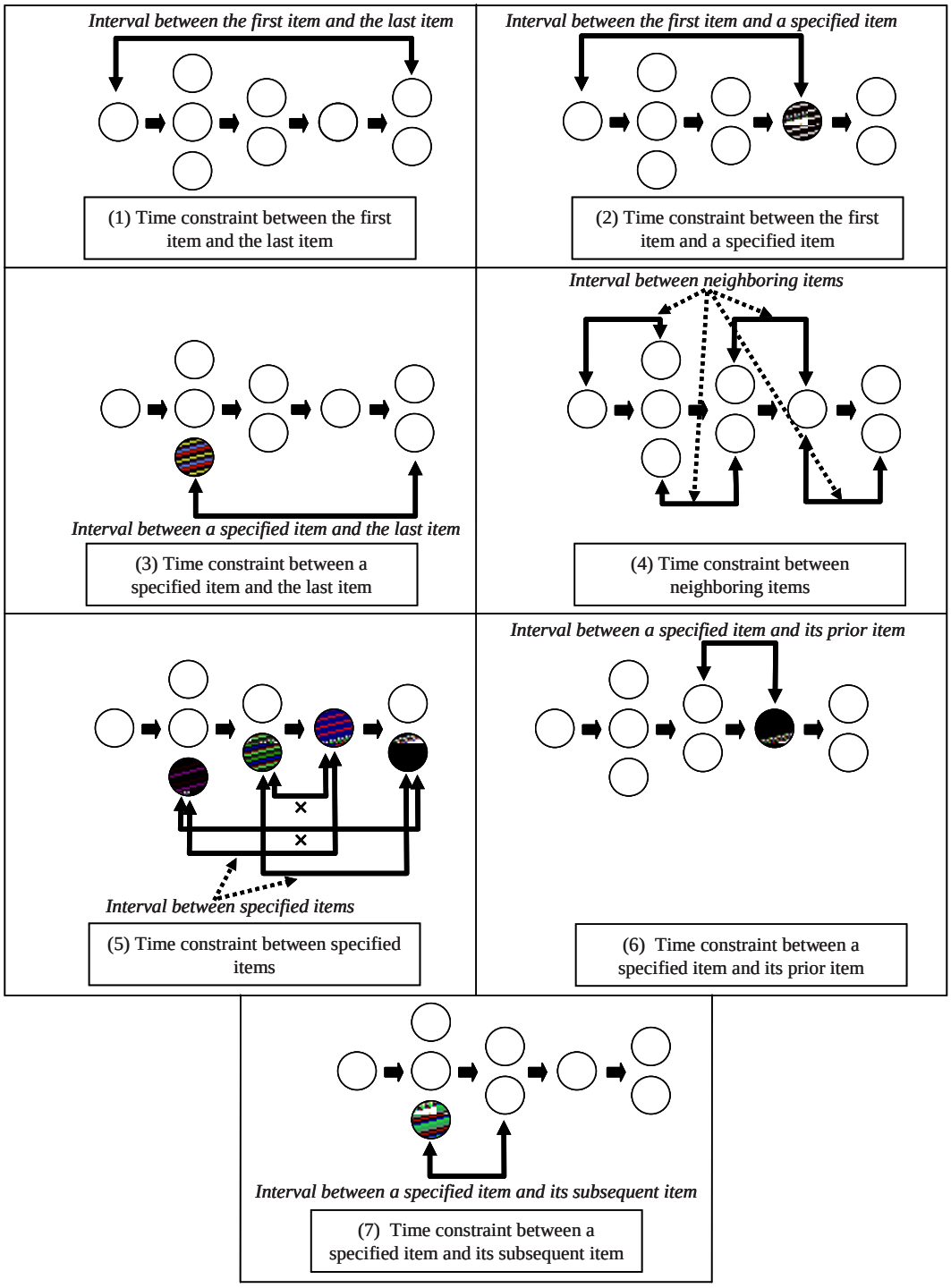
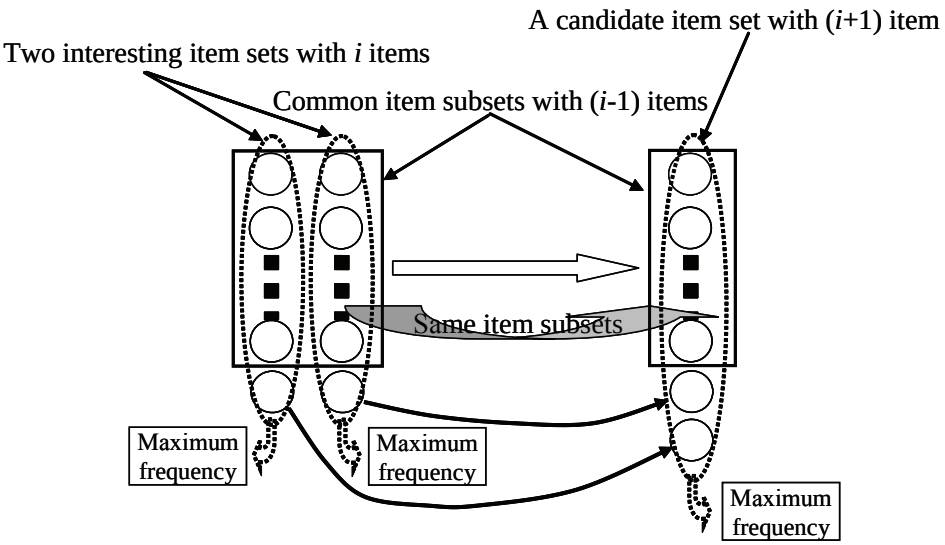

Interval between the first item and the last item
(1) Time constraint between the first item and the last item
Interval between the first item and a specified item
(2) Time constraint between the first item and a specified item
Interval between a specified item and the last item
(3) Time constraint between a specified item and the last item
Interval between neighboring items
(4) Time constraint between neighboring items
Interval between specified items
(5) Time constraint between specified items
Interval between a specified item and its prior item
(6) Time constraint between a specified item and its prior item
Interval between a specified item and its subsequent item
(7) Time constraint between a specified item and its subsequent item

Figure 3.



When the k-th interesting sequential patterns are described as (s_p, l_1) and (s_p, l_2), (k +1)-th candidate sequential pattern is described as (s_p, l_1, l_2). The process evaluates whether the (k +1)-th candidate sequential pattern is interesting. Then, it is necessary for the process to calculate the frequency $f((s_p, l_1, l_2))$ and the maximum frequency of items included in the pattern $\max_{v \in (s_p, l_1, l_2)} \{f(v)\}$. The process is repeated until all interesting item sets are discovered.

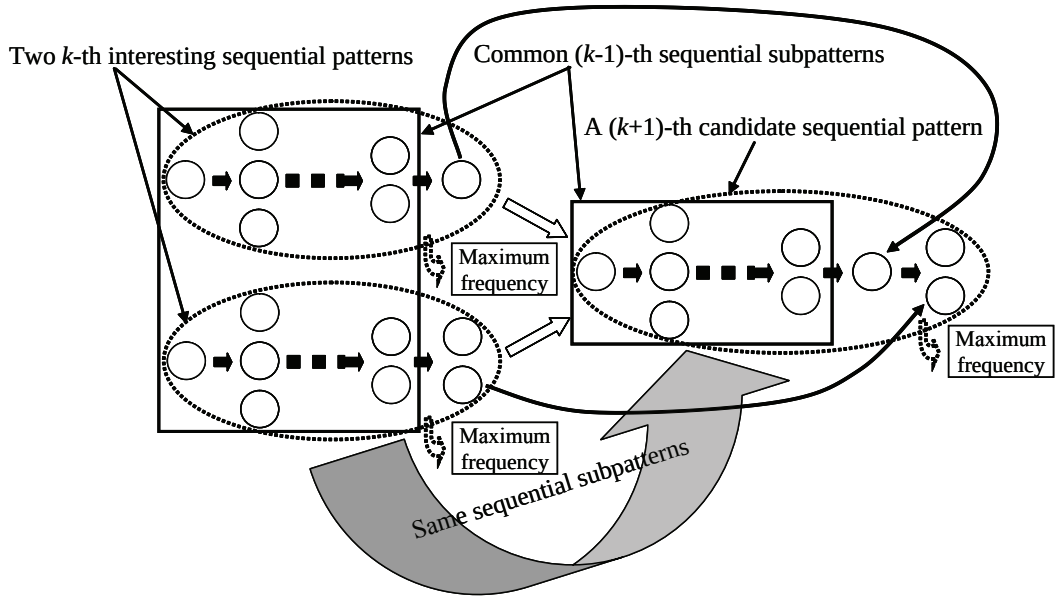
Based on the above discussions, the sequential pattern discovery method can efficiently discover all interesting sequential patterns by expanding the number of items and the length of the sequential pattern from discovery of the first interesting sequential patterns with an item.

FUTURE TRENDS

This chapter focuses on evaluation criteria of sequential patterns, constraint based on background knowledge, and discovery methods of sequential patterns based on the Apriori property. There are many other techniques related to the discovery methods. For example, Yan et al. (2003) propose a method that discovers only sequential patterns containing no superpattern with the same support. Also, Tzvetkov et al. (2003) propose a method that discovers sequential patterns whose evaluation values are within top k. In addition, Parthasarathy et al. (1999) propose a method that discovers sequential patterns from dynamic databases.

On the other hand, the author expects a method to be developed that seamlessly deals with numerical sequential data and discrete sequential data, because there are many sequential data including numerical sequential data and discrete sequential data. Also, the author expects the sequential pattern discovery method to make good progress, enabling it to deal with more complicated sequential data such as streaming data and movie data, because the data includes much information and it is difficult for the discovery method to discover valid sequential patterns speedily. In addition, the author expects a framework to be established that includes preprocessing, the discovery method, post-processing, and user interaction, because all these elements are

Figure 4.



indispensable for the discovery of new knowledge from sequential data.

Moreover, application fields of discovery methods are expanding. For example, the author expects methods to be used for sequential data collected from ubiquitous computing environments. In the near future, many sensors will be embedded in such environments where people live and work, and much sensor information will be collected. The data will be used in order to improve health care and other aspects of people's lives. Also, the author expects the methods to be applied to sequential data related to bioinformatics. Much gene information of various species is stored in databases. The analysis of the information may lead to advances in animal and plant breeding, veterinary medicine, and medical treatment for human beings.

The use of sequential pattern discovery methods will be extended to additional research and application fields.

CONCLUSION

This chapter explained some techniques related to the discovery of sequential patterns. At first, this chapter explained basic terminology and basic definitions. Next, this chapter explained sequential interestingness, which is an evaluation criterion of sequential patterns, and explained some of its properties. This chapter also explained time constraints in order to flexibly use the background knowledge of analysts. In addition, this chapter explained a discovery method based on the sequential interestingness and explained how to use the time constraints. Lastly, this chapter introduced future trends, namely, the extension of the use of the sequential pattern discovery methods to additional research and application fields. This chapter is intended to help readers understand this field and cultivate an interest in it.

REFERENCES

Agrawal, R., & Srikant, R. (1995). Mining Sequential Patterns. *Proceedings of the 11th International Conference Data Engineering*, Taiwan, (pp. 3-14).

Blanchard, J. et al. (2005). Assessing Rule Interestingness with a Probabilistic Measure of Deviation from Equilibrium. *Proceedings of the 11th International Symposium on Applied Stochastic Models and Data Analysis*, France, (pp. 191-200).

Brin, S. et al. (1997). Beyond Market Baskets: Generalizing Association Rules to Correlations, *Proceedings of the 1997 ACM SIGMOD International Conference on Management of Data*, USA, (pp. 265-276).

Garofalakis, et al. (1999). SPIRIT: Sequential Pattern Mining with Regular Expression Constraints. *Proceedings of the 25th International Conference on Very Large Data Bases Conference*, UK, (pp. 223-234).

Parthasarathy, S. et al. (1999). Incremental and Interactive Sequence Mining. *Proceedings of the 8th International Conference on Information and Knowledge Management*, USA, (pp. 251-258).

Pei, J. et al. (2001). PrefixSpan: Mining Sequential Patterns Efficiently by Prefix-Projected Pattern Growth. *Proceedings of the 2001 International Conference Data Engineering*, Germany, (pp. 215-224).

Pei, J. et al. (2002). Mining Sequential Patterns with Constraints in Large Databases. *Proceedings of the 11th ACM International Conference on Information and Knowledge Management*, USA, (pp. 18-25).

Sakurai, S. et al. (2008a). Discovery of Time Series Event Patterns based on Time Constraints from Textual Data. *International Journal of Computational Intelligence, 4*(2), 144-151.

Sakurai, S. et al. (2008b). A Sequential Pattern Mining Method based on Sequential Interestingness. *International Journal of Computational Intelligence, 4*(4), 252-260.

Sakurai, S. et al. (2008c). Discovery of Sequential Patterns Coinciding with Analysts' Interests. *Journal of Computers, 3*(7), 1-8.

Silberschatz, A., & Tuzhilin, A. (1996). What Makes Patterns Interesting in Knowledge Discovery Systems. *IEEE Transactions on Knowledge and Data Engineering, 8*(6), 970-974.

Srikant, R., & Agrawal, R. (1996). Mining Sequential Patterns: Generalizations and Performance Improvements. *Proceedings of the 5th International Conference Extending Database Technology*, France, (pp. 3-17).

Suzuki, E., & Zytkow, J. M. (2005). Unified Algorithm for Undirected Discovery of Exception Rules. *International Journal of Intelligent Systems, 20*(7), 673-691.

Tzvetkov, P. et al. (2003). TSP: Mining Top-K Closed Sequential Patterns. *Proceedings of the 2003 International Conference on Data Mining*, USA, (pp. 347-354).

Yan, X. et al. (2003). CloSpan: Mining Closed Sequential Patterns in Large Datasets, *Proceedings of the SIAM International Conference on Data Mining*, USA, (pp. 166-177).

Yen, S. -J. (2005). Mining Interesting Sequential Patterns for Intelligent Systems. *International Journal of Intelligent Systems, 20*(1), 73-87.

Zaki, M. J. (2001). SPADE: An Efficient Algorithm for Mining Frequent Sequences, *Machine Learning, 42*(1/2), 31-60.

KEY TERMS

Apriori Property: The Apriori property is the property showing that values of evaluation criteria of sequential patterns are smaller than or equal to those of their sequential subpatterns.

Confidence: The confidence is an evaluation criterion of sequential patterns. The criterion evaluates ratio of sequential patterns in the case that their subpatterns are given. The criterion does not satisfy the Apriori property.

Inclusion: Inclusion is a relationship between two sequences. When all item sets in one sequence are included in item sets in another sequence with given order, the former sequence is included in the latter one.

Item: An item is a minimum element included in rows of sequential data. Also, an item is a special case of sequential patterns.

Sequential Interestingness: The sequential interestingness is an evaluation criterion of sequential patterns. The criterion evaluates ratio of sequential patterns in the case that their subpatterns with minimum frequency are given. The criterion satisfies the Apriori property.

Sequential Pattern: A sequential pattern is a characteristic subrow extracted from sequential data.

Support: The support is an evaluation criterion of sequential patterns. The criterion evaluates frequency of sequential patterns. The criterion satisfies the Apriori property.

Time Constraint: A time constraint is a constraint defined between time stamps of items.

Chapter LXVIII
From Chinese Philosophy to Knowledge Discovery in Databases:
A Case Study: Scientometric Analysis

Pei Liu
Université du Sud Toulon Var, France

Eric Boutin
Université du Sud Toulon Var, France

INTRODUCTION

The field of scientometrics has been looking at the identification of co-authorship through network mapping. Research on this topic focuses on the cooperation of two authors who have published papers together. However, this paper is exploring the latent association of two authors. By 'latent association', we mean that the collaboration between two researchers has not yet occurred but might very likely take place in the future. In this paper, we will aim to find out a couple of authors who have never published together and who bear similar academic interests or study similar subjects. We will also show how the concepts of Yuan (Interdependent arising), Kong (Emptiness), Shi (Energy) and Guanxi (Relationship) in Chinese philosophy contribute to understand 'latent associations'. These four Chinese concepts are the theoretical basis of this paper. By explaining one by one what each concept is about we hope to tackle the two following questions: What do those four concepts exactly tell us? And how are they linked together? Finally, we will look at the empirical case study in scientometrics. We hope to show that this application of Chinese concepts can unravel latent associations between researchers in Database.

CONCEPTUAL MODEL

The Interdependent Arising ("Yuan" in Chinese 缘)

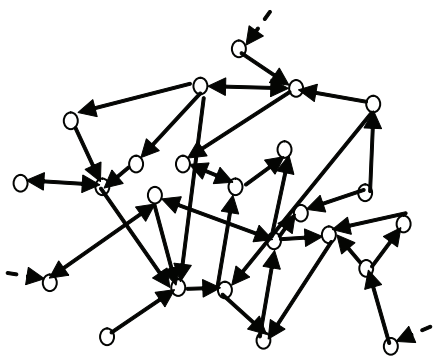
The Chinese believe that every phenomenon arise within the context of a mutually interdependent web of cause and effect as much in time as in

space. This concept is also the basis of Buddhism thought, as it is encompassed in the following classical formulation:

"When this is, that is.
From the arising of this comes the arising of that.
When this isn't, that isn't.
*From the cessation of this comes the cessation of that." (Samyutta Nikaya*雜阿含經*)*

This formulation illustrates the Asian ontology which is that everything appears only because of the arrival of its "arising dependent" (Yuan) and it exists because its "arising dependent" exists. They believe that everything is conditioned and reciprocally influenced (Lai 2003, Cai 1990, Wei 1982, Fo 1992, Duo 1996, 2006). There is nothing in the universe that can survive without its interconnection with other things. Everything depends on everything else. This interdepence forms an incredibly complex web of causation and result encompassing the three universes of foretime, present and future (Figure 1). Whatever we are in whatever form, whenever and wherever we are situated in the world, we are the elements of a connected web within a multidimensional causal nexus (of time and space). In other words, we are just the interconnected nodes of an intricate network receiving the arrows and then sending them to others.

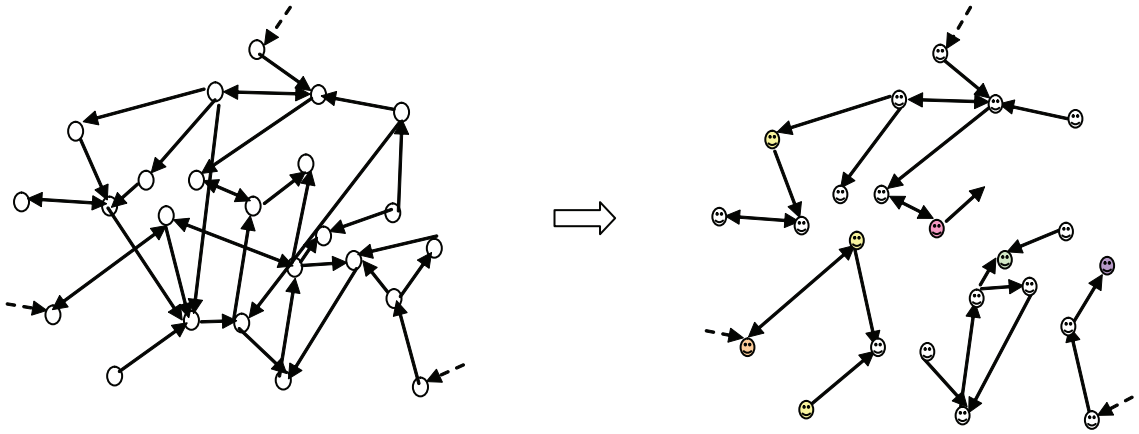
Figure 1. Complex web of causation and result of all things



Yuan (缘) → Guanxi (in Chinese 关系)

The belief of this arising dependent (缘) has deeply influenced the Chinese people. They often conclude that something is inexplicable as a result of the arising dependent, known in Chinese as "Yuan". Two persons can know each other as a result of the Yuan. Someone who knows many people will be complimented as "you ren Yuan", means "have persons' Yuan". Owing to the Yuan's power on which the existence of everything in the word depends, the person involved in many relationships with others can become stronger. If the nodes of the Figure 1 are replaced by people, the web of causation and result of all things is translated into the web of a person's relationships. On the other hand, as the "Yuan" produces outcomes, the relationship between persons is transformed from cause->effect into relationship->outcome. Consequently a person who knows a lot of people has lots of "ren yuan". What is more, this person is more powerful when connected to other persons. He can produce many outcomes and advantages. These outcomes and advantages can also be exchanged for other favours in order to achieve certain purposes, like for example, business activities within a network of informal and interpersonal relationships (Lovette et al., 1999) called "Guanxi". Although, it is not the point to discuss in this paper the theory of Guanxi, it is not difficult to understand why Chinese people and even other East Asian people unconsciously share this same Guanxi philosophy. Guanxi is an intricate and pervasive relational network consisting of mutual obligations, assurances and understandings (Park and Luo, 2001). The source of Guanxi philosophy has its roots in the web of all arising dependent. In that respect, Guanxi can be considered as a subset of the web of Yuan (Figure 2). The whole web and its subset (Guanxi) are also mutually interdependent.

Figure 2. Guanxi network

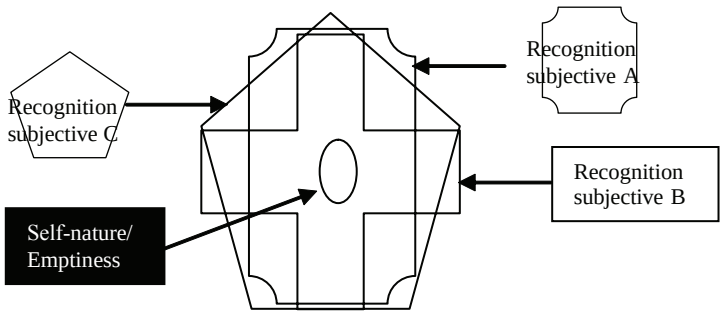


Yuan → Kong (Emptiness, in Chinese空)

All things are inter-conditioned and exist in the universe in a way of impermanence (in Chinese called Wuchang, 无常). As a result, the appearance of phenomena stems from an interdependent effect of the main cause and of countless intrinsic and extrinsic factors. If this arising interdependent stops existing, the phenomena disappear. So there are no permanent phenomena. None truly exists. Everything is therefore asomatous and empty (Nagarjuna, Műlamadhamaka kârikas). This concept of the 'emptiness' means that things do not have a natural form like what we can see (Wu 1996). An object or a person we see is only an identity shaped by our imagination and recognition (Wu 1999). This absolute subjective identification is based on a kind of incorporeal inherent existence. Just like the figure 3, different people see diverse things from the same object, and the object himself does not have an exact shape either. It is a composition of natural factors, like the water and the earth in an inter-conditioned background.

These natural factors which are united by Yuan build up the Self of this object. They become invisible when the Yuan flies away. So the Kong implicates an identification of the self-nature of things. It offers us the opportunity to apprehend and predict the evolution of things. If we keep a vision of the "emptiness", we can find out the self-nature of things and the change it undergoes. The development and the movement of things are in our control. While cultivating this vision of

Figure 3. Different recognition based on the same 'self-nature (emptiness)'



emptiness, we can release our thinking which is often limited and pre-defined by the appearance of phenomena to search in the background the cause of things. So it is important to keep at all time a view of emptiness for every phenomenon.

Kong → Shi (Force and Energy, in Chinese 势)

A metaphor of water is often used by Chinese people. Although small brooks are weak and easy to divert it is strongly believed that they should not be ignored. A powerful large river always starts from hundreds of weak brooks which later become bigger and bigger. "The rush of a torrent which will even roll stones along in its course (Art of the war by Suntsu)". It's the result of the 'Shi' (势) which in Chinese means the energy and force. This implies that we should use the momentum of the water in movement which possesses 'Shi'. The vigour of the torrent will make us more powerful to achieve our final goal. This approach has twofold. One is to study and analyse clearly the environment in order to position ourselves in the right place as early as possible and the other is to be pushed by the force of Shi in the right direction. In other words, a vision of emptiness (Kong) needs to be inside everything in order to control every opportunity of evolution.

Only if we can clearly analyze the entire situation by using the vision of emptiness (Kong) in the context of a cause-effect web of Yuan, we will be able to situate ourselves in a small streamlake. This streamlake has the potential to become a big river in the future when we need it. In contrast if we situate ourselves but not in a river, which for now seems to possess a good Shi, this river might however become droughty tomorrow. That is why it is always worthwhile to choose carefully since the emptiness is a latent force for great changes, such as the Western attitude to the study of weak signals.

Shi and Guanxi (关系)

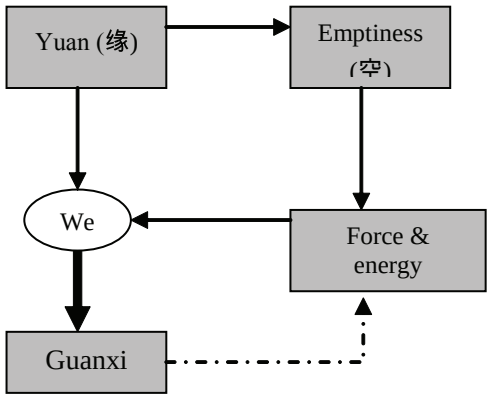
Guanxi (Part 2 Yuan->Guanxi) signifies in China the value of a social capital accumulated by a person. (Warren et al., 2004).Somebody can be hired just because he has a good Guanxi. (Aufrecht and Bun, 1995). A good Guanxi in China equals a good social situation. Guanxi is absolutely necessary to get through social life in China.

Chinese people like to be in "Guanxi Wang", which consists of cultivating carefully a web of relationships and to only use these Guanxi when really needed. This Guanxi philosophy is herewith a good way to balance the power of the Shi. This is because if we consider each tie of the Guanxi network as a small brook, the day when someone has many ties in his web, he will become as powerful as the great river. He will as such be able to get rid of any problem easily. As a consequence, it is viewed logical and efficient for a Chinese person to make use of relationships as a resource and to mobilise one's Guanxi in order to create a favourable Shi and to realize one's own desire.

Brief Summary

The relationship of Yuan, Kong, Shi and Guanxi are illustrated in the Figure 4. Everything in the world comes with the arrival of Yuan and disappears when Yuan vanish. This makes us conclude that everything is empty and temporal. The Kong (emptiness) implies that everything in the world does not truly exist. Therefore the importance of a vision free from what we see through our eyes to find out about the self-nature is necessary. It is useful to cultivate this valuable vision to determine and create the Shi because the Shi is powerful. We can also achieve the target more easily if we control it. The world we live in can be seen as an immense causal web. A web which ties people's relationship is called "Guanxi Wang" in Chinese. It's a type of Chinese working platform: people first react by mobilising their

Figure 4. Causal relations of four concepts



web to position themselves in a good Shi and to pursue their goals.

CREATION OF SHI FOR SCIENTIFIC RESEARCH

The development of our conceptual model brings up the following question "Can we stimulate the development of scientific research?" The affirmative response to this question has led us to another question "Can we re-establish intelligently our 'Guanxi Wang' in order to scientifically enrich ourselves and to ameliorate the production of publications?" According to the perspective of the mentioned above Chinese philosophy, the idea of a Guanxi and Shi arises spontaneously. In the second part of the paper, I would like to describe the identification of Latent Association using the scientometrics method.

In the field of scientometrics, many publications have focused on analyzing the output of scientific research (Aksnes & Taxt, 2004; Weingart, 2005). In particularly in bibliometrics, most scientists studied the exploration of quotation as well as co-publication. They consider these elements as impact factors of scientific evaluation (Moed, 2005; Yi H. & Ao X. et al. 2008; Ulrich S. & Torben S. 2008). But the research which pays attention to the identification of co-authorship through network mapping is normally focusing on the cooperation of two authors who have published papers together. This co-publication relationship represents the actual Guanxi among researchers and shape Shi's existence. In line with our above conclusion, the creation of the new Guanxi and powerful latent Shi in scientific research could facilitate the development of research at a high speed thanks to the force of Shi. So we endeavour to identify the potential of Guanxi between scholars and at the same time to create a new Shi for the scientific research through scientometrics analysis based on the points just raised.

We first define the latent Guanxi of two authors, which refers to the collaboration of researchers that has not yet occurred but might very likely take place in the future.

The concepts of Yuan and Kong teach us a constructive view of the cognitive process that is useful to understand the Yuan (the origin of things) and to find out about the self-nature of things. Co-authorship comes from the fact that the two authors have written an article together. In such association, it must be noted that they also share the same title, keywords and bibliographies. In other words, because scientists in the same field of research are interested in similar subjects, there is a possibility of co-publication. So the identification criteria for latent co-authorship might be, for example:

- Two researchers publish articles using the same keywords
- Two researchers publish articles which are associating the same people
- Two researchers publish articles which have similar references in bibliography.

Methodology

The methodology of this empirical study is technically based on data mining and knowledge discovery. Knowledge Discovery is normally used in

bibliographical databases. The study will employ this method to carry out a comprehensive analysis of network research through studying the latent Guanxi by looking at similar keywords. Hence we will build a network of latent keywords by relating two authors which have not published together, but who have at least published separately one article which is at least described by one identical keyword.

Suppose two articles for which we have the information of authors and keywords. The authors and keywords are expressed as letters and numbers (Table 1).

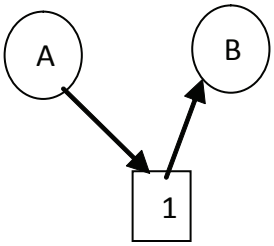
The keyword "1" has been used by author A and B, so if we follow the transitive reasoning we get the Figure 5 because A is described by keyword 1 and the keyword is found in the article of B (Figure 5).

From the example, there are present in six associations: AB, AC, BC, CD, BD and AD. Among these identified associations, we can from now on filter to retain the latent associations. AC, AD, DC, BD are not the latent associations because they have published the same paper. The list of the latent associations is therefore: AB and BC.

Table 1. Demo corpus

Article α: Authors: A, C, D Keywords: 1, 2, 3 Article β: Authors: B, D Keywords: 1, 5, 7

Figure 5. Transitive logic



Experimental Validation

From the data base of archiveSIC[1], we construct a corpus of 30 articles of bibliometrics and scientometrics. The Table 2 shows the first 5 articles of the corpus.

From these data, we create a table of 49 real co-authorship associations by respecting the criterion of two authors who have at least published one paper together. We show a part of associations in Table 3.

Since the original reference doesn't include the keywords of the articles so we enter into each article to find the keywords. Then according to the methodology mentioned above, we can obtain any possible association which link authors who share at least one common keyword. As these associations also compose the existing Guanxi of the co-publication due to the fact that a same article has exactly the same keywords, we removed real associations and listed in Table 4 21 latent Guanxi. The column of frequency is the synthetical value of common keywords and the frequency of each common keyword. It implies the possibility to transpose latent-reality association. Thus the Shi contains the energy of the torrent that can even roll the stones.

The network analysis (Wasserman & Faust 1994) provides a methodology of representation and characterization of the interactions between objects. "*Networks are the language of our times.*" (McCarthy et al. 2004) We therefore represent the interactions of real Guanxi in Figure 6. In Figure 7, we represent the interactions of all relationships including the latent association obtained by network analysis. Each node symbolizes an author, and the size represents the number of articles for each author. A bigger size denotes that more papers have been published in the Bibliometrics field of ArchiveSIC database. For the reason of publication, the different relationships are dis-

Table 2. The first 5 articles of archiveSIC in bibliometrics

Les nouvelles formes d'évaluation scientifique : quelles évolutions en sciences, technique et médecine ?
Durand-Barthez, M.
[sic_00260459 − version 1]

Citations des revues de paléontologie : bilan 2006 et 2007
Néraudeau, D., Dubigeon, I., Le Gall, A.-H.
[sic_00254730 − version 1]

Citations des revues de paléontologie : bilan 2005
Néraudeau, D., Le Gall, A.-H.
[sic_00166236 − version 1]

Citations des revues de paléontologie : bilan 2004
Néraudeau, D., Le Gall, A.-H.
[sic_00166233 − version 1]

Citations des revues de paléontologie : bilan 2003
Néraudeau, D., Le Gall, A.-H.
[sic_00166231 − version 1]

Table 3. Several co-authorship associations

Author	Author
Alain-Hervé_Le_Gall	Didier_Néraudeau
Bador_P.	Boukacem_C.
Bador_P.	Lafouge_T.
Bador_P.	Prost_H.
Bador_P.	SchÖpfel_J.
Beigbeder_M.	Lafouge_T.

tinguished by thinner and thicker ties. Thinner ties mean real associations and the thicker ones signify latent ones. The thickest ties represent a higher frequency of latent associations.

Table 4. Latent association

Latent association		
auteur1	**auteur2**	**Frequency**
Alain_Hervé_Le_Gall	Boukacem_C.	3
Alain_Hervé_Le_Gall	Lafouge_T.	3
Alain_Hervé_Le_Gall	LE_COADIC_Y._F.	3
Alain_Hervé_Le_Gall	Salaün_J._M.	3
Boukacem_C.	Didier_Néraudeau	3
Didier_Néraudeau	Lafouge_T.	3
Didier_Néraudeau	LE_COADIC_Y._F.	3
Didier_Néraudeau	Salaün_J._M.	3
Prime_C.	Smolczewska_A.	3
Lafouge_T.	LE_COADIC_Y._F.	2
Michel_C.	Prime_C.	2
Beigbeder_M.	Michel_C.	1
Beigbeder_M.	Smolczewska_A.	1
Boukacem_C.	Epron_B.	1
Boukacem_C.	LE_COADIC_Y._F.	1
Boukacem_C.	Pouchot_S.	1
Boukacem_C.	Prime_C.	1
Ibekwe_SanJuan_F.	Michel_C.	1
LE_COADIC_Y._F.	Salaün_J._M.	1
Michel_C.	SanJuan_E.	1
Michel_C.	Smolczewska_A.	1

CONCLUSION: INTERPRETATION OF RESULTS AND FUTURE TRENDS

The real research of Guanxi is dispersed in the graphic. Latent ties connect the largest research group with several other researches. Communications and Guanxi could be stabilised with these ties. The core axe of research is more balanceable than horizontal cooperation, if it creates some cooperating opportunities to build new Guanxi

Figure 6. Real association

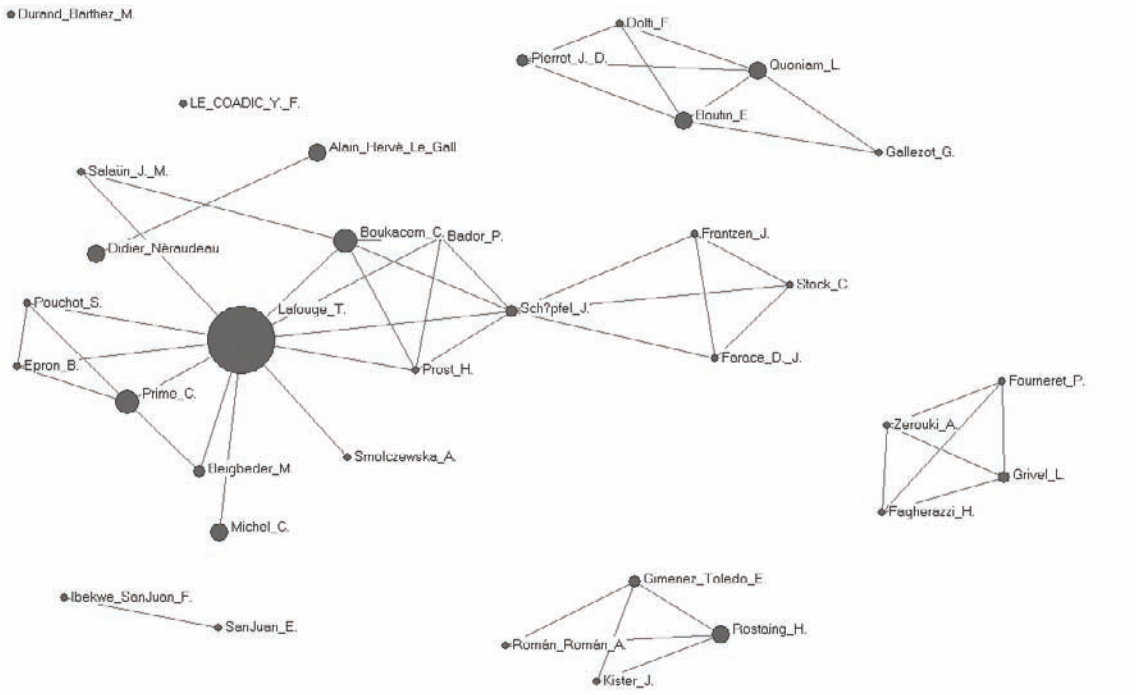
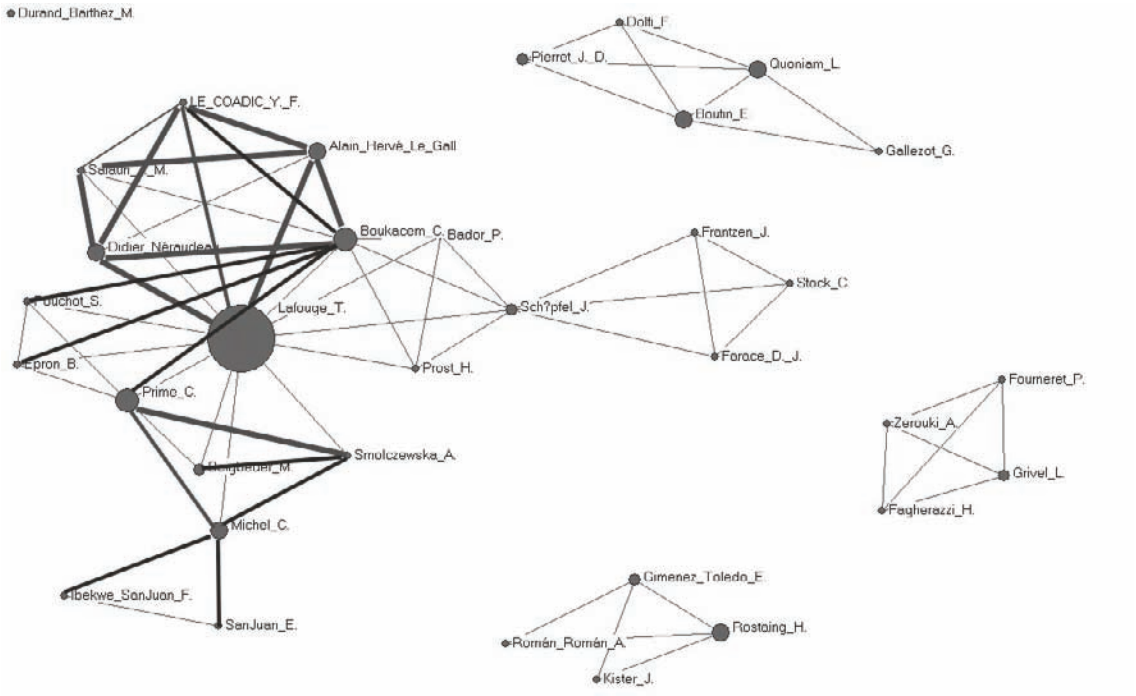


Figure 7. Real network (thinner tie) and latent network (thicker tie)



with the upper author. We clearly see from the figure that there is a potential to create new energy and that the research process could advance more smoothly by enlarging the Shi of the research group's origin.

The approach helps to identify the potential relationships within researchers of the same field who haven't yet published papers together. It contributes not only to the identification of latent research associations, but also it inspires researchers to network for potential cooperation across institutions, countries and field of studies. The potential for relationship brings the Shi to the field of scientific research. Shi as it is developed in latent association can become "the momentum of a round stone rolling down a mountain thousands of feet in height. There is so much on the subject of Shi." (Art of the war by Suntsu)

In our research, the technical method is based on the author and the keywords' information. It could also be studied by basing it on the author and a series of quotations or even on a combination of multi-information including author, keywords, year of publication and citations. Because the year of publication announces the age and the historical context of papers, and citations reflect the way of thinking and the appreciation of relative achievements of authors, since it is considered like a prize awarded for past research (Merton, 1979) and it could be used as a tool to convince others (Latour & Wolgar, 1979). More refined criteria also allow to reinforce the pertinence of identified latent association as well as a direct access to a mighty Shi.

EPILOGUE

We have taken a somewhat unconventional approach to conclude this article. In the first part of the paper, we have adopted a theoretical approach to try to present a new concept inspired by Chinese philosophy. And in the second part, we came back to the fields of scientometrics to analyse the network and verify our conceptual model. Specialised in scientometric study and influenced deeply by Chinese philosophy, we recognize the intellectual connection between these two completely separate concepts. We also consider necessary to share this original thinking with other researchers by presenting this theoretical and empirical research. Since Chinese philosophy is usually talked about by sociologists and is most unlikely acquainted to the researchers in the field of scientometrics, we had to write many texts to describe this conceptual model. The experimentation is based on the title, author and keyword information. The method could also use information such as references, years of publications and other descriptive of the authors, like university or country as this would help to find out and create Shi across institutions and countries.

REFERENCES

Aksnes, D., & Taxt, R. (2004). Peer review and bibliometrics indicators: A comparative study at a Norwegian university. *Research Evaluation, 13*(1), 33-41.

Aufrecht, S. E., & Bun, L. S. (1995). Reform with Chinese Characteristics: The Context of Chinese Civil Service Reform. *Public Administration Review, 55*, 175-183.

Cai, H. M. (1990). The standpoint of the primitive buddhism (原始佛教的缘起观). *Dharmaghosa (The Voice of Dharma), 1990*(05), 25-27.

Duo, S. (1996). Some characters of Buddhism different from the other religion. *Tibetan Folklore, 1996*(03), 52-64.

Duo, S. (2006). Several Key Features of the Difference between Buddhism and Other Religions. *N.W.Ethno-National Studies, 2006*(02).

Fo, R. (1992). Dependent origination and nature origination. *Dharmaghosa (The Voice of Dharma), 1992*(04), 7-15.

Lai, Y. (2003). The interdependent arising is the fundamental theory of Buddhism. *Social Science Front, 2003*(5), 50-52.

Latour, B., & Wolgar, S. (1979). *Laboratory Life: The Social Construction of Scientific Facts.* Sage Pulishing, Los Angeles.

Lovett, S., & Simmons, L. C., & Kali, R. (1999). Guanxi Versus the Market: Ethics and Efficiency. *Journal of International Business Studies, 30*(2), 231–248.

McCarthy, H., & Miller, P., & Skidmore, P. (Eds.) (2004). *Network logic: Who governs in an interconnected world?* London: Demos. Individual essays available at http://www.demos.co.uk/catalogue/networks/.

Merton, R. (1979). Foreword. In E. Garfield (Ed.), *Citation Indexing: Its Theory and Application in Science, Technology, and Humanities, vii–xi.* John Wiley and Sons, New York.

Moed, H. F. (2005). Citation Analysis in Research Evaluation. *Information Science and Knowledge Management.* Springer, Dordrecht.

Park, S. H., & Luo, Y. (2001). Guanxi and Organizational Dynamics: Organizational Networking in China Firms. *Strategic Management Journal, 22*(5), 455–477.

Ulrich, S., & Torben, S. (2008), Are international co-publications an indicator for quality of scientific research? *Scientometrics, 74*(3), 361-377.

Warren, D. E., Dunfee, T. W., & Li, N. (2004). Social Exchange in China: The Double-Edged Sword of Guanxi. *Journal of Business Ethics, 55*, 355-372.

Wasserman, S., & Faust, K. (1994). *Social Network Analysis: Methods and Applications.* Cambridge, England, and New York: Cambridge University Press.

Wei, A. (1982). The dependent origination-the core of Buddhism. *Dharmaghosa (The Voice of Dharma), 1982*(01), 31-33.

Weingart, P. (2005). Impact of bibliometrics upon the science system: Inadvertent consequences? *Scientometrics, 62*, 117-131.

Wu, G. (1996). Commentary to the dependent arising and emptiness. *Wuhan University Journal (Philosophy & Social Science Edition), 1996*(02), 35-38.

Wu, X. (1999). The reconstruction of dependent arising and emptiness contributed by the mind-only theory. *Fudan Journal (Social Sciences), 1999*(03), 67-88.

Yi, H., Ao, X., & Ho, Y. (2008). Use of citation per publication as an indicator to evaluate pentachlorophenol research. *Scientometrics, 75*, 67-80.

KEY TERMS

Guanxi: Chinese concept, a kind of interpersonal relationship, from a type point of view, Guanxi is considered as a standard of personal social evaluation.

Kong (Emptiness): Buddhism philosophy. It means everything in the world is empty and it implies many possibilities. By finding out the self-nature of things, the Kong vision enhance a clearly observation and a profound thinking that are often blinded by the appearance.

Knowledge Discovery: A kind of nontrivial study on extracting implicit and potential, useful information and then transform them accurately into intelligent information.

Latent Association: From a type point of view, latent associations refers to the co-work of researchers that has not occurred until present but will be very possible to take place in the future.

Scientometrics: It means a science of all applications of mathematical and statistical methods which are often developed to measure and evaluate the scientific publications.

Shi (Energy): A kind of momentum contained in the surrounding environment that could be borrowed, transferred or even created to help people achieve his target.

Yuan: It means all phenomena arise in the context of a mutually independent web of cause and effect as much in time as in space. Yuan could be considered as the reason of everything.

ENDNOTE

[1] archiveSIC: an open archive of Science of Information and Communication in France http://archivesic.ccsd.cnrs.fr/

Section VII
Physical Issues

Chapter LXIX
An Overview on Signature File Techniques

Yangjun Chen
University of Winnipeg, Canada

INTRODUCTION

An important question in information retrieval is how to create a database index which can be searched efficiently for the data one seeks. Today, one or more of the following four techniques have been frequently used: full text searching, B-trees, inversion and the signature file. Full text searching imposes no space overhead, but requires long response time. In contrast, B-trees, inversion and the signature file work quickly, but need a large intermediary representation structure (index), which provides direct links to relevant data. In this paper, we concentrate on the techniques of signature files and discuss different construction approaches of a signature file.

The signature technique cannot only be used in document databases, but also in relational and object-oriented databases. In a document database, a set of semistructured (XML) documents is stored and the queries related to keywords are frequently evaluated. To speed up the evaluation of such queries, we can construct signatures for words and superimpose them to establish signatures for document blocks, which can be used to cut off non-relevant documents as early as possible when evaluating a query. Especially, such a method can be extended to handle the so-called containment queries, for which not only the key words, but also the hierarchical structure of a document has to be considered. We can also handle queries issued to a relational or an object-oriented database using the signature technique by establishing signatures for attribute values, tuples, as well as tables and classes.

BACKGROUND

The signature file method was originally introduced as a text indexing methodology (Faloutsos, 1985; Faloutsos, Lee, Plaisant & Shneiderman,

1990). Nowadays, however, it is utilized in a wide range of applications, such as in office filing (Christodoulakis, Theodoridou, Ho, Papa & Pathria, 1986), hypertext systems (Faloutsos, Lee, Plaisant & Shneiderman, 1990), relational and object-oriented databases (Chang & Schek, 1989; Ishikawa, Kitagawa & Ohbo, 1993; Lee & Lee, 1992; Sacks-Davis, Kent, Ramamohanarao, Thom & Zobel, 1995; Yong, Lee & Kim, 1994, Tousidou et al., 2002; Chen, 2004), as well as in data mining (Andre-Joesson & Badal, 1997). It requires much smaller storage space than inverted files, and can handle insertion and update operations in databases easily.

A typical query processing with the signature file is as follows: When a query is given, a query signature is formed from the query value. Then each signature in the signature file is examined over the query signature. If a signature in the file matches the query signature, the corresponding data object becomes a candidate that may satisfy the query. Such an object is called a drop. The next step of the query processing is the false drop resolution. Each drop is accessed and examined whether it actually satisfies the query condition. Drops that fail the test are called false drops while the qualified data objects are called actual drops.

A variety of approaches for constructing signature files have been proposed, such as bit-slice files (Ishikawa, Kitagawa & Ohbo, 1993), S-trees (Deppisch, 1986), and signature trees (Chen, 2002; Chen, 2005), as well as their different variants. In the following, we overview all of them and analyze their computational complexities.

SIGNATURE FILES AND SIGNATURE FILE ORGANIZATION

Signature Files

Intuitively, a signature file can be considered as a set of bit strings, which are called signatures. Compared to the inverted index, the signature file is more efficient in handling new insertions and queries on parts of words. But the scheme introduces information loss. More specifically, its output usually involves a number of false drops, which may only be identified by means of a full text scanning on every text block short-listed in the output. Also, for each query processed, the entire signature file needs to be searched (Faloutsos, 1985; Faloutsos, 1992). Consequently, the signature file method involves high processing and I/O cost. This problem is mitigated by partitioning the signature file, as well as by exploiting parallel computer architecture (Ciaccia & Zezula, 1996).

When creating a signature file, each word is processed separately by a hashing function. The scheme sets a constant number (m) of 1s in the [1..F] range. The resulting binary pattern is called the word signature. Each text is seen to consist of fixed size logical blocks and each block involves a constant number (D) of non-common, distinct words. The D word signatures of a block are superimposed (bit OR-ed) to produce a single F-bit pattern, which is the block signature stored as an entry in the signature file.

Fig. 1 depicts the signature generation and comparison process of a block containing three words (then D = 3), say "John", "12345678", and "professor". Each signature is of length F = 12, in which m = 4 bits are set to 1. When a query arrives, the block signatures are scanned and many non-qualifying blocks are discarded. The rest are either checked (so that the "false drops" are discarded; see below) or they are returned to the user as they are. Concretely, a query specifying certain values to be searched for will be transformed into a query signature s_q in the same way as for word signatures. The query signature is then compared to every block signature in the signature file. Three possible outcomes of the comparison are exemplified in Fig. 1: (1) the block matches the query; that is, for every bit set in s_q, the corresponding bit in the block signature s is

Figure 1. Signature generation and signature file

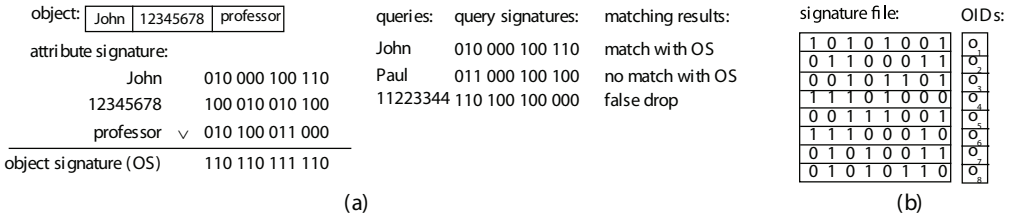
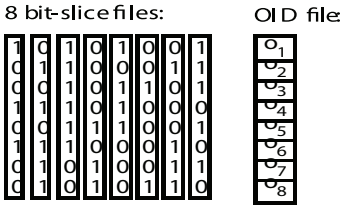


Figure 2. Illustration for bit-slice file



also set (i.e., $s \wedge s_q = s_q$) and the block contains really the query word; (2) the block doesn't match the query (i.e., s Ù $s_q \neq s_q$); and (3) the signature comparison indicates a match but the block in fact doesn't match the search criteria (false drop). In order to eliminate false drops, the block must be examined after the block signature signifies a successful match.

In a signature file, a set of signatures is sequentially stored, which is easy to implement and requires low storage space and low update cost. However, when evaluating a query, a full scan of the signature file has to be performed. Therefore, it is generally slow in retrieval.

Fig. 1(b) shows a simple signature file. To determine the length of signatures, the following formula can be used (Faloutsos, 1985):

$$F \times \ln 2 = m \times D \qquad (1)$$

Bit-Slice Files and Compressed Bit-Slice Files

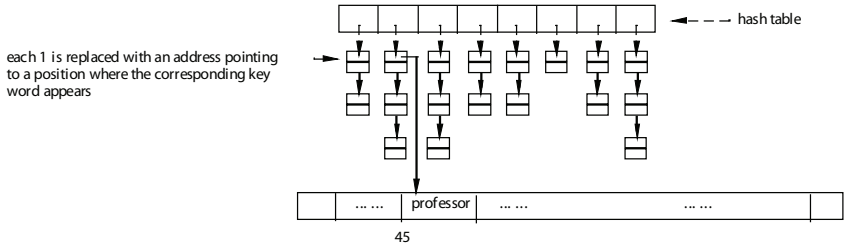
A signature file can be stored in a column-wise manner. That is, the signatures in the file are *vertically* stored in a set of files (Ishikawa, Kitagawa & Ohbo, 1993), i.e., in F files, in each of which one bit per signature for all the signatures is stored as shown in Fig. 2.

With such a data structure, the signatures are checked slice-by-slice (rather than signature-by-signature) to find matching signatures. To demonstrate the retrieval, consider the query signature s_q = 10110000. First, we check the first bit-slice file and find that only three positions: 1st, 4th and 6th positions match the first bit in s_q. Then, we check the second bit-slice file. This time, however, only those three positions will be checked. Since the 2nd bit in s_q is 0, no positions will be filtered. Next, we check the third bit-slice file against the 3th bit in s_q. Because all the three positions are set to 1 in it, the same positions in the next bit-slice file, i.e., in the fourth bit-slice file will be checked against 4th bit in s_q. Since none of the three positions in the fourth bit-slice file matches this bit, the search stops and reports a *nil*.

From this process, we can see that only part of the F bit-slice files have to be scanned. So the search cost must be lower than that of a sequential file. However, update cost becomes larger. For example, an insertion of a new set signature requires about F disk accesses, one for each bit-slice file.

A bit-slice file can also be compressed. According to (Faloutsos & Chan, 1988; Kocberber and Can; 1999), if a proper hash function is chosen, which maps each key word to a signature with only one bit set to 1 (i.e., k = 1), a signature file

Figure 3. Illustration for compressed bit-slice file



will contain much less 1s than 0s. In this case, a signature file can be considered as a sparse matrix, and can be effectively compressed. A simple way to compress a bit-slice file is to replace each 1 with an address to a text position where the corresponding key word can be found. All the addresses in a slice are then stored in a collection of buckets that are linked together. For example, the bit-slice file shown in Fig. 2(b) can be compressed as shown in Fig. 3.

In Fig. 3, an array, called a hash table in ((Faloutsos & Chan, 1988) is used, in which each entry is a pointer to the head of a linked list of buckets. Each of them contains several addresses. Assume that we have a word 'professor' that hashes to a signature with the 2nd bit set to 1 (i.e., hash('professor') = 2), and that it appears in the document starting at the 45th byte of the text file. Then, searching the second linked bucket list, we will find the position where the word 'professor' appears.

The space saved by this method can be estimated as follows. Let l be the size of an address. Then, the size of all the linked bucket lists is on $O(n \cdot D \cdot l)$. The size of the corresponding bit-slice file is $O(n \cdot m + n \cdot l)$. So if $D < \frac{m}{l} - 1$, some space can be saved. However, such a space optimization is achieved at cost of query evaluation time since the probability of false drops will be definitely increased in the case of sparse signature files. In terms of (Christodoulakis & Faloutsos, 1984), when a signature file is half-populated with 1s and half-populated with 0s, we have the lowest false drop probability. Obviously, the precondition for compressing a bit-slice file is just opposite to that.

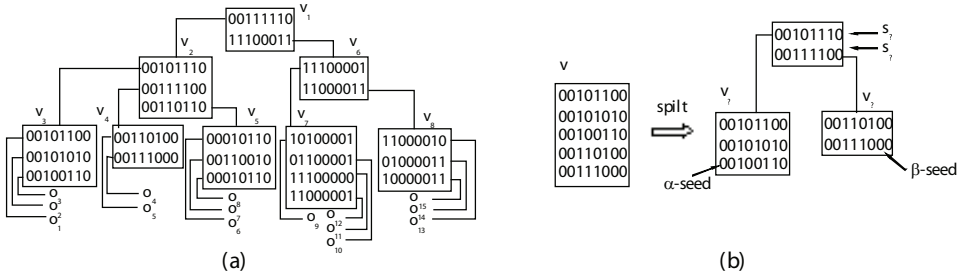
S-Trees and Multilevel Signature Files

Similar to a B^+-tree, an S-tree is a height balanced multiway tree (Deppisch, 1986). Each internal node corresponds to a page, which contains a set of signatures and each leaf node contains a set of entries of the form <*s*, *oid*>, where the object is accessed by the *oid* and *s* is its signature. Let *N* be the parent node of *N'*. Then, there exists a signature in *N*, whose value is obtained by superimposing all the signatures in *N'*. See Fig. 3 for illustration.

To retrieve a query signature, we search the S-tree top-down. However, more than one paths may be visited. The first signature in the root N_1 leads us to its child node N_2 because the third and fourth bits are set to 1. In N_2, the second and third signatures match s_q. Then, we go to the leaf node N_4 and N_5. In N_4, we find two matching candidates $\{o_4, o_5\}$, and in N_5, we have only one $\{o_7\}$.

The construction of an S-tree is an insertion-splitting process. At the very beginning, the S-tree contains only an empty leaf node and signatures in a file are inserted into it one by one. When a leaf node *N* becomes full, it will be split into two nodes and at the same time a parent node N_{parent}

Figure 4. Illustration for S-tree and node splitting



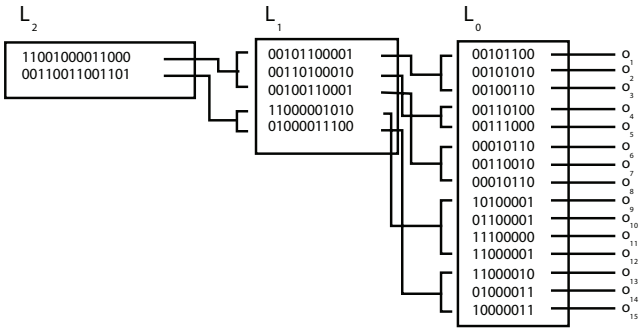
will be generated if it does not exist. In addition, two new signatures will put in N_{parent}. Assume that the capacity of N is K (*i.e.*, N can accommodate K signatures.) Then, when we try to insert the (K + 1)th signature into N, it has to be split into two nodes N_α and N_β. To do this, we will pick a signature in N, which has the heaviest signature weight (*i.e.*, with the most 1s) in N. It is called the α-seed and will be put in N_α. Then, we select a second signature, which has the maximum number of 1s in those positions where α has 0. That is, the signature provides the maximal weight increase to α. This signature is called the β-seed and put in N_β. Any of the rest K - 1 signatures are assigned to N_α or N_β, depending on whether it is closer to N_α or N_β. The two new signatures (denoted s_α and s_β) to be put into the parent node are obtained by superimposing the signatures in N_α and N_β, respectively. See Fig. 3(b) for illustration.

The advantage of this method is that the scanning of a whole signature file is replaced by searching several paths in S-tree. However, the space overhead is almost doubled. Furthermore, due to superimposing, the nodes near the root tend to have heavy weights and thus have low selectivity. This is improved by Tousidou *et al* (Tousidou, Bozanis & Manolopoulos, 2002). They elaborate the selection of the α-seed and the β-seed so that their distance is increased. However, this kind of improvement is achieved at cost of time, i.e., by checking more signatures, which makes the insertion of a signature into a S-tree extremely inefficient.

The multilevel signature file method discussed in (Lee, et al., 1995) follows the same principle as the S-tree. The difference between them is in the way that signatures at a higher level are constructed. In the multilevel signature file method, a signature at a higher level is a superimposed code generated *directly* from a group of p text blocks, instead of superimposing p signatures at the lower level. In other words, a signature at a higher level can be considered as being generated form a bigger block containing more words. Assume that any signature at the lowest level (leaf level) is created from a block of size D. Then, any signature at the second lowest level is generated from a block of size $p \cdot D$. For a better understanding, consider the S-tree shown in Fig. 4(a) once again. Corresponding to this S-tree, we may have a multilevel signature file as shown in Fig. 5.

As in a S-tree, any signature at the lowest level corresponds to an object (or a text block). But any signature at a higher level is constructed in a different way. We pay attention to the level L_1 shown in Fig. 5, at which each signature is not generated by superimposing some signatures at the level L_0, but created directly from the text blocks. For example, the first signature at L_1 is generated by taking o_1, o_2, and o_3 as a single block. So it will have a different length than the signatures in L_0. Obviously, such a data structure needs more space than S-trees. However, the filtering power of any signature at a higher level is effectively increased. It is because the ratio of 1s

Figure 5. Illustration multilevel signature files



over the signature length at a higher level is kept unchanged. Recall that in an S-tree a signature in an internal node may be populated with more 1s than in a leaf node, decreasing its selectivity significantly.

In the following, we discuss a quite different method, called signature trees, by means of which all the drawbacks of S-trees are removed.

Signature Trees

Consider a signature s_i of length F. We denote it as $s_i = s_i[1]s_i[2] \ldots s_i[F]$, where each $s_i[j] \in \{0, 1\}$ ($j = 1, ..., F$). We also use $s_i(j_1, ..., j_h)$ to denote a sequence of pairs w.r.t. s_i: $(j_1, s_i[j_1])(j_2, s_i[j_2]) \ldots (j_h, s_i[j_h])$, where $1 \le j_k \le F$ for $k \in \{1, ..., h\}$.

Definition 1 (*signature identifier*) Let $S = s_1.s_2 \ldots .s_n$ denote a signature file. Consider s_i ($1 \le i \le n$). If there exists a sequence: $j_1, ..., j_h$ such that for any $k \ne i$ ($1 \le k \le n$) we have $s_i(j_1, ..., j_h) \ne s_k(j_1, ..., j_h)$, then we say $s_i(j_1, ..., j_h)$ identifies the signature s_i or say $s_i(j_1, ..., j_h)$ is an identifier of s_i.

Definition 2 (*signature tree*) A signature tree for a signature file $S = s_1.s_2 \ldots .s_n$, where $s_i \ne s_j$ for $i \ne j$ and $|s_k| = F$ for $k = 1, ..., n$, is a binary tree T such that

]

1. For each internal node of T, the left edge leaving it is always labeled with 0 and the right edge is always labeled with 1.
2. T has n leaves labeled 1, 2, ..., n, used as pointers to n different positions of $s_1, s_2 \ldots$ and s_n in S (signature file). For a leaf node u, $p(u)$ represents the pointer to the corresponding signature in S.
3. Each internal node v is associated with a number, denoted $sk(v)$ which is the bit offset of a given bit position in the block signature pattern. That bit position will be checked when v is encountered.
4. Let $j_1, ..., j_h$ be the numbers associated with the nodes on a path from the root to a leaf node labeled i (then, this leaf node is a pointer to the ith signature in S). Let $p_1, ..., p_h$ be the sequence of labels of edges on this path. Then, $(j_1, p_1) \ldots (j_h, p_h)$ makes up a signature identifier for s_i, $s_i(j_1, ..., j_h)$.

Example 1. In Fig. 6(b), we show a signature tree for the signature file shown in Fig. 4(a). In this signature tree, each edge is labeled with 0 or 1 and each leaf node is a pointer to a signature in the signature file. In addition, each internal node is associated with a positive integer (which is used to tell how many bits to skip when searching). Consider the path going through the nodes marked 1,

7 and 4. If this path is searched for locating some signature *s*, then three bits of *s*: *s*[1], *s*[7] and *s*[4] will have been checked at that moment. If *s*[4] = 1, the search will go to the right child of the node marked "4". This child node is marked with 5 and then the 5th bit of *s*: *s*[5] will be checked.

See the path consisting of the dashed edges in Fig. 4(b), which corresponds to the identifier of s_6: $s_6(1, 7, 4, 5) = (1, 0)(7, 1)(4, 1)(5, 1)$. Similarly, the identifier of s_3 is $s_3(1, 4) = (1, 1)(4, 1)$ (see the path consisting of the thick edges in Fig. 4(b)).

Now we discuss how to search a signature tree to model the behavior of a signature file as a filter. Let s_q be a query signature. The *i*-th position of s_q is denoted as $s_q(i)$. During the traversal of a signature tree, the inexact matching is defined as follows:

(i) Let *v* be the node encountered and $s_q(i)$ be the position to be checked.
(ii) If $s_q(i) = 1$, we move to the right child of *v*.
(iii) If $s_q(i) = 0$, both the right and left child of *v* will be visited.

In fact, this definition just corresponds to the signature matching criterion.

Example 2 Consider the signature file and the signature tree shown in Fig. 6(b) once again.

Assume s_q = 000 100 100 000. We search the signature tree top-down. First, we check the root *r*. Since *sk*(*r*) = 1, we check the 1st bit in s_q. It is 0. Then, both the child nodes of *r* will be explored. (See the thick edges in Fig. 6(c).) When we visit the left child node *v* of *r*, the 7th bit in s_q will be checked since *sk*(*v*) = 7. It is equal to 1. Then, only the right child node of *v* will be checked. We repeat this process until all the possible leaf nodes are visited. Obviously, this process is much more efficient than a sequential searching. For this example, only 42 bits are checked (6 bits during the tree search and 36 bits during the signature checking). But by the scanning of the signature file, 96 bits will be checked. In general, if a signature file contains *N* signatures, the method discussed above requires only $O(N/2^l)$ comparisons in the worst case, where *l* represents the number of bits set in s_q, since each bit set in s_q will prohibit half of a subtree from being visited. Compared to the time complexity of the signature file scanning $O(N)$, it is a major benefit.

Due to limitation of space, we don't discuss here the maintenance of signature trees. An interested reader is referred to (Chen, 2004) for detailed description.

Balanced Signature Trees

A signature tree can be quite skewed as shown in Fig. 7.

But the tree shown in Fig. 7(c) is completely balanced for the same signature file. However, the signature identifiers for the signatures are different from those shown in Fig. 7(b).

The problem is how to control the process of constructing a signature tree in such a way that the generated signature tree is almost balanced.

In the following, we propose a weight-based method, which needs more time than the method discussed above, but always returns a balanced tree.

Weight-Based Method

An observation shows that a signature tree for a signature file corresponds to a recursive partitioning of the file. For instance, the left and right subtrees of the root divides the signature file into two parts: *A* and *B*. If $|A| = |B|$, we say the signature file is evenly divided. Again, each of *A* and *B* is further divided along the signature tree. If the partitioning at each level is even, then the corresponding signature tree must be well balanced and its height is on the order of $O(\log n)$.

Based on the above observation, we propose a method to recursively divide a signature file in such a way that any time a sub-file is partitioned

Figure 6. Signature file, signature tree, and signature tree searching

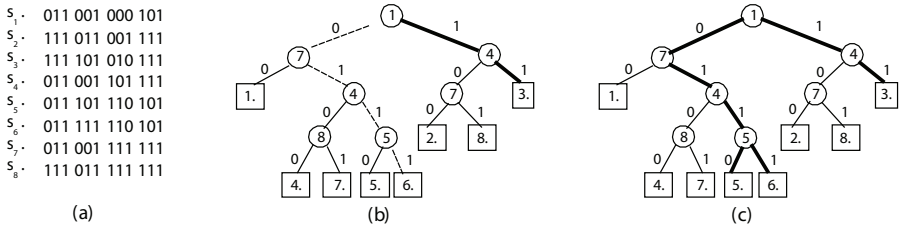
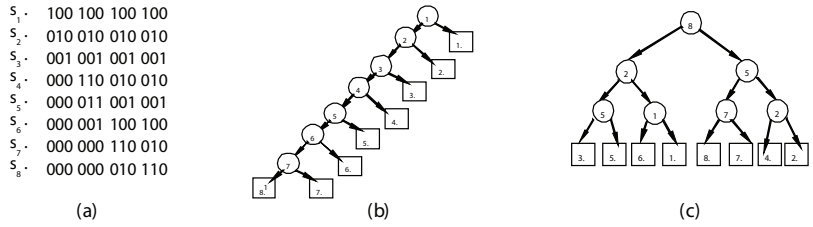


Figure 7. Signature file and signature trees



as evenly as possible. The signature tree is created as a data structure 'recording' this process.

Intuitively, a signature file $S = s_1.s_2 \ldots .s_n$ can be considered as a boolean matrix. We use $S[i]$ to represent the ith column of S. We calculate the weight of each $S[i]$, *i.e.*, the number of 1s appearing in $S[i]$, denoted $w(S[i])$. This needs O(n m) time. Then, we choose an j such that $|w(S[i]) - \frac{1}{2} n|$ is minimum. Here, the tie is resolved arbitrarily. Using this j, we divide S into two groups $g_1 = \{ s_{i_1}, s_{i_2}, \ldots, s_{i_k} \}$ with each $s_{i_p}[j] = 0$ ($p = 1, \ldots, k$) and $g_2 = \{ s_{i_{k+1}}, s_{i_{k+2}}, \ldots, s_{i_N} \}$ with each $s_{i_q}[j] = 1$ ($q = k + 1, \ldots, n$); and generate a tree as shown in Fig. 8(a). In a next step, we consider each g_i ($i = 1, 2$) as a single signature file and perform the same operations as above, leading to two trees generated for g_1 and g_2, respectively. Replacing g_1 and g_2 with the corresponding trees, we get another tree as shown in Fig. 8(b). We repeat this process until the leaf nodes of a generated tree cannot be divided any more.

In Fig. 12(a), $g_1 = \{s_1, s_3, s_5, s_6\}$ and $g_2 = \{s_2, s_4, s_7, s_8\}$; and in In Fig. 12(b), $g_{11} = \{s_3, s_5\}$, $g_{12} = \{s_6, s_1\}$, $g_{21} = \{s_8, s_7\}$, and $g_{22} = \{s_4, s_2\}$.

Exhibit 1 is a formal description of the above process.

By applying this algorithm to the signature file shown in Fig. 10(a), a balanced signature tree as shown in Fig. 911will be created. Since O(n·m) time is needed to generate the nodes at each level of the tree, the time complexity of the whole process is on the order of O(n·m· log$n$).

Figure. 8. Illustration for generation of balanced signature trees

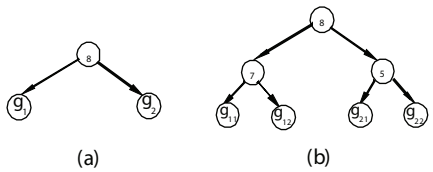


Exhibit 1.

Algorithm *balanced-tree-generation*(*file*)
input: a signature file.
output: a signature tree.
begin
let S = *file*; $N \leftarrow |S|$;
if $N > 1$ **then** {

choose j such that $|w(S[i]) - \frac{1}{2} N|$ is minimum;

let $g_1 = \{ S_{i_1}, S_{i_2}, ..., S_{i_k} \}$ with each $S_{i_p}[j] = 0$ ($p = 1, ..., k$);

let $g_2 = \{ S_{i_{k+1}}, S_{i_{k+2}}, ..., S_{i_N} \}$ with each $S_{i_q}[j] = 1$ ($q = k + 1, ..., N$)
generate a tree containing a root r and two child nodes marked with g_1 and g_2, respectively;
skip(r) $\leftarrow j$;
replace the node marked g_1 with *balanced-tree-generation*(g_1);
replace the node marked g_2 with *balanced-tree-generation*(g_2);}
else return;
end

FUTURE TREND

As discussed above, the signature file is a useful technique for the text indexing and query evaluation in databases. It can also be utilized for some problems in the computational graph theory, i.e., for the tree inclusion problem. As a future research, we will concentrate on an interesting issue: how to reduce the size of a signature file. This may be done by elaborating equation (1) shown in section Signature Files, which is only an empirical formula. What we want is to find a mathematical method to determine, for a given set of key words which are distributed in a collection of blocks, the minimum length of the signatures and the best choice of the number of bits that are set to 1.

CONCLUSION

In this paper, four methods for constructing signature files are described. They are the sequential signature file, the bit-slice signature file, the S-tree and the signature tree. Among these methods, the signature file has the simplest structure and is easy to maintain, but slow for information retrieval. In contrast, the bit-sliced file and the S-tree are efficient for searching, but need more time for maintenance. In addition, an S-tree needs much more space than a sequential signature file or a bit-slice file. The last method, i.e., the signature tree structure, improves the S-tree by using less space for storage and less time for searching. Finally, as an important application, the signatures can be integrated into the top-down tree inclusion strategy to speed up the evaluation of containment queries. This can also be considered as a quite different way to organize a signature file.

REFERENCES

Alonso, L. and Schott, R. (1993). On the tree inclusion problem. In *Proceedings of Mathematical Foundations of Computer Scienc*e, 211-221.

Andre-Joesson, H. and Badal, D. (1997). Using signature files for querying time-series data. In *Proc. 1st European Symp. on Principles of Data Mining and Knowledge Discovery.*

Chang, W.W. and Schek, H.J. (1989). A signature access method for the STARBURST database system. In *Proc. 19th VLDB Conf.*, 145-153.

Chen, Y. (2002). Signature files and signature trees. *Information Processing Letters,* 82(4):213-221, 2002.

Chen, Y. (2004). Building Signature Trees into OODBs. *Journal of Information Science and Engineering,* 20: 275-304.

Chen, Y. (2005). On the signature trees and balanced signature trees. *Processings of 21th Conf. on Data Engineering,* Tokyo, Japan, April 2005, pp. 742-753.

Christodoulakis, S., Theodoridou, M., Ho, F., Papa, M. and Pathria, A. (1986). Multimedia document presentation, information extraction and document formation in MINOS - A model and a system. *ACM Trans. Office Inform. Systems,* 4(4), 345-386.

Christodoulakis, S. and Faloutsos, C. (1984). "Design consideration for a message file server," *IEEE Trans. Software Engineering,* 10(2) (1984) 201-210.

Ciaccia, P. and Zezula, P. (1996). Declustering of key-based partitioned signature files. *ACM Trans. Database Systems,* 21(3), 295-338.

Deppisch, U. (1986). S-tree: A Dynamic Balanced Signature Index for Office Retrieval. *ACM SIGIR Conference,* 77-87.

Faloutsos, C. (1985). Access Methods for Text. *ACM Computing Surveys,* 17(1), 49-74.

Faloutsos, C. and Chan, R. (1988). Fast Text Access Methods for Optical and Large Magnetic Disks: Designs and Performance Comparison, in: *Proc. 14th Intl. Conf. on VLDB,* Aug. 1988, pp. 280-293.

Faloutsos, C., Lee, R., Plaisant, C. and Shneiderman, B. (1990). Incorporating string search in hypertext system: User interface and signature file design issues. *HyperMedia,* 2(3), 183-200.

Faloutsos, C. (1992). Signature Files. In: *Information Retrieval: Data Structures & Algorithms,* edited by W.B. Frakes and R. Baeza-Yates, Prentice Hall, New Jersey, 44-65.

Ishikawa, Y., Kitagawa, H. and Ohbo, N. (1993). Evaluation of signature files as set access facilities in OODBs. In *Proc. of ACM SIGMOD Int. Conf. on Management of Data,* Washington D.C., 247- 256.

Kocberber, S., and Can, F. (1999). Compressed Multi-Framed Signature Files: An Index Structure for Fast Information Retrieval, in: *Proc. ACM SAC'99,* Texas, USA, 1999, pp. 221-226.

Kilpelainen, P. and Mannila, H (1995). Ordered and unordered tree inclusion. *SIAM Journal of Computing,* 24:340-356.

Lee, W. and Lee, D.L. (1992). Signature File Methods for Indexing Object-Oriented Database Systems. *Proc. ICIC'92 - 2nd Int. Conf. on Data and Knowledge Engineering: Theory and Application*, Hong Kong, 616-622.

Lee, D.L, Kim, Y.M., and Patel, G. (1995). Efficient Signature File Methods for Text Retrieval, *IEEE Transaction on Knowledge and Data Engineering*, Vol. 7, NO. 3, June pp.423-435.

Richter, T. (1997). A new algorithm for the ordered tree inclusion problem. In *Proceedings* of the 8th Annual Symposium on Combinatorial Pattern Matching (CPM), in Lecture *Notes of Computer Science (LNCS), volume 126*4, 150-166.

Sacks-Davis, R., Kent, A., Ramamohanarao, K., Thom, J. and Zobel, J. (1995). Atlas: A nested relational database system for text application. *IEEE Trans. Knowledge and Data Engineering,* 7(3), 454-470.

Tousidou, E., Bozanis, P. and Manolopoulos, Y. (2002). Signature-based structures for objects with set-values attributes. *Information Systems,* 27(2):93-121.

Yong, H.S., Lee, S. and Kim, H.J. (1994). Applying Signatures for Forward Traversal Query Processing in Object-Oriented Databases. *Proc. of 10th Int. Conf. on Data Engineering*, Houston, Texas, 518-525.

KEY TERMS

Bit-Slice Signature File: A bit slice file is a file, in which one bit per signature for all the signatures is stored. For a set of signatures of length *F*, *F* bit slice files will be generated.

Multilevel Signature File: A method discussed follows the same principle as the S-tree, but different in that a signature at a higher level is a superimposed code generated *directly* from a group of text blocks, instead of superimposing signatures at a lower level.

S-Tree: An S-tree is a height balanced multiway tree. Each internal node corresponds to a page, which contains a set of signatures and each leaf node contains a set of entries of the form <*s*, *oid*>, where the object is accessed by the *oid* and *s* is its signature.

Sequential Signature File: A signature file is a set of signatures stored in a file in a sequential way.

Signature Identifier: A signature identifier for a signature in a signature file is a positioned bit string which can be used to identify it from others.

Signature Tree: A signature tree is an index structure, in which each path represents a signature identifier for the signature pointed to by the corresponding leaf node.

Signatures: A bit string generated for a key word by using a hash function.

Chapter LXX
On the Query Evaluation in XML Databases

Yanjun Chen
University of Winnipeg, Canada

INTRODUCTION

With the growing importance of XML in data exchange, much research has been done in providing flexible query mechanisms to extract data from XML documents. A core operation for XML query processing is to find all occurrences of a twig pattern *Q* (or small tree) in a document *T*. Prior work has typically decomposed *Q* into binary structural relationships, such as parent-child and ancestor-descendant relations, or root-to-leaf paths. The twig matching is achieved by: (a) matching the binary relationships or paths against XML databases, and (b) using the join algorithms to stitch together all the matching binary relationships or paths. In the worst case, the time for doing joins can be exponential (in the number of query nodes or decomposed paths). In this chapter, we discuss a new algorithm for this task with no path joins involved. The time and space complexities of the algorithm are bounded by O ($|T|\cdot Q_{leaf}$) and O ($T_{leaf}\cdot Q_{leaf}$), respectively, where T_{leaf} stands for the number of the leaf nodes in *T* and Q_{leaf} for the number of the leaf nodes in *Q*. Our experiments show that our method is efficient in supporting twig pattern queries.

BACKGROUND

In an XML databases, a set of XML documents is maintained. Abstractly, each document can be considered as a tree structure with each node labeled with an element name from a finite alphabet $\sum$ or a string value, and an edge for the elment-subelement relationship. For instance, the XML document shown in Figure. 1(a) can be represented a tree structure as shown in Figure 1(b).

Queries in XML query languages, such as XPath (Florescu and Kossman, 1999; World Wide Web Consortium, 2007), XQuery (World Wide Web Consortium, 2001), XML-QL (Dutch, Fernandez, Florescu, 1999), and Quilt (Chamberlin,

Figure 1. A sample XML file and its tree representation

```
<hotel-room-reservation filecode="1302">
    <name>Travel-lodeg</name>
    <location>
        <city-or-district>Winnipeg</city-or-district>
        <state>Manitoba</state>
        <country>Canada</country>
        <address>
            <number>500</number>
            <street>Portage Ave.</street>
            <post-code>R3B 2E9</post-code>
        </address>
    </location>                                        (a)
    <type>
        <room>one-bed-room</room>
        <price>$119.00</price>
    </type>
    <reservation-time>
        <from>April 28, 2007</from>
        <to>May 01, 2007</to>
    </ reservation-time>
</hotel-room-reservation>
```

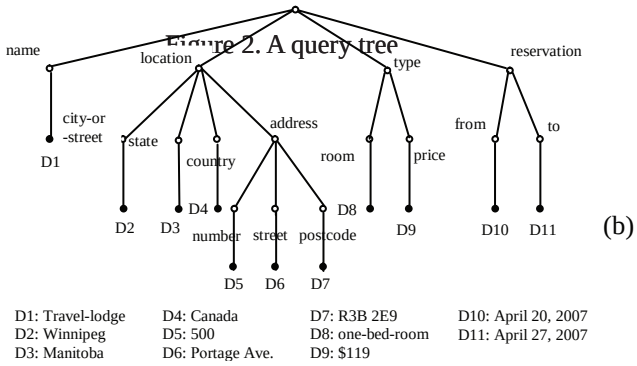


Figure 2. A query tree

et al., 1999; Chamberlin, et al., 2000), typically specify patterns of selection predicates on multiple elements that also have some specified tree structured relations. As an example, consider the query tree shown in Figure 3.

This query asks for any node labeled with *location* (node 2) that is a child of some node labeled with *hotel-room-reservation* (node 1). In addition, *location* node (node 2) is the parent of some *city* node (node 4) and an ancestor of the text '*Portage Ave.*' (node 5). *location* (node 3) can also be the parent of some *address* node (node 7).

The query corresponds to the following XPath expression:

hotel-room[*location*[*city* and //'500 Portage Ave.']]/*location*[*city* and *address*//'500 Portage Ave.'].

In Figure 3, there are two kinds of edges: child edges (*c*-edges) for parent-child relationships, and descendant edges (*d*-edges) for ancestor-descendant relationships. A *c*-edge from node *v* to node *u* is denoted by $v \rightarrow u$ in the text, and represented by a single arc; *u* is called a *c-child* of *v*. A *d*-edge is denoted $v \Rightarrow u$ in the text, and represented by a double arc; *u* is called a *d-child* of *v*. Such a query is often called a twig pattern. In addition, a node in *Q* can be a wildcard '*' that matches any type in *T*.

In any DAG (*directed acyclic graph*), a node *u* is said to be a descendant of a node *v* if there exists a path (sequence of edges) from *v* to *u*. In the case of a twig pattern, this path could consist of any sequence of *c*-edges and/or *d*-edges. Based on these concepts, the tree embedding can be defined as follows.

Definition 1. (*tree embedding*) An embedding of a tree pattern *Q* into an XML document *T* is a mapping f: $Q \rightarrow T$, from the nodes of *Q* to the nodes of *T*, which satisfies the following conditions:

(i) Preserve node label: For each $u \in Q$, *u* and $f(u)$ are of the same label (i.e., $label(u) = label(f(u))$).

(ii) Preserve *c*/*d*-child relationships: If $u \rightarrow v$ in *Q*, then $f(v)$ is a child of $f(u)$ in *T*; if $u \Rightarrow v$ in *Q*, then $f(v)$ is a descendant of $f(u)$ in *T*.

If there exists a mapping from *Q* into *T*, we say, *Q* can be imbedded into *T*, or say, *T* contains *Q*.

Notice that an embedding could map several nodes of the query (of the same type) to the same

Figure 2. A query tree

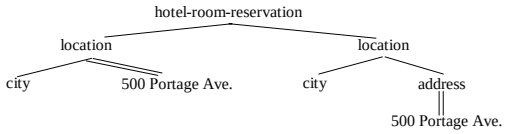
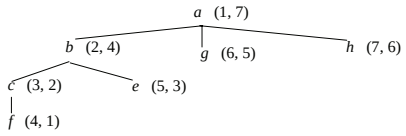


Figure 3. Labeling a tree



node of the database. It also allows a tree mapped to a path. This definition is quite different from the tree matching defined in (Hoffmann and O'Donnell, 1982).

There is much research on how to find such a mapping efficiently and almost all the proposed methods can be categorized into two groups. By the first group (Abiteboul, et al., 1999; Choi, et al., 2003; Cooper, et al., 2001; McHugh and Widom, 1999; Li, and Moon, 2001; Wang, et al., 2003; Wang, and Meng, 2005; Zhang,et al., 2001; Kaushik, et al., 2002; Schmidt, et al., 2001), a tree pattern is typically decomposed into a set of binary relationships between pairs of nodes, such as parent-child and ancestor-descendant relations. Then, an index structure is used to find all the matching pairs that are joined together to form the final result. By the second group (Bruno, et al., 2002; Chen, et al., 2005; Chen. et al., 2006; Lu, et al., 2005; Seo, C., Lee, et al., 2003), a query pattern is decomposed into a set of paths. The final result is constructed by joining all the matching paths together. For all these methods, the join operations involved require exponential time in the worst case. For example, if we decompose a twig pattern into paths to find all the matching paths from a database, we need $O(p^{\lambda})$ time to join them together, where p is the largest length of a matching path and λ is the number of all such paths.

TREE ENCODING

In [17], a tree encoding was proposed, which can be used to facilitate the checking of reachability (whether a node can be reached from another node through a path).

Consider a tree *T*. By traversing *T* in *preorder*, each node *v* will obtain a number *pre*(*v*) to record the order in which the nodes of the tree are visited. In a similar way, by traversing *T* in *postorder*, each node *v* will get another number *post*(*v*). These two numbers can be used to characterize the ancestor-descendant relationships as follows.

Let *v* and *v'* be two nodes of a tree *T*. Then, *v'* is a descendant of *v* iff *pre*(*v'*) > *pre*(*v*) and *post*(*v'*) < *post*(*v*). See Exercise 2.3.2-20 in (Knuth, 1969).

As an example, have a look at the pairs associated with the nodes of the tree shown in Figure 3. The first element of each pair is the preorder number of the corresponding node and the second is its postorder number. Using such labels, the ancestor-descendant relationships can be easily checked.

For instance, by checking the label associated with *b* against the label for *f*, we see that *b* is an ancestor of *f* in terms of Proposition 1. Note that *b*'s label is (2, 4) and *f*'s label is (4, 1), and we have 2 < 4 and 4 > 1. We also see that since the pairs associated with *g* and *c* do not satisfy the condition given in Proposition 1, *g* must not be an ancestor of *c* and *vice versa*.

Let (*p*, *q*) and (*p'*, *q'*) be two pairs associated with nodes *u* and *v*, respectively. We say that (*p'*, *q'*) is subsumed by (*p*, *q*), denoted (*p'*, *q'*) — (*p*, *q*), if *p'* > *p* and *q'* < *q*. Then, *u* is an ancestor of *v* if (*p'*, *q'*) is subsumed by (*p*, *q*).

In addition, if *p'* < *p* and *q'* < *q*, *u* is to the left of *v*.

Finally, we can associate each node v with a level number $l(v)$ (the nesting depth of the element in a document). In conjunction with the tree encoding, this number can be utilized to tell whether a node is the parent of another node. For example, if $pre(v') > pre(v)$, $post(v') < post(v)$ and $l(v) = l(v') + 1$, then v' is a child node of v.

ALGORITHM FOR SIMPLE CASES

In this section, we describe an algorithm for simple cases that a twig pattern contains only *d*-edges, wildcards and branches. First, we give a basic algorithm to show the main idea of our method in 3.1. Then, in 3.2, we prove the correctness of the algorithm and analyze its computational complexities.

Basic Algorithm

The basic algorithm to be given works bottom-up. During the process, two data structures are maintained and computed to facilitate the discovery of subtree matchings according to Definition 1.

- Each node v in T is associated with a set, denoted $\alpha(v)$, contains all those nodes q in Q such that $Q[q]$ can be imbedded into $T[v]$.
- Each q in Q is associated with a value $\delta(q)$, defined as follows.

Initially, for each $q \in Q$, $\delta(q)$ is set to φ. During the tree matching process, $\delta(q)$ is dynamically changed as below.

(i) Let v be a node in T with parent node u.
(ii) If q appears in $\alpha(v)$, change the value of $\delta(q)$ to u.

Then, each time before we insert q into $\alpha(v)$, we will do the following checkings:

1. Check whether $label(q) = label(v)$.
2. Let $q_1, ..., q_k$ be the child nodes of q. For each q_i ($i = 1, ..., k$), check whether $\delta(q_i)$ is equal to v.

If both (1) and (2) are satisfied, insert q into $\alpha(v)$.

Below is the algorithm, working in a recursive way and taking a node v in T as the input (which represents $T[v]$). Initially, the input is the root of T. The algorithm will mark any node u in $T[v]$ if it finds that $T[u]$ contains Q. In the process, two functions are called:

- *node-check*(u, q): It checks whether $T[u]$ contains $Q[q]$. If it is the case, return $\{q\}$. Otherwise, it returns an empty set $\varnothing$.
- *leaf-node-check*(u): It returns a set of leaf nodes in Q: $\{q_1, ..., q_k\}$ such that for each q_i $(1 \le i \le k)$ $label(u) = label(q_i)$.

The algorithm *tree-matching*( ) searches T bottom-up in a recursive way (see line 4). During the process, for each encountered node v in T, we first check whether it is a leaf node (see line 2). If it is a leaf node, the function *leaf-node-check*( ) is called (see line 12), by which all the matching leaf nodes in Q will be stored in a temporary variable S_2 that will be added to $\alpha(v)$ (see line 13). If v is an internal node, lines 3 - 11 are first conducted and then the function *leaf-node-check*( ) is invoked. By executing line 4, *tree-matching*( ) is recursively called for each child node v_i of v. After that, for each q appearing in $\alpha(v_i)$, its δ value is set to be v (see line 7). In addition, q's parent is inserted into S, another temporary valuable to be used in a next step. Since $\alpha(v_i)$'s will not be used any more after this step, they are simply removed (see line 8). By executing lines 9 - 10, we check, for each q in S, whether $T[v]$ contains $Q[q]$ by calling *node-check*(v, q), in which the δ values of q's child nodes are utilized to facilitate the checkings (see lines 3 - 5 in *node-check*( )).

Algorithm *tree-matching*(v)
input: v - a node of tree T.
output: mark any node u in $T[v]$ if $T[u]$ contains Q.
begin
1. $S := \emptyset$; $S_1 := \emptyset$; $S_2 := \emptyset$;
2. **if** v is not a leaf node in T **then**
3. {let v_1, ..., v_k be the child nodes of v;
4. **for** i = 1 to k **do** call *tree-matching*(v_i);
5. $\alpha := \alpha(v_1) \cup ... \cup \alpha(v_k)$;
6. **for** each $q \in \alpha$ **do**
7. {$\delta(q) := v$; $S := S \cup \{q$'s parent$\}$;}
8. remove all $\alpha(v_j)$ (j = 1, ..., k);
9. **for** each q in S do
10. $S_1 := S_1 \cup$ *node-check*(v, q);
11. }
12. $S_2 :=$ *leaf-node-check*(v);
13. $\alpha(v) := \alpha \cup S_1 \cup S_2$;
end

The following example helps for illustration.

Example 1. Consider T and Q shown in Figure 4.

The algorithm works in a bottom-up way. First, v_3 in T is visited. It is a leaf node. Among the two leaf nodes in Q, q_3 matches it. Therefore, $\alpha(v_3) = \{q_3\}$ (see lines 12). In the same way, we will get $\alpha(v_5) = \{q_2\}$. In a next step, v_4 is encountered. It is the parent of v_5. In terms of $\alpha(v_5) = \{q_2\}$, $\delta(q_2)$ is set to be v_4. After that, *node-check*(v_4, q_1) is invoked. Since $type(v_4) \neq type(q_1)$, it returns $S_1 = \emptyset$. *leaf-node-check*(v_4) also returns $S_1 = \emptyset$. So $\alpha(v_4) = \{q_2\}$ (see line 13). When v_2 is met, we will first set $\delta(q_2) = \delta(q_3) = v_2$ (in terms of $\alpha(v_4) = \{q_2\}$ and $\alpha(v_3) = \{q_3\}$). Next, we call *node-check*( ), in which we will check whether $type(v_2) = type(q_1)$. It is the case. So we will check whether $\delta(q_i)$ (i = 2, 3) is equal to v_2 or a descendant of v_2. Since both $\delta(q_2)$ and $\delta(q_3)$ are equal to v_2, we have $T[v_2]$ contains $Q[q_1]$. So we will set $\alpha(v_2) = \{q_1\}$. See Figure 5 for illustration.

In a next step, we will meet v_6. It is a leaf node, matching q_3. Therefore, $\alpha(v_6) = \{q_3\}$. Finally, we will meet v_1 and set $\delta(q_1) = v_1$ and $\delta(q_3) = v_1$. Since $type(v_1) = type(q_1)$, $\delta(q_2) = v_4$ (a descendant of v_1) and $\delta(q_3) = v_1$, we set $\alpha(v_1) = \{q_1\}$.

Figure 4. A document tree and a query tree

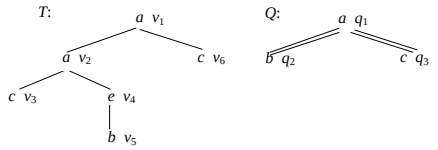


Figure 5. Sample trace

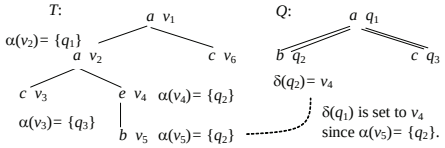
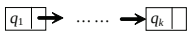


Figure 6. A linked list



Correctness and Computational Complexity

In this subsection, we prove the correctness of *tree-matching*( ) and analyze its computational complexities.

Proposition 1. Let v be a node in T. Then, for each q in $\alpha(v)$ generated by *tree-matching*( ), we have that $T[v]$ contains $Q[q]$.

Proof. We prove the proposition by induction on the height of Q, *height*(Q).

Basic step. When *height*(Q) = 1, the proposition trivially holds.

Induction step. Assume that the proposition holds for any query tree Q' with $height(Q') \le h$. We consider a query tree Q of height $h+1$. Let r_Q be the root of Q. Let $q_1, ..., q_k$ be the child nodes of r_Q. Then, we have $height(Q[q_j]) \le h$ ($j = 1, ..., k$). In terms of the induction hypothesis, for each q in $Q[q_j]$ ($j = 1, ..., k$), if it appears in $\alpha(v_i)$ (where v_i is a child node of v), we have $T[v_i]$ contains $Q[q]$ and $\delta(q)$ will be set to be v. Especially, if $T[v_i]$ contains $Q[q_j]$ ($j = 1, ..., k$), we must have $q_j \in \alpha(v_i)$ and $\delta(q_j)$ will be set to be v before v is checked against r_Q. Obviously, if $label(v) = label(r_Q)$ and for each q_j ($j = 1, ..., k$), $\delta(q_j)$ is equal to v, Q can be embedded into $T[v]$. So r_Q will be inserted into $\alpha(v)$.

Now we consider the time complexity of the algorithm, which can be divided into four parts:

1. The first part is the time spent on unifying $\alpha(v_1), ..., \alpha(v_k)$, where v_i ($i = 1, ..., k$) is a child node of some node v in T. This part of cost is bounded by

   $$O(\sum_{i}^{|T|} d_i |Q|) = O(|T||Q|),$$

   where d_i represents the ourdegree of a node v_i in T.
2. The second part is the time used for generating S from α ($= \alpha(v_1) \cup ... \cup \alpha(v_k)$). Since the size of α is bounded by $O(|Q|)$, so this part of cost is also bounded by $O(|Q|)$.
3. The third part is the time for checking a node v_i in T against each node q_j in an S. This can be estimated by the following sum:

   $$O(\sum_{i}^{|T|}\sum_{j}^{|S|} c_j) \le O(\sum_{i}^{|T|}\sum_{j}^{|Q|} c_j) = O(|T||Q|),$$

   where c_j represents the ourdegree of a node q_j in S.
4. The fourth part is the time for checking each node in T against the leaf nodes in Q. Obviously, this part of cost is bounded by

   $$O(\sum_{i}^{|T|} |Q|) = O(|T||Q|).$$

In terms of the analysis, we have the following proposition.

Proposition 2. The time complexity of *tree-matching*( ) is bounded by $O(|T||Q|)$.

Proof. See the above discussion.

However, this computational complexity can be improved by reducing the size of each $\alpha(v)$. Assume that q and q' are two query nodes appearing in $\alpha(v)$. If q' is subsumed by q, then we can remove q' from $\alpha(v)$ since the containment of $Q[q]$ in $T[v]$ implies the containment of $Q[q']$ in $T[v]$. This can be done as follows.

First of all, we notice that the algorithm searches T bottom-up. For a leaf node v in T, $\alpha(v)$ is initialized with all those leaf nodes in Q, which match v. This can be carried out by searching the leaf nodes in Q from left to right. Then, for any two leaf nodes q and q' in $\alpha(v)$, if q appears before q', we have that $pre(q) < pre(q')$ and $post(q) < post(q')$. That is, $\alpha(v)$ is initially sorted by the *preorder* and *postorder* values. We can store $\alpha(v)$ as a linked list. Let α_1 and α_2 be two sorted lists with $|\alpha_1| \le Q_{leaf}$ and $|\alpha_2| \le Q_{leaf}$. The union of α_1 and α_2 ($\alpha_1 \cup \alpha_2$) can be performed by scanning both α_1 and α_2 from left to right and inserting the elements in α_2 into α_1 one by one. During this process, any element in α_1, which is subsumed by some element in α_2 will be removed; and any element in α_2, which is subsumed by some element in α_1, will not be inserted into α_1. The result is stored in α_1. Obviously, the resulting linked list is still sorted and its size is bounded by Q_{leaf}. We denote this process as $merge(\alpha_1, \alpha_2)$ and define $merge(\alpha_1, ..., \alpha_{k-1}, \alpha_k)$ to be $merge(merge(\alpha_1, ...\alpha_{k-1}), \alpha_k)$. In this way, the time and space complexities of the algorithm can be improved to $O(|T|Q_{leaf})$ and $O(T_{leaf} \cdot Q_{leaf})$, respectively.

General Cases

The algorithm discussed in Section 3 can be easily extended to general cases that a query tree contains both *c*-edges and *d*-edges, as well as wildcards and branches. (See Exhibit 1.)

Let q_1, ..., q_k be the child nodes of q. Let v_1, ..., v_l be the child nodes of v. If $T[v]$ contains $Q[q]$, the following two conditions must hold:

- for each *c*-edge (q, q_i) $(1 \le i \le k)$, there must exist a v_j $(1 \le j \le l)$ such that (v, v_j) matches (q, q_i), and $T[v_j]$ contains $Q[q_i]$.

In terms of this analysis, we modify Algorithm *tree-matching*( ) as follows.

The first difference of the above algorithm from the algorithm *tree-matching*( ) is that before we set the value for $\delta(q)$ we will check whether q is a *d*-child or a *c*-child. If q is a *c*-child, we will further check whether it matches v_i (see lines 7 - 9). We notice that q appearing in $\alpha(v_i)$ only indicates that $Q[q]$ can be embedded into $T[v_i]$, but not necessarily means that q matches v_i.

The second difference is line 11 and lines 13 - 14. In line 11, we use the merge operation to union $\alpha(v_1)$, ..., and $\alpha(v_k)$ together. In lines 13 -14, we generate a set S that contains the parent nodes of all those nodes appearing in α (= merge($\alpha(v_1)$, ..., $\alpha(v_k)$), where v_j is a child node of the current node v. Since the nodes in α are sorted (according to the nodes' *preorder* and *postorder* values), if there are more than one nodes in α sharing the same parent, they must appear consecutively in the list. So each time we insert a parent node q' (of some q in α) into S, we need to check whether it is the same as the previously inserted one. If it is the case, q' will be ignored. Thus, the size of S is also bounded by $O(Q_{leaf})$.

Concerning the correctness of the algorithm, we have to answer a question: whether any *c*-edge in Q is correctly checked.

First, we note that any *c*-edge in Q cannot be matched to any path with length larger than 1 in T. That is, it can be matched only to a single edge in T. It is exactly what is done by the algorithm.

Each time we check a node v in T against some q in Q, we will first change δ values for any q_i appearing in $\alpha(v_j)$'s, where v_j is a child node of v. If q_i is a *c*-child, $\delta(q_i)$ is changed to be v only if q_i matches v_j. Thus, only for some q_i's, their δ values are changed (to v). Assume that the current δ value for q_i is v' (i.e., $\delta(q_i) = v$'). Then, v' must be a descendant of v since the algorithm searches T in a bottom-up way. However, we need to change $\delta(q_i)$ from v' to v since a *c*-edge can match only a single edge in T and the fact that q_i matches v_j should be recorded so that the *c*-edge matching is not missed.

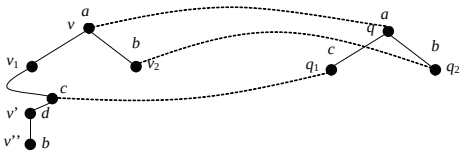
See Figure 7 for illustration.

In Figure 7, v'' is a descendant of v and matches q_2. So $\delta(q_2)$ will be set to v' when v' is encountered. However, (q, q_2) is a *c*-edge. Therefore, the fact that v'' matches q_2 makes no contribution to the

Exhibit 1.

```
Algorithm general-tree-matching(v)
input: v - a node of tree T.
output: mark any node u in T[v] if T[u] contains Q.
begin
1.  S := ∅; S1 := ∅; S2 := ∅;
2.  if v is not a leaf node in T then
3. {  let v1, ..., vk be the child nodes of v;
4.    for i = 1 to k do call tree-matching(vi);
5.    for i = 1 to k do {
6.      for q ∈ α(vi) do {
7.        if ((q is a d-child) or
8.        (q is a c-child and q matches vi))
9.        then δ(q) := v
10.   }}
11.   α := merge(α(v1), ..., α(vk));
12.   assume that α = {q1, ..., qj};
13.   for i = 1 to j do {
14.       if (qi's parent ≠ qi-1's parent)
          then S := S ∪ {qi's parent};}
15.   remove all α(vj) (j = 1, ..., k);
16.   for each q in S do
17.   S1 := S1 ∪ node-check(v, q);
18. }
19. S2 := leaf-node-check(v);
20. α(v) := merge(α, S1, S2);
end
```

Figure 7. Illustration for c-edge checkings



containment of $Q[q]$ in $T[v]$. However, since q_2 also matches v_2, $\delta(q_2)$ will be changed to v, which enables us to find that $T[v]$ contains $Q[q]$.

In conjunction with Proposition 1, the above analysis shows the correctness of the above algorithm. We have the following proposition.

Proposition 3. Let Q be a twig pattern containing both *c*-edges and *d*-edges, as well as wildcards and branches. Let v be a node in T. For each q in $\alpha(v)$ generated by *general*-tree-matching( ), we have that $T[v]$ contains $Q[q]$.

Proof. See the above discussion.

The time and space complexities for the general cases are the same as for the simple cases.

FUTURE WORK

As a future work, we will extend the algorithm to a streaming environment, as found with stock market data, network monitoring or sensor networks. In such an environment, data streams, which can be potentially infinite, arrive continuously and must be processed using a single sequential scan because of the limited storage space available. Query results should be distributed incrementally and as soon as they are found possibly before all the data are read. It is a preorder searching process of trees. However, our algorithm works bottom-up, which barricades a direct application of it to the problem. Therefore, how to incorporate the main idea of our algorithm into a streaming process should further be investigated.

CONCLUSION

In this paper, a new algorithm is proposed for a kind of tree matching, the so-called twig pattern matching. This is a core operation for XML query processing. The main idea of the algorithm is to explore both T and Q bottom-up, by which each node q in Q is associated with a value (denoted $\delta(q)$) to indicate a node v in T, which has a child node v' such that $T[v']$ contains $Q[q]$. In this way, the tree embedding can be checked very efficiently. In addition, by using the tree encoding, as well as the subsumption checking mechanism, we are able to minimize the size of the lists of the matching query nodes associated with the nodes in T to reduce the space overhead. The algorithm runs in $O(|T|\cdot Q_{leaf})$ time and $O(T_{leaf}\cdot Q_{leaf})$ space, where T_{leaf} and Q_{leaf} represent the numbers of the leaf nodes in T and in Q, respectively. More importantly, no costly join operation is necessary.

REFERENCES

Abiteboul, S., Buneman, P., & Suciu, D. (1999). *Data on the web: from relations to semistructured data and XML*. Los Altos, CA: Morgan Kaufmann Publisher.

Bruno, N., Koudas, N., & Srivastava, D. (2002). Holistic twig hoins: Optimal XML pattern matching. In *Proc. SIGMOD Int. Conf. on Management of Data*, 310-321. Madison, Wisconsin.

Chamberlin, D. D., Clark, J., Florescu, D., & Stefanescu, M. (2000). XQuery 1.0: An XML query language. http:/ /www.w3.org/TR/query-datamodel/.

Chamberlin, D. D., Robie, J., & Florescu, D. (2000). Quilt: An XML query language for heterogeneous data sources. *WebDB 2000.*

Chen, T., Lu, J., & Ling, T. W. (2005). On boosting holism in XML twig pattern matching. In *Proc. SIGMOD*, 2005, 455-466.

Choi, B., Mahoui, M., & Wood, D. (2003). On the optimality of holistic algorithms for twig queries. In *Proc. DEXA*, 235-244.

Chung, C., Min, J., & Shim, K. (2002, June). APEX: An adaptive path index for XML data. *ACM SIGMOD.*

Chen, S. (2006). *Twig²Stack*: Bottom-up processing of generalized-tree-pattern queries over XML documents. In *Proc. VLDB*, 283-323. Seoul, Korea.

Cooper, B. F., Sample, N., Franklin, M., Hialtason, A. B., & Shadmon, M. (2001, September). A fast index for semistructured data. In *Proc. VLDB*, 341-350.

Dutch, A., Fernandez, M., & Florescu, D. (1999, May). A query language for XML, A. Levy, D.Suciu (Eds.). In *Proc. 8th World Wide Web Conf.*, 77-91.

Florescu, D., & Kossman, D. (1999). Storing and querying XML data using an RDMBS. *IEEE Data Engineering Bulletin, 22*(3), 27-34, 1999.

Hoffmann, C. M., & O'Donnell, M. J. (1982). Pattern matching in trees. *J. ACM*, 29(1), 68-95.

Knuth, D. E. (1969). *The art of computer programming, 1.* Reading: Addison-Wesley.

Lu, J., Ling, T. W., Chan, C. Y., & Chan, T. (2005). From region encoding to extended dewey: On efficient processing of XML twig pattern matching. In *Proc. VLDB*, 193-204.

McHugh, J., & Widom, J. (1999). Query optimization for XML. In *Proc. of VLDB.*

Seo, C., Lee, S., & Kim, H. (2003). An efficient index technique for XML documents using RDBMS. *Information and Software Technology, 45*(2003), 11-22. B.V.: Elsevier Science.

Li, Q., & Moon, B. (2001, September). Indexing and querying XML data for regular path expressions. In *Proc. VLDB*, 361-370.

Wang, H., Park, S., Fan, W., & Yu, P. S. (2003, June). ViST: A dynamic index method for querying XML data by tree structures. *SIGMOD Int. Conf. on Management of Data*, San Diego, CA..

Wang, H., & Meng, X. (2005, April). On the sequencing of tree structures for XML indexing In *Proc. Conf. Data Engineering*, 372-385. Tokyo, Japan.

World Wide Web Consortium. (2007). *XML path language (XPath). W3C Recommendation*, 2007. See http:// www.w3.org/TR/xpath20.

World Wide Web Consortium. (2001). *XQuery 1.0: An XML Query Language, W3C Recommendation, Version 1.0*, Dec. 2001. See http://www.w3.org/TR/xquery.

Zhang, C., Naughton, J., Dewitt, D., Luo, Q., & Lohman, G. (2001). On supporting containment queries in relational database management systems. In *Proc. of ACM SIGMOD.*

Kaushik, R., Bohannon, P., Naughton, J., & Korth, H. (2002, June). Covering indexes for branching path queries. In *ACM SIGMOD.*

Schmidt, A. R., Waas, F., Kersten, M. L., Florescu, D., Manolescu, I., Carey, M. J., & Busse, R. (2001). The XML benchmark project, Technical Report INS-Ro1o3. *Centrum voor Wiskunde en Informatica.*

KEY TERMS

Node Subsumption: A node labeled with (*p*', *q*') is said to be subsumed by (*p, q*),), if *p*' > *p* and *q*' < *q*.

Tree Embedding: An embedding *f* of a tree pattern *Q* into an XML document *T* satisfying the following conditions:

(i) Preserve node label: For each $u \in Q$, u and $f(u)$ are of the same label (i.e., $label(u) = label(f(u))$).

(ii) Preserve c/d-child relationships: If $u \rightarrow v$ in Q, then $f(v)$ is a child of $f(u)$ in T; if $u \Rightarrow v$ in Q, then $f(v)$ is a descendant of $f(u)$ in T.

Tree Encoding: A way labeling the nodes of a tree, which can be used to identify the relationships between the nodes, such as parent-child, and ancestor-descendant, as well as left-to-right relationships.

Tree Pattern Query: Queries that are based on the path expressions including element tags, attributes, and key words, which are often represented as a tree structure.

XML Database: A database designed for managing and manipulating XML documents or even more generic SGML documents.

XML Document: A document consisting of an (optional) XML declaration, followed by either an (optional) DTD or XML schema, and then followed by a document element.

XML Schema: An alternative to DTDs. It is a schema language that assesses the validity of a well-formed element and attribute information items within an XML document. There are two major schema models: W3C XML Schema and Microsoft Schema.

XPath Expression: An expression contains tree node names separated by 'parent/child' separators '/' or 'ancestor/descendant' separators '//'. A node name may also associated with a predicate.

Chapter LXXI
XML Document Clustering

Andrea Tagarelli
University of Calabria, Italy

INTRODUCTION

The ability of providing a "standardized, extensible means of coupling semantic information within documents describing semistructured data" (Chaudhri, Rashid, & Zicari, 2003) has led to a steady growth of XML (extensible markup language) data sources, so that XML is touted as the driving force for representing and exchanging data on the Web.

The motivation behind any clustering problem is to find an inherent structure of relationships in the data and expose this structure as a set of clusters where the objects within the same cluster are each to other highly similar but very dissimilar from objects in different clusters.

The clustering problem finds in text databases a fruitful research area. Since today semistructured text data has become more prevalent on the Web, and XML is the de facto standard for such data, clustering XML documents has increasingly attracted great attention. Any application domain that needs organization of complex document structures (e.g., hierarchical structures with unbounded nesting, object-oriented hierarchies) as well as data containing a few structured fields together with some largely unstructured text components can be profitably assisted by an XML document clustering task.

In principle, the availability of schemas for XML data may be useful to drive or simplify a clustering task. For instance, in case of structural classification, XML documents with different element values but similar schemas could be grouped together. An XML DTD defines the document schema by means of constraints (element content models) that specify the element types, hierarchical relationships between elements, and other properties such as multiple occurrences of elements (operator +), optional elements (operator ?), and alternate elements (operator |). XML documents available from real data sources tend to have such characteristics.

However, exploiting XML schemas profitably for classification purposes is not always feasible in practice. On one hand, most XML sources provide documents that are schema-less, that is, documents without an explicitly associated element type definition. On the other hand, XML documents available from the same data source may have quite different size and structure mainly due to nesting and repetition of elements, although they conform to a unique DTD. Also, XML documents with different schemas may have similar contents, or, in a more complicated case, XML documents coming from heterogeneous sources may represent semantically related data even if

they use different markup tags that refer to different schemas. The above are main reasons to conceive the task of clustering XML documents without assuming the availability of predefined XML schemas.

Within this view, we aim to provide a broad overview of the state of the art and a guide to recent advances and emerging challenges in the research field of clustering XML documents.

BACKGROUND

Several approaches and methodologies have been proposed in the last years to address the problem of clustering XML documents, and major differences can be found in how the following issues have been settled:

- **Representation model:** A natural interpretation of an XML document is a labeled rooted tree or, if references between elements are included, a labeled, rooted, directed graph. The element and attribute names are mapped to inner nodes (or, alternatively, to edges) of the tree (or graph), and the text sequences enclosed by the innermost elements are assigned to leaf nodes.
- **Features:** An XML feature is a component of the XML representation model and expresses what and how information available in an XML document is considered and modeled as its attribute. Structural information (e.g., element tag names, element location in the tree hierarchy), content information (e.g., textual elements, attribute values), or both information can be involved in XML features.
- **Related tasks**
  - **Pattern matching:** Any mechanism of information retrieval needs a method of locating substructures of a larger structure, the target, by comparing them against a given form called the pattern. In the XML domain, a common problem is tree matching, which is concerned with finding the instances, or matches, of a given pattern tree in a given target tree.
  - **Similarity detection:** Defining a suitable distance or similarity measure between XML documents requires one to consider at least the features upon which documents are identified as similar and the method of pattern matching that is used according to the model chosen for representing an XML document.
  - **Summarization:** The capability of summarizing an XML document or sets of XML documents has a likely impact on the performance of the similarity computation in a clustering task. In particular, XML summaries can help in estimating the selectivity of path expressions and devising indexing techniques for clusters to finally improve the construction of query plans.

In the following we describe major aspects, merits, and shortcomings of significant solutions existing in the current research literature concerning representation and summarization models, feature extraction, similarity computation, and clustering schemes for XML documents. In particular, the focus here is on approaches and methods conceived to address the following main problems: the detection of structural similarities among XML documents, clustering of XML documents by structure, summarization of XML documents, and clustering of XML documents by content.

CLUSTERING XML DOCUMENTS BY STRUCTURE

Clustering XML documents has its roots in the problem of comparing semistructured documents, which originally arises from several applications in the management of semistructured data, such

as the integration of data sources and query processing (Abiteboul, Buneman, & Suciu, 1999; Widom, 1999). Such applications were initially focused on structure information available from documents.

Comparing XML documents represented as labeled rooted trees naturally leverages results from research on tree matching. These results include a number of algorithms for computing a minimum cost sequence of edit operations (insert node, delete node, relabel node) to align a pattern document tree to a target document tree. A well-known fast algorithm for general-ordered labeled tree matching is defined by Zhang and Shasha (1989), which allows edit operations anywhere in a tree and computes the distance between two trees T_1 and T_2 in time $O(|T_1| \times |T_2| \times min\{depth(T_1), leaves(T_1)\} \times min\{depth(T_2), leaves(T_2)\})$ and space $O(|T_1| \times |T_2|)$, where $|T|$, $depth(T)$, and $leaves(T)$ denote the number of nodes, and the depth and the number of leaf nodes, respectively, of tree T. The fastest algorithm available is by Chawathe (1999), which permits insertion and deletion only at leaf nodes. This assumption could fit better in some contexts of XML data where, for instance, it is necessary that constraints on the hierarchical structure of XML trees hold. It is worth noticing that the scenario in which XML documents are modeled as unordered labeled trees is much less considered, mainly because the tree matching problem for unordered labeled trees has been demonstrated to be NP complete (Zhang, Statman, & Shasha, 1992).

Early approaches to detecting structural similarities between XML documents are thus based on tree edit distances. In Nierman and Jagadish (2002), an XML-aware edit distance is exploited in a standard hierarchical clustering algorithm to evaluate how closely cluster documents correspond to their respective DTDs. However, in general, computing tree edit distances turns out to be unpractical, especially for large collections of XML documents, as it requires a quadratic number of comparisons between document elements.

A rather different approach to structural similarity detection has been recently proposed in Flesca, Manco, Masciari, Pontieri, and Pugliese (2005). Here, the structure of an XML document is represented as a time series in which each occurrence of a tag corresponds to an impulse, and the degree of similarity between documents is computed by analyzing the frequencies of the corresponding Fourier transforms. This approach does not consider the element ordering and fares as well as long as a few elements are in common between documents.

Another alternative XML structural representation is introduced in Leung, Chung, Chan, and Luk (2005). The key idea is to represent an XML document by a set of simple XPath expressions, each leading from the document root to a leaf node. Based on this representation model, the XML sequences corresponding to a collection of XML documents represented as XPath expressions are subject to a sequential pattern mining algorithm to compute a set of maximal frequent paths. These paths, called common XPaths, encode the frequently occurring structures across a collection of XML documents and act as features for clustering. Clearly, the common XPath approach requires a user-specified minimum support threshold for computing frequent XML sequences.

Exploiting Structural Summaries

In general, a structural summary is designed to provide a concise representation of XML data while preserving the main structural relationships between XML elements. What main really means depends on the model of summarization but also on the target of summarization, which can be an individual XML document, a set (cluster) of XML documents, or both.

Costa, Manco, Ortale, and Tagarelli (2004) present a tree-based method for clustering XML documents by structure, focusing on a notion of a structure-driven XML cluster representative. A cluster representative is a prototype XML document that is built to capture the most relevant structural features of the documents within a cluster. This is accomplished by exploiting the tree nature of XML documents and providing suitable

strategies for tree matching, merging, and pruning. Tree matching allows the identification of structural similarities to build an initial substructure that is common to all the XML document trees in a cluster. The phase of tree merging leads to an XML tree that even contains uncommon substructures; finally, a pruning strategy serves to minimize the distance between the documents in a cluster and the document to be built as the cluster representative. A hierarchical clustering algorithm exploits the devised notion of a representative to assign XML documents to structurally homogeneous groups in a cluster hierarchy.

Another tree-based, summary-aware framework for clustering XML documents by structure is described in Dalamagas, Cheng, Winkel, and Sellis (2006). XML documents represented as ordered, labeled, rooted trees are individually summarized to reduce nesting and the repetition of elements. Such structural summaries, rather than their original documents, are compared using a tree edit distance computed by a variant of Chawathe's (1999) algorithm. The clustering scheme is based on a traditional graph-theoretic divisive approach. Given a collection of *n* XML documents, a fully connected, weighted graph is built over the *n* structural summaries of the input documents. In this graph, the weight of an edge corresponds to the structural distance between two nodes (structural summaries). A minimum spanning tree (MST) of the graph is computed, and then all the MST edges with a weight above a user-specified threshold are deleted. The connected components of the remaining graph are the single-link clusters.

A graph-based summarization of relevant structural features of XML documents, called an s-graph, is proposed in Lian, Cheung, Mamoulis, and Yiu (2004). Given a set *S* of XML documents, the associated s-graph is built as a directed graph whose nodes correspond to all the elements and attributes in the documents of *S*, and there exists an edge between nodes n_1 and n_2 if and only if, in some document of *S*, n_2 is a child element of n_1 or n_2 is an attribute of n_1. The notion of an s-graph is generalizable to sets of documents, thus the comparison between a document and a target cluster can be easily accomplished by comparing their respective s-graphs.

A major problem with the s-graph approach relies on the loose-grained similarity that may occur. Indeed, two documents can share the same s-graph prototype and still have significant structural differences, such as in hierarchical relationships between elements (e.g., different roots, different elements with multiple occurrences at a certain level, etc.). It is clear that the s-graph approach may fail in dealing with application domains requiring finer structural dissimilarities.

A more refined synopsis model, called XSketch, is described in Polyzotis and Garofalakis (2002a, 2002b) mainly for the purpose of approximating path-expression selections in the general setting of graph-structured XML data, also including element values. The construction of an accurate synopsis is based on some statistical assumptions that compensate for the lack of detailed path and value information in the synopsis. However, building an effective XSketch has been demonstrated to be an NP-hard problem, even for the much simpler structure-only case (Polyzotis & Garofalakis, 2002a); thus, a heuristic synopsis refinement strategy must be used.

CLUSTERING XML DOCUMENTS: THE ROLE OF CONTENT

The need for organizing XML data according to both their content and structural features has become challenging due to the increase of application contexts for which it may be relevant to consider both structure and content of the documents rather than the structure information alone. For instance, XML documents gathered from heterogeneous sources may be characterized through the logical structure underlying the XML document trees and the topics that may be inferred from the textual content associated with XML elements. However, effectively mining XML documents by addressing both structure and content information turns out to be more difficult than the structure-only

case. Indeed, mining XML content inherits some problems faced in traditional knowledge discovery from text data, such as semantic ambiguity, while new ones arise when content is, as usual, contextually dependent of the logical structure.

Viewed in this respect, mining XML data from a structure and content combination viewpoint has still not been deeply investigated, thus current literature on this subject provides much less work than the structure case.

Early attempts are given in Guillaume and Murtagh (2000) and Doucet and Myka (2002). In Guillaume and Murtagh, the clustering of data-centric XML documents is seen as a partitioning problem based on a weighted graph-based representation in which nodes are documents, edges are links between documents, and edge weights are computed by considering keywords in the documents. A naive clustering procedure based on the popular *k*-means algorithm is proposed by Doucet and Myka to cluster homogeneous collections of text-centric XML documents. The vector-space model is used for the representation of documents, and features are selected either as the most significant words or tag names, or a combination of both.

A simple adaptation of the vector-space model to semistructured documents, called the structured link vector model (SLVM), is introduced in Yang and Chen (2002). SLVM represents the leaf elements of an XML document as a collection of vector-space models, that is, as a term-element matrix whose generic $(i,j)^{th}$ component contains the TF-IDF weight[1] of term w_i with respect to element e_j in the context of the specific document. For SLVM, the document similarity is defined by introducing a kernel matrix in the conventional cosine measure. SLVM is clearly conceived for XML content retrieval.

An alternative XML representation is BitCube, a three-dimensional bitmap index of triplets (document, XML element path, word) presented in Yoon, Raghavan, Chakilam, and Kerschberg (2001). BitCube indexes can be manipulated to partition documents into clusters by exploiting bit-wise distance and popularity measures. In order to speed up query answering, slice, dice, and project operations are performed to subcubes resulting from the clustering phase. However, no critical discussion is provided by the authors about possible improvements in document clustering. In general, the approach suffers from typical disadvantages of Boolean representation models, such as the lack of partial matching criteria and natural measures of document ranking.

SEMANTIC CLASSIFICATION OF XML DOCUMENTS

XML allows the definition of semantic markup, that is, customized tags describing the data enclosed by them. Semantic information is given about what kind of data is represented. Thus, the increase of volume and the heterogeneity of application scenarios within XML make data sources exhibit not only different structures and contents, but also make different ways to semantically annotate the data. Markup tags reflect latent subjective factors that brand the authorship in coding information.

As a consequence, differently annotated XML documents may be semantically related at a certain degree. Is the intended meaning of sets of documents in different sources the same? If two related sets of documents do not have exactly the same structure and/or contents, how are they related? Does the one's structure or content subsume the other, or do they overlap in some way? In attempting to answer these semantic questions, most of the available information is of syntactic kind, thus a basic assumption is that if the expected syntactic properties are satisfied then a semantic relationship is discovered.

As XML becomes increasingly widespread, semantic classification arises in the context of XML data management and knowledge discovery as one of the hardest challenges. Indeed, there is an inherent difficulty in devising suitable notions of semantic features and semantic relatedness among XML documents. This is partially due to the weak support offered by traditional XML

representation and summarization models, which are typically based on tree or graph structures. On the other hand, features for XML data should be generated not only by combining (syntactic) structure and content information, but also enriching such information by means of an ontology knowledge base.

CLUSTERING SEMANTICALLY RELATED XML DOCUMENTS

A first important move in this direction is made by Tagarelli and Greco (2004, 2006), who identify the novel problem of clustering semantically related XML documents. The semantic relatedness of XML documents is detected through an in-depth analysis of structure as well as content features generated with the support of lexical ontology knowledge.

Structure analysis concerns XML tag paths and involves a novel word sense disambiguation method to select the most appropriate sense for each tag name in the context of an XML path. This aims to solve the inherent ambiguity in the meaning of some tag names, which is typically expressed in synonymies (e.g., different tags may have related meanings in different contexts) and polysemies (e.g., the same tag can be used with different meanings in different contexts). XML data with related structure semantics can be then grouped together even if they are syntactically different.

Content analysis applies to textual elements (i.e., values of element attributes or #PCDATA contents) and combines syntactic and semantic weighting methods to assess the term relevance. Terms in XML texts are weighted on the basis of statistical criteria such as term density in a local text element and term rarity in the reference text collection, similar to the conventional TF-IDF function. In addition, semantic analysis can be applied to leverage the lexical ambiguity problem by examining the degree of polysemy of terms. Intuitively, the higher the number of meanings of a term, the lower the semantic relevance of the term.

A major novelty of the methodology proposed in Tagarelli and Greco (2004, 2006) is the introduction of an XML representation model that allows for mapping XML document trees into transactional data. In a generic application domain, a transaction data set is a multiset of variable-length sequences of objects with categorical attributes; in the XML domain, transactions substantially represent a collection of XML data and any transaction is a set of items, each of which embeds a distinct combination of both structure and content features of the original XML data.

However, XML documents are not directly transformed into transactional data; rather, they are decomposed on the basis of the notion of tree tuples. Intuitively, given an XML document, a tree tuple is a tree representation of a complete set of distinct concepts that are correlated according to the structure semantics of the original document tree. Tree tuples extracted from the same tree maintain similar or identical structure while reflecting different ways of associating content with structure as they can be naturally inferred from the original tree.

In such a way, tree tuples enable a flat, relational-like XML representation that is well-suited to meet the requirements for clustering semantically related XML documents according to structure and content information. Moreover, a traditional disadvantage of relational data, namely, attributes with missing values, is overcome by transactional data since items of transactions always refer to nonnull values.

FUTURE TRENDS

The superiority of XML in supporting the development of interoperable domain-specific vocabularies for descriptive markup languages has been well confirmed in the last years. However, even though XML markup identifies the underlying meaningful structure of a document, XML itself is not able to formally specify semantic relationships among documents and their elements. The resource description framework (RDF; Beckett,

2004; Hayes, 2004) is the recommendation of the World Wide Web Consortium to model metadata about information resources and encode data semantics. Using XML tags, RDF provides assertions that particular things have properties with certain values, making the underlying semantics of XML data explicit.

The key element in many tasks involving a notion of semantics is represented by ontologies. An ontology is understandable to both humans and machines as it provides a shared conceptualization of a specific domain, thus enabling for machine consumption the objects, properties, relationships, constraints, and textual representations of that domain.

In order to meet the XML semantics challenge, XML data need to be interpreted with respect to ontologies by means of mechanisms of semantic mapping, possibly supported by RDF. Ontology knowledge can be fundamental in improving the detection of semantic relatedness between XML data collections.

The idea behind ontology-based clustering of XML documents is that an application ontology may act as a backbone for any phase of the clustering task. In the preprocessing phase, considering different views onto the data leads to different aggregation levels at which data may be represented, thus reducing high dimensionality. The taxonomy of concepts and their aggregations may be exploited in the mining phase: Documents may be grouped according to different views (e.g., a business view vs. a technical view). Finally, an ontology may be useful in focusing on specific results of a clustering task; that is, results may be provided to a user using an ontology as a scaffold for the presentation interfaces.

CONCLUSION

We reviewed the problem of clustering XML documents and related research work from different perspectives, namely, the representation models, the kind of information used to extract document features, and the tasks. We then motivated the role of semantic relatedness in the context of the classification of XML collections and presented our approach to the problem of clustering semantically related XML documents. Semantic integration and classification of XML data will be one of the most challenging problems for database researchers in semistructured data management to tackle in the near future.

REFERENCES

Abiteboul, S., Buneman, P., & Suciu, D. (1999). *Data on the Web: From relations to semistructured data and XML*. Morgan Kaufmann Publishers.

Beckett, D. (Ed.). (2004). RDF/XML syntax specification. *W3C recommendation.* Retrieved from http://www.w3.org/TR/2004/REC-rdf-syntax-grammar-20040210

Chaudhri, A. B., Rashid, A., & Zicari, R. (Eds.). (2003). *XML data management: Native XML and XML-enabled database systems*. Addison-Wesley.

Chawathe, S. S. (1999). Comparing hierarchical data in external memory. *Proceedings of the International Conference on Very Large Data Bases* (pp. 90-101).

Costa, G., Manco, G., Ortale, R., & Tagarelli, A. (2004). A tree-based approach to clustering XML documents by structure. *Proceedings of the European Conference on Principles and Practice of Knowledge Discovery in Databases* (pp. 137-148).

Dalamagas, T., Cheng, T., Winkel, K. J., & Sellis, T. (2006). A methodology for clustering XML documents by structure. *Information Systems, 31*(3), 187-228.

Doucet, A., & Myka, H. (2002). Naive clustering of a large XML document collection. *Proceedings of the Annual ERCIM Workshop of the Initiative for the Evaluation of XML Retrieval (INEX)* (pp. 81-88).

Flesca, S., Manco, G., Masciari, E., Pontieri, L., & Pugliese, A. (2005). Fast detection of XML structural similarity. *IEEE Transactions on Knowledge and Data Engineering, 17*(2), 160-175.

Guillaume, D., & Murtagh, F. (2000). Clustering of XML documents. *Computer Physics Communications, 127,* 215-227.

Hayes, P. (Ed.). (2004). RDF semantics. *W3C Recommendation.* Retrieved from http://www.w3.org/TR/2004/REC-rdf-mt-20040210

Leung, H., Chung, F., Chan, S., & Luk, R. (2005). XML document clustering using common XPath. *Proceedings of the IEEE International Workshop on Challenges in Web Information Retrieval and Integration* (pp. 91-96).

Lian, W., Cheung, D. W., Mamoulis, N., & Yiu, S.-M. (2004). An efficient and scalable algorithm for clustering XML documents by structure. *IEEE Transactions on Knowledge and Data Engineering, 16*(1), 82-96.

Nierman, A., & Jagadish, H. (2002). Evaluating structural similarity in XML documents. *Proceedings of the ACM SIGMOD International Workshop on the Web and Databases (WebDB)* (pp. 61-66).

Polyzotis, N., & Garofalakis, M. (2002a). Statistical synopses for graph-structured XML databases. *Proceedings of the ACM SIGMOD International Conference on Management of Data* (pp. 358-369).

Polyzotis, N., & Garofalakis, M. (2002b). Structure and value synopses for XML data graphs. *Proceedings of the International Conference on Very Large Data Bases* (pp. 466-477).

Tagarelli, A., & Greco, S. (2004). Clustering transactional XML data with semantically-enriched content and structural features. *Proceedings of the International Conference on Web Information Systems Engineering* (pp. 266-278).

Tagarelli, A. & Greco, S. (2006). Toward semantic XML clustering. *Proceedings of the SIAM International Conference on Data Mining* (pp. 188-199).

Widom, J. (1999). Data management for XML: Research directions. *IEEE Data Engineering Bulletin, 22*(3), 44-52.

Yang, J. W., & Chen, X. O. (2002). A semi-structured document model for text mining. *Journal of Computer Science and Technology, 17*(5), 603-610.

Yoon, J., Raghavan, V., Chakilam, V., & Kerschberg, L. (2001). BitCube: A three-dimensional bitmap indexing for XML documents. *Journal of Intelligent Information Systems, 17*(1), 241-252.

Zhang, K., & Shasha, D. (1989). Simple fast algorithms for the editing distance between trees and related problems. *SIAM Journal on Computing, 18*(6), 1245-1262.

Zhang, K., Statman, R., & Shasha, D. (1992). On the editing distance between unordered labeled trees. *Information Processing Letters, 42*(3), 133-139.

KEY TERMS

Clustering XML Documents by Content: The task of clustering XML documents according to features based on information available from XML content, that is, textual elements and attribute values.

Clustering XML Documents by Structure: The task of clustering XML documents according to features based on information available from XML structure, that is, elements and their (hierarchical) relationships.

Document Clustering: The task of organizing a collection of documents, whose classification is unknown, into meaningful groups (clusters) that are homogeneous according to some notion of proximity (distance or similarity) among documents.

Schema-less Document: A document with no explicitly associated document type definition (schema).

Semantic Relatedness: The state of being related by semantic affinities beyond syntactic correspondence or similarity.

XML Cluster Representative: A prototype XML document capturing the most relevant features of the XML documents assigned to a cluster.

XML Transactional Model: A representation model that allows for mapping XML data to variable-length sequences of items, where each item may bear upon structure information, content information, or both.

Chapter LXXII
Indices in XML Databases

Hadj Mahboubi
University of Lyon (ERIC Lyon 2), France

Jérôme Darmont
University of Lyon (ERIC Lyon 2), France

INTRODUCTION

Since XML (eXtensible Markup Language) (Bray, Paoli, Sperberg-McQueen, Maler & Yergeau, 2004) emerged as a standard for information representation and exchange, storing, indexing, and querying, XML documents have become major issues in database research. Query processing and optimization are very important in this context, and indices are data structures that help enhance performances substantially. Though XML indexing concepts are mainly inherited from relational databases, XML indices bear numerous specificities.

The aim of this chapter is to present an overview of state-of-the-art XML indices and to discuss the main issues, trade-offs, and future trends in XML indexing. Furthermore, since XML is gaining importance for representing business data for analytics (Beyer, Chamberlin, Colby, Özcan, Pirahesh & Xu, 2005), we also present an index we developed specifically for XML data warehouses.

BACKGROUND

Indexing and querying XML documents through path expressions expressed in XPath (Clark & DeRose, 1999) and XQuery (Boag, Chamberlin, Fernandez, Florescu, Robie & Siméon, 2006) have been the focus of many research studies. Two families of approaches aim at efficiently processing path join queries. They are based on structural summaries and numbering schemes, respectively.

Structural Summary-Based Indices

Structural index-based approaches help traverse XML documents' hierarchies by referencing structural information about these documents. These techniques extract structural information directly from data and create a structural summary that is a labeled, directed graph. Graph schemas can be used as indices for path queries. Dataguide (Goldman & Widom, 1997) and 1-index (Milo & Suciu, 1999) belong to this family of indices.

Dataguide's structure describes by one single label all the nodes whose labels (names) are identical. Its definition is based on targeted path sets (i.e., sets of nodes that are reached by traversing a given path).

1-index clusters nodes according to a bisimilarity relationship. Two nodes are said to be bisimilar if they share identical label paths in the XML data graph. Bisimilar nodes are grouped into one index node. A 1-index is smaller than the initial data graph and thereby facilitates query evaluation. To help select labels or evaluate path expressions, hash tables or B-trees are used to index graph labels.

Dataguide and 1-index code all paths from the root node. The size of such summary structures may be larger than the original XML document, which degrades query performance. A(k)-index (Kaushik, Shenoy, Bohannon & Gudes, 2002) is a variant of 1-index based on k-dissimilarity and builds an approximate index to reduce its graph's size. An A(k)-index can retrieve, without referring to the data graph, path expressions of length of at most k, where k controls index resolution and influences index size in a proportional manner. However, for large values of k, index size may still become very large. For small values of k, index size is substantially smaller, but A(k)-index cannot handle long path expressions.

To accommodate path expressions of various lengths without unnecessarily increasing index size, D(k)-index (Qun, Lim & Ong, 2003) assigns different values of k to index nodes. These values conform to a given set of frequently used path expressions (FUPs). Small or large values of k are assigned to index parts that are targeted by short or long path expressions, respectively. To help evaluate path expressions with branching, a variant called UD(k, l)-index (Wu, Wang, Xu Yu, Zhou & Zhou, 2003) also imposes downward similarity.

AD(k)-index (He & Yang, 2004) builds a coarser index than A(k)-index but suffers from over-refinement. M(k)-index, an improvement of D(k)-index, and solves the problem of large scan space within the index without affecting path coverage. However, there is a drawback in this design: M(k)-index requires adapting to a given list of FUPs.

U(*)-index (universal, generic index) (Boulos & Karakashian, 2006), like 1-index, exploits bisimilarity. However, U(*)-index exploits a special node-labeling scheme to prune the search space and accelerate XPath evaluations. Furthermore, U(*)-index does not need to be adapted to any particular list of FUP; it has a uniform resolution and is hence more generic.

APEX (Chung, Min & Shim, 2002) is an adaptive index that searches for a trade-off between size and effectiveness. Instead of indexing all paths from the root, APEX only indexes frequently used paths and preserves the structure of source data in a tree. However, since FUPs are stored in the index, path query processing is quite efficient. APEX is also workload-aware (i.e., it can be dynamically updated according to changes in query workload). A data mining method is used to extract FUPs from the workload for incremental update (Agrawal & Srikant, 1995).

The main weakness of these indices is that they can only answer single path expressions directly. To process so-called branching path expressions whose graphical representation contains branches and corresponds to a small tree (or twig), they must perform a costly join operation. To reduce

the number of joins, XJoin-index (Bertino, Catania & Wang, 2004) precomputes some structural semijoin results to support attribute selection, possibly involving several attributes, detection of parent-child relationships, and counting.

Finally, other techniques such as extended inverted lists (Zhang, Naughton, DeWitt, Luo & Lohman, 2001) and Fabric (Cooper, Sample, Franklin, Hjaltason & Shadmon, 2001) are aimed at processing containment queries over XML data stored in relational databases. Containment queries are based on relationships among elements, attributes, and their contents. Extended inverted lists include a text index (T-index) (Milo & Suciu, 1999) similar to traditional indices in information retrieval systems and an element index (E-index) that maps elements into inverted lists.

Fabric indexes several XML documents by encoding paths, from root to leaves. The resulting indicators are then inserted into a Patricia trie (Cooper et al., 2001), which processes them like simple strings. A dictionary stores correspondence between indicators and path label names. To use this index, query labels are also transformed into indicators by exploiting the dictionary.

Numbering Scheme-Based Indices

A numbering scheme encodes each XML element by its positional information in its document's hierarchy. Most numbering schemes reported in the literature are based either on a tree-traversal order or on the textual positions of start and end tags (Srivastava, Al-Khalifa, Jagadish, Koudas, Patel & Wu, 2002). If such a numbering scheme is embedded in the labeled trees of XML documents, a structural relationship (e.g., ancestor-descendant) between a pair of elements can be determined quickly without traversing the whole tree.

To evaluate queries involving structural relationships, structural join indices efficiently support functions such as *findDescendants* and *findAncestors* that are needed in structural joins. For instance, a B+-tree may be built on the joining element's *StartPos* attribute (Chien, Vagena, Zhang, Tsotras & Zaniolo, 2002). XR-tree (XML Region Tree) (Jiang, Lu, Wang & Ooi, 2003) is a dynamic external memory index structure specifically designed for strictly nested XML data. Actually, an XR-tree is a B+-tree with a complex index key entry and extra stab lists associated with its internal nodes.

XB-tree (Bruno, Koudas & Srivastava, 2002) combines structural features of both B+-tree and R-tree. XB-tree first indexes pre-assigned intervals of elements from a tree structure. Next, it organizes the intervals' starting points as a B+-tree. Each internal node maintains a set of regions that completely includes all regions in their child nodes. Regions in XB-tree nodes may overlap partially.

XML structural join-based experiments performed on these indices indicate that they achieve comparable performances for nonrecursive XML data (i.e., XML documents with no node-to-node internal references), while XB-tree outperforms the other indices for highly recursive XML data (Li, Lee, Hsu & Chen, 2004).

INDICES IN XML DATA WAREHOUSES

XML data warehouses form an interesting basis for decision-support applications that exploit so-called complex data (Darmont, Boussaïd, Ralaivao & Aouiche, 2005). Several studies address the issue of designing and building XML data warehouses. They use XML documents to manage or represent warehouse facts and/or dimensions (Hümmer, Bauer & Harde, 2003; Park, Han & Song, 2005; Pokorný, 2002). This approach helps store XML documents natively and query them easily with XML query languages. However, decision-support queries are generally complex. They indeed typically involve several join and aggregation operations. In addition,

many XML-native DBMSs show relatively poor performances when data volume is very large and queries are complex.

Most existing XML indexing techniques are applicable only onto XML data that are targeted by single path expressions. However, in XML data warehouses, queries are complex and include several path expressions. Furthermore, building existing indices on an XML warehouse causes a loss of information in decision-support query resolution. Indeed, clustering (1-index and variants) or merging (Dataguide) identical labels causes the disappearance of fact-to-dimension relationships, which are essential to process analytical queries. We illustrate this issue in the following example.

Let us consider a sample warehouse document, *cube.xml*, composed of *cell* (fact) elements (Figure 1(a)). Each cell is identified by a combination of dimension identifiers and one or more measures. Figure 1(b) features the corresponding 1-index. Since 1-index represents cells linearly (i.e., all labels sharing the same label path are represented by one label only), the dimension combinations that identify facts are lost.

Eventually, most of the approaches we presented in the previous section can only index one XML document at a time, whereas in XML warehouses, data are typically stored in several fact and dimension XML documents, and analytic queries must be performed over these documents. Fabric does handle multiple documents, but it is not adapted to XML data warehouses either, because it does not take into account relationships between XML documents (i.e., fact-to-dimension references, in our case).

To address the issues of multiple path expressions in analytic queries, loss of referential information, and multidocument indexing, we have proposed a new index specifically adapted to XML, multidimensional data warehouses (Mahboubi, Aouiche & Darmont, 2006a). This data structure helps optimize access time to several XML documents by eliminating join costs, while preserving information contained in the initial warehouse.

To implement our indexing strategy, we selected XCube (Hümmer, Bauer & Harde, 2003) as a reference data warehouse model. Since other XML warehouse models from the literature are relatively similar, this is not a binding choice. XCube's advantage is its simple structure for representing facts and dimensions in a star schema: one document for facts (*facts.xml*) and another one for all dimensions (*dimensions.xml*).

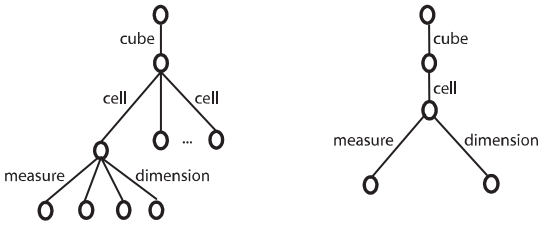
Our index structure is designed to preserve relationships between facts and dimensions. To achieve this goal, we move data from *facts.xml* and *dimensions.xml* into a common structure that actually is our index. This process helps store facts, dimensions, and their attributes into the same XML element. It wholly eliminates join operations since all necessary information for a join operation is stored in the same index cell. This data structure is also stored into an XML document (*index.xml*). Queries need to be rewritten to exploit our index instead of the initial warehouse. This rewriting process consists in preserving selection and aggregation operations while eliminating joins.

To validate our proposal, we performed both a complexity study and field experiments. Our tests showed that using our index structure significantly improved the response time of a typical decision-support query expressed in XQuery. Furthermore, they also demonstrated that well-indexed XML-native DBMSs could compete with relational, XML-compatible DBMSs.

FUTURE TRENDS

As we underlined in the background section, structural summary approaches generate large indices and do not support partial path matching queries. Labeling schemes allow to quickly identify relationships among element nodes and reduce index size, but fail to support dynamic

Figure 1. cube.xml document structure (a) and corresponding 1-index (b)



XML data. Furthermore, the semistructured nature of XML data and requirements on query flexibility poses unique challenges to indexing methods. Hence, quite recently, researchers proposed hybrid indexing techniques (Catania, Maddalena & Vakali, 2006). XML indexing is likely to keep on following this path, while more specific solutions may also appear (e.g., for XML data warehouses).

The XML warehouse index structure we propose also suffers from these common weaknesses (index size and construction cost). We indeed merge all warehouse data into the same structure. In addition, this process needs to parse all elements within the warehouse XML documents. Index construction is thus very costly. FUPs proposed by Min, Chung & Shim (2005) might be a solution. FUPs are obtained from a representative workload with data mining approaches and represent frequent join operations. This could help us materialize these operations only within our index structure.

More generally, XML indexing strategies should be better integrated in host XML-native DBMSs. This would certainly help develop incremental strategies for index maintenance. Moreover, in our particular case, query rewriting mechanisms would also be more efficient if they were part of the system.

Finally, it is crucial to carry on adapting or developing highly efficient optimization techniques in XML-native DBMSs and relational, XML-enabled DBMSs. Several interesting leads are currently being researched, such as XML view materialization (Mahboubi, Aouiche & Darmont, 2006b; Phillips, Zhang, Ilyas & Ozsu, 2006) or partitioning (Bonifati, Cuzzocrea & Zinno, 2006).

CONCLUSION

Neither XML-native nor XML-enabled DBMSs implement most of the indexing techniques presented in this chapter. Both classes of systems indeed support only basic solutions. Relational, XML-enabled DBMSs use simple structural indices such as B-trees and their derivatives. Similarly, most XML-native DBMSs index only element and attribute contents and tag names. In both cases, either full-text inverted indices for indexing textual contents or path indices are typically adopted. Hence, in conclusion, we strongly believe that XML DBMSs should now feature state-of-the-art, XML-specific indexing schemes to be able to compete with relational DBMSs in terms of performance.

REFERENCES

Agrawal, R., & Srikant, R. (1995). *Mining sequential patterns.* Proceedings of the 11th International Conference on Data Engineering (ICDE 95). Taipei, Taiwan, 3–14.

Bertino, E., Catania, B., & Wang, W.Q. (2004). *XJoin index: Indexing XML data for efficient handling of branching path expressions.* Proceedings of the 20th International Conference on Data Engineering (ICDE 04), Boston, Massachusetts, 828.

Beyer, K.S., Chamberlin, D.D., Colby, L.S., Özcan, F., Pirahesh, H., & Xu, Y. (2005). *Extending XQuery for analytics.* Proceedings of the 2005 ACM SIGMOD International Conference on Management of Data (SIGMOD 05), Baltimore, Maryland, 503–514.

Boag, S., Chamberlin, D., Fernandez, M., Florescu, D., Robie, J., & Siméon, J. (2006). XQuery 1.0: An XML query language [W3C working draft]. Retrieved September 20, 2006, from http://www.w3.org/TR/xquery/

Bonifati, A., Cuzzocrea, A., & Zinno, B. (2006). *Fragmenting XML documents via structure constraints.* Proceedings of the 10th East European Conference on Advances in Databases and Information Systems (ADBIS 06), Thessalonica, Greece, 17–29.

Boulos, J., & Karakashian, S. (2006). *A new design for a native XML storage and indexing manager.* Proceedings of the 10th International Conference on Extending Database Technology (EDBT 06), Munich, Germany, 3896, 755–772.

Bray, T., Paoli, J., Sperberg-McQueen, C.M., Maler, E., & Yergeau, F. (2004). Extensible Markup Language (XML) 1.0 (third edition). World Wide Web Consortium.

Bruno, N., Koudas, N., & Srivastava, D. (2002). *Holistic twig joins: Optimal XML pattern matching.* Proceedings of the 2002 ACM SIGMOD International Conference on Management of Data (SIGMOD 02), Madison, Wisconsin, 310–321.

Catania, B., Maddalena, A., & Vakali, A. (2006). XML document indexes: A classification. *IEEE Internet Computing Journal, 9*(5), 64–71.

Chien, S-Y., Vagena, Z., Zhang, D., Tsotras, V.J., & Zaniolo, C. (2002). *Efficient structural joins on indexed XML documents.* Proceedings of the 28th International Conference on Very Large Data Bases (VLDB 02), Hong Kong, China, 263–274.

Chung, C., Min, J., & Shim, K. (2002). APEX: An adaptive path index for XML data. Proceedings of the 2002 ACM SIGMOD International Conference on Management of Data (SIGMOD 02), Madison, Wisconsin, 121–132.

Clark, J., & DeRose, S. (1999). XML path language (XPath), Version 1.0 [W3C working draft]. Retrieved September 20, 2006, from http://www.w3.org/TR/xpath/

Cooper, B.F., Sample, N., Franklin, M.J., Hjaltason, G.R., & Shadmon, M. (2001). *A fast index for semistructured data.* Proceedings of the 27th International Conference on Very Large Data Bases (VLDB 01), Roma, Italy, 341–350.

Darmont, J., Boussaïd, O., Ralaivao, J.C., & Aouiche, K. (2005). *An architecture framework for complex data warehouses.* Proceedings of the 7th International Conference on Enterprise Information Systems (ICEIS 05), Miami, Florida, 370–373.

Goldman, R., & Widom, J. (1997). *DataGuides: Enabling query formulation and optimization in semistructured databases.* Proceedings of the 23rd International Conference on Very Large Data Bases (VLDB 97), Athens, Greece, 436–445.

He, H., & Yang, J. (2004). *Multiresolution indexing of XML for frequent queries.* Proceedings of the 20th International Conference on Data Engineering (ICDE 04), Boston, Massachusetts, 683–694.

Hümmer, W., Bauer, A., & Harde, G. (2003). *XCube: XML for data warehouses.* Proceedings of the 6th International Workshop on Data Warehousing and OLAP (DOLAP 03), New Orleans, Louisiana, 33–40.

Jiang, H., Lu, H., Wang, W., & Ooi, B.C. (2003). *XR-Tree: Indexing XML data for efficient structural joins.* Proceedings of the 19th International Conference on Data Engineering (ICDE 03), Bangalore, India, 253–263.

Kaushik, R., Shenoy, D., Bohannon, P., & Gudes, E. (2002). *Exploiting local similarity to efficiently index paths in graph-structured data.* Proceedings of the 18th International Conference on Data Engineering (ICDE 02), San Jose, California, 129–140.

Li, H., Lee, M.L., Hsu, W., & Chen, C. (2004). An evaluation of XML indexes for structural join. *SIGMOD Record, 33*(3), 28–33.

Mahboubi, H., Aouiche, K., & Darmont, J. (2006a). *Un index de jointure pour les entrepôts de données XML*. Proceedings of the *6èmes Journées Francophones Extraction et Gestion des Connaissances (EGC 06), Lille, France, E-6, 89–94.*

Mahboubi, H., Aouiche, K., & Darmont, J. (2006b). Materialized view selection by query clustering in XML data warehouses. Proceedings of the *4th International Multiconference on Computer Science and Information Technology (CSIT 06), Amman, Jordan,* 68–77.

Meier, W. (2002). *eXist: An open source native XML database*. Proceedings of the Web, Web-Services, and Database Systems, NODe 2002 Web and Database-Related Workshops, Erfurt, Germany, 2593, 169–183.

Milo, T., & Suciu, D. (1999). *Index structures for path expressions.* Proceedings of the 7th International Conference on Database Theory (ICDT 99), Jerusalem, Israel, 1540, 277–295.

Min, J., Chung, C., & Shim, K. (2005). An adaptive path index for XML data using the query workload. *Information Systems, 30*(6), 467–487.

Park, B.K., Han, H., & Song, I.Y. (2005). *XML-OLAP: A multidimensional analysis framework for XML warehouses.* Proceedings of the 7th International Conference on Data Warehousing and Knowledge Discovery, (DaWaK 05), Copenhagen, Denmark, 3589, 32–42.

Phillips, D., Zhang, N., Ilyas, I.F., & Ozsu, M.T. (2006). *InterJoin: Exploiting indexes and materialized views in XPath evaluation*. Proceedings of the 18th International Conference on Scientific and Statistical Database Management (SSDBM 06), Vienna, Austria, 13–22.

Pokorný, J. (2002). *XML data warehouse: Modelling and querying.* Proceedings of the 5th International Baltic Conference (BalticDB&IS 02), Tallinn, Estonia, 267–280.

Qun, C., Lim, A., & Ong, K.W. (2003). *D(k)-index: An adaptive structural summary for graph-structured data*. Proceedings of the 2003 ACM SIGMOD International Conference on Management of Data, San Diego, California, 134-144.

Srivastava, D., Al-Khalifa, S., Jagadish, H.V., Koudas, N., Patel, J.M., & Wu, Y. (2002). *Structural joins: A primitive for efficient XML query pattern matching.* Proceedings of the 18th International Conference on Data Engineering (ICDE 02), San Jose, California, 141.

Wu, H., Wang, Q., Xu Yu, J., Zhou, A., & Zhou, S. (2003). *UD(k and l)-index: An efficient approximate index for XML data*. Proceedings of the 4th International Conference on Advances in Web-Age Information Management (WAIM 03), Chengdu, China, 2762, 68–79.

X-Hive Corporation. (2007). The fastest and most scalable XML database powered by open standards. Retrieved February 26, 2007, from http://www.x-hive.com/products/db/

Zhang, C., Naughton, J.F., DeWitt, D., Luo, Q., & Lohman, G. (2001). *On supporting containment queries in relational database management systems.* Proceedings of the 2001 ACM SIGMOD Conference on Management of Data (SIGMOD 01), Santa Barbara, California, 425–436.

KEY TERMS

Database Management System (DBMS): Software set that handles structuring, storage, maintenance, update, and querying of data stored in a database.

Index: Physical data structure that allows direct (vs. sequential) access to data and thereby considerably improves data access time.

Numbering Scheme-Based Index. Tree structure in which each XML data node is uniquely identified by an interval.

Structural Summary-Based Index: Labeled-graph structure that summarizes XML graph structural information.

XML Data Warehouse: XML database specifically modeled (i.e., multidimensionally with a starlike schema) to support XML decision-support and analytic queries.

XML-Enabled DBMS: Database system in which XML data may be stored and queried from relational tables. Such a DBMS must either map XML data into relations and translate queries into SQL or implement a middleware layer allowing native XML storing and querying.

XML-Native DBMS (NXD): Database system in which XML data are natively stored and queried as XML documents. An NXD provides XML schema storage and implements an XML query engine (typically supporting XPath and XQuery). eXist (Meier, 2002) and X-Hive (X-Hive Corporation, 2007) are examples of NXDs.

Chapter LXXIII
Integrative Information Systems Architecture:
Document & Content Management

Len Asprey
Practical Information Management Solutions Pty Ltd, Australia

Rolf Green
OneView Pty Ltd, Australia

Michael Middleton
Queensland University of Technology, Australia

INTRODUCTION

Purpose

This chapter discusses the benefits of managing business documents and Web content within the context of an integrative information systems architecture. This architecture incorporates database management, document and Web content management, integrated scanning/imaging, workflow and capabilities for integration with other technologies.

Business Context

The ubiquitous use of digital content (such as office documents, email and Web content) for business decision-making makes it imperative that adequate systems are in place to implement management controls over digital content repositories. The traditional approach to managing digital content has been for enterprises to store it in folder structures on file or Web servers. The content files stored within folders are relatively unmanaged, as there are often inadequate classification and indexing structures (taxonomies

and metadata), no adequate version control capabilities and no mechanisms for managing the complex relationships between digital content. These types of relationships include embedded or linked content, content renditions, or control over authored digital documents and published Web content.

In some cases enterprises have achieved a form of management control over hardcopy documents that are records of business transactions by using database applications to register, track and manage the disposal of physical files and documents. These types of file or document "registers" do not provide adequate controls over the capture, retrieval and accessibility to digital content.

This deficiency has led to many organizations seeking solutions, such as document management systems, to manage digital business content. Document management systems have generally been implemented to meet regulatory compliance within the context of document recordkeeping requirements, or management of digital archive collections. Otherwise, they have been implemented as solutions for managing specific types of content objects, such as ISO9001 quality management system documentation, engineering drawings, safety documents, and similar.

More recently, organizations have sought to acquire Web content management systems with the view to providing controls over digital content that is published to Web sites. The imperative for such a solution may be a commercial one, motivated by product to market visibility, customer service, and profitability. There may also be a response to compliance needs, motivated by managing Web content in the context of "recordkeeping" to satisfy regulatory or governance requirements.

The methodology of implementing document or Web content management systems has often been based on a silo approach, with more emphasis on tactical business imperatives than support for strategic enterprise information architecture initiatives. For example, organizations may attempt a Web content management solution without taking into full account digital documents that may be used to create content outside the constraints of Web compatible formats such as XML-defined, but which are subsequently required for publication. Thus, document and Web content management may be viewed as discrete solutions, and business applications may be implemented without an integrative approach using workflow and systems for managing both business documentation and Web content.

Another example of a silo approach is the deployment of database solutions without cognizance of document or Web content management requirements. For example, organizations may deploy a solution for managing contracts, including database application capabilities for establishing the contract, recording payments and variations, and managing contract closure. However, the management of contract documents may not be viewed as an integral part of the application design, or the workflow review and approval, or managing the published contract materials on Web sites. The result is that users often miss vital information rather than manually relate data retrieved through a number of separate applications.

There are compelling reasons for organizations as they address the constructs of enterprise information architecture, to consider the management of digital content within the context of an integrative approach to managing business documents and Web content. The strategic rationale for such an approach encompasses the following types of business imperatives:

- Customer satisfaction is a key commercial driver for both business and government. In the case of the commercial sector, the need to attract and retain customers, and in the public sector, the need to support government initiatives directed at taxpayer benefits. Organizations are adopting strategic

approaches such as single view of customer and one-source solution for customer information, invoking the use of information and knowledge management tools.
- Speed and quality of product to market is another major business driver. The rapid adaptation of the WWW and e-Commerce systems to support online business transactions opens markets to global competition. Commercial enterprises are not only required to deliver product to market rapidly, but also within quality management constraints, to attract and retain customers.
- Regulatory imperatives, such as Sarbanes-Oxley in the U.S. (US Congress, 2002) have introduced new measures for creating greater transparency within organizations, which impact corporate governance, and require disclosure with real-time reporting requirements.

The enterprise information architecture would include information policy, standards and governance for the management of information within an organization and provide supporting tools in the form of an integrative information systems architecture as the platform for managing information. An integrative systems architecture would provide a platform that enables businesses to meet the challenges of both commercial and regulatory imperatives, benefit from reusable information, and provide a coherent view of relevant information enterprise-wide to authorized users.

In respect to document and Web content management, an integrative document and content management (IDCM) model (Asprey and Middleton, 2003) offers a framework for unification of these components into an enterprise information architecture. The model features the management of both documents and Web content within an integrative business and technology framework that manages designated documents and their content throughout the document/content continuum and supports recordkeeping requirements.

Many of the major software vendors, including Oracle, IBM, EMC and Microsoft, have embraced this concept by providing portal, document and content management, collaboration, and workflow capabilities as a combined offering. SharePoint from Microsoft provides these capabilities through integration with their .NET framework allowing for the development of highly customized applications. Vendors such as IBM, Oracle and EMC have provided these capabilities by acquiring and integrating various applications, and these may be marketed as enterprise content management (ECM) suites.

SCOPE

The core IDCM elements that address document and Web content management requirements (capturing content, reviewing/authorizing content, publishing content and archival/disposal) comprise:

- Integrated document and Web publishing/ content management capabilities.
- Integration of document imaging capabilities.
- Recognition technologies, such as barcodes to assist with capturing document information, or conversion of image data to text as a by-product of scanning/imaging.
- Enterprise data management capabilities.
- Workflow.

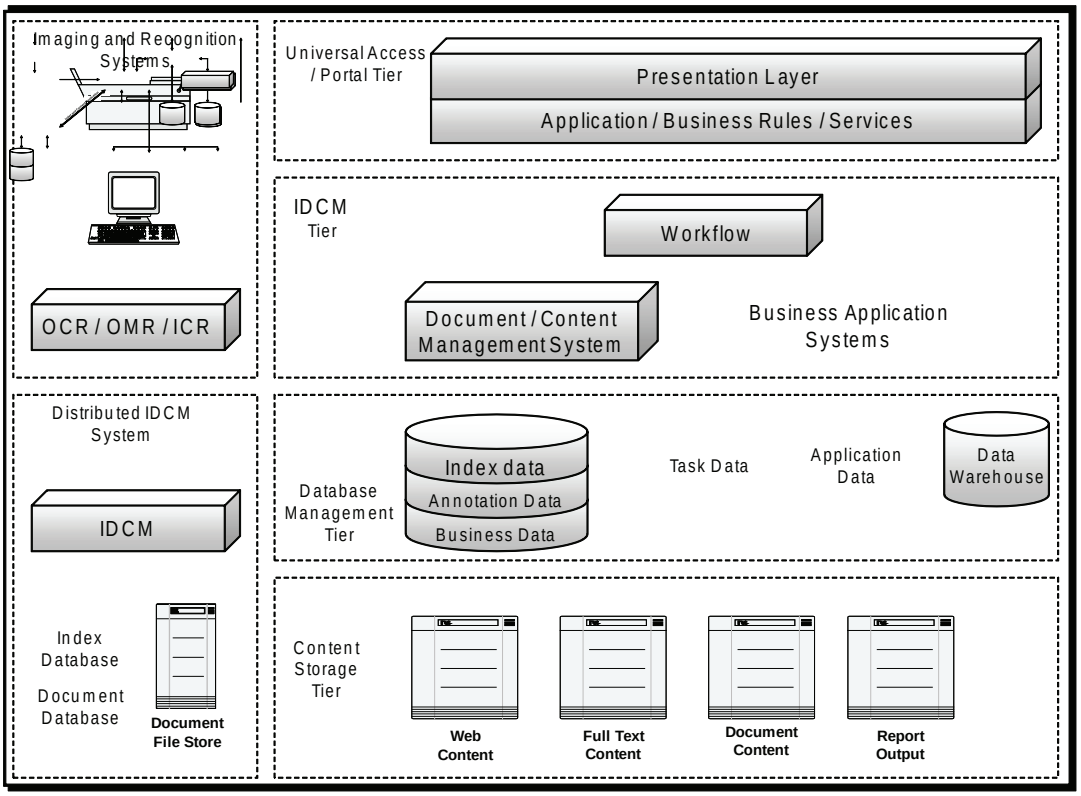
However, when determining requirements within the context of process improvement initiatives that help to address business imperatives (such as customer satisfaction, product to market, and regulatory compliance), these capabilities might be supported by other technologies. This technology support may help businesses to achieve an integrative systems architecture for deployment of innovative and integrated solutions. Typical integration technologies include:

- Universal Access/Portal, which allows users to invoke functions and view information (including digital content) via a Web based interface.
- Data Warehouses, which are centralized repositories for storing enterprise data elements that are derived from business information systems.
- Enterprise Resource Planning (ERP) systems, human resource systems, financial systems, and vertical line of business systems.

These types of capabilities, when combined, augment an integrative systems architecture to support the development of solutions that take advantage of digital content in managed repositories. Users that access business information then have the confidence that they are accessing, retrieving and printing the most current digital content. This confidence can aid decision making, as end users may not then be required to access physical copies of documents, which can be both cumbersome and time-consuming due to customer expectations on speed of service in a modern technology environment.

The following section discusses those features of an integrative information systems architecture that would support a requirement for business users to gain rapid access to the most up to date digital content via an interface that is simple and intuitive.

Figure 1. Information systems architecture – document/content management



SYSTEM FEATURES

Schematic

Figure 1 provides a schematic of database management within the context of IDCM and supporting technologies, such as universal interface (e.g. portal), and interfaces to business applications.

Document Management

Document management applications are used within enterprises to implement management controls for digital and physical document objects. These objects may include office documents, email and attachments, images and multimedia files, engineering drawings and technical specifications. The systems manage complex relationships between documents and provide capabilities to manage integrity, security, authority and audit. Business classification and metadata capabilities support access, presentation and disposal (Bielawski & Boyle, 1997; Wilkinson et al, 1998), thus supporting organizational knowledge management initiatives. Document management applications typically feature recordkeeping tools to support the implementation of organizational recordkeeping policies, file plans and retention policies

Web Content Management

Web content management applications are implemented by organizations to manage digital content that is published to the Web, including Internet, Intranet and Extranet sites. The functionality of Web content management applications can be summarized as content creation, presentation, and management (Arnold, 2003; Robertson, 2003; Boiko, 2002). Organizations are discerning the requirement to have integrated architectures for managing documents and Web content, and to consider the implications of recordkeeping requirements when seeking unified document and content solutions.

Document Imaging

Document imaging systems are used to scan and convert hardcopy documents to either analog (film) or digital format (Muller, 1993). Film based images can be registered and tracked within a document management system, similar to the way in which a paper document can be registered and tracked. Digital images can be captured into either a document or Web content management system via integration with the scanning system.

Digital imaging may involve black and white, grayscale or color imaging techniques, depending on the business need, as the requirement may involve scanning a range of document types, e.g. physical copies of incoming correspondence, forms processing or capture of color brochures.

Some systems allow users to search and view film images on their local computer by the film being scanned on demand. The speed of viewing film images is slow due to the need for the specified film roll needing to be loaded in the scanning device, for the film to be wound to the correct position, and then for the image to be scanned. Digital images may be stored on magnetic storage allowing rapid recall by users and so supporting not only the records archive process but the demands of workflow and document management.

Recognition Systems

Recognition systems such as Barcode recognition, Optical Mark Recognition (OMR), Optical Character Recognition (OCR) or Intelligent Character Recognition (ICR), are often used to facilitate data capture when documents are scanned to digital image.

Systems can take advantage of encoding within barcodes, which can be intelligently encoded to assist with improving business processes. For example, a barcode may contain specific characters that identify a geographical region, product type, or details of a specific subject/topic of interest.

OMR systems are used to capture recognition marks on physical documents, such as bank checks, during digital scanning.

OCR and ICR technologies can allow text to be extracted from a document during digital scanning and conversion processes. The text might be an alphabetical string, or unique number, or words within a zoned area of a form. Depending on the quality of the hardcopy original document, much of the printed information might be convertible to text within acceptable error constraints.

OCR and ICR may also be used to convert existing digital documents, such as Tagged Image File (TIF) Format or Portable Document Format (PDF) documents, to a full text rendition capable of being searched for specific words or phrases and so increasing the ability to recall the primary file being the PDF format.

Workflow

Workflow systems allow more direct support for managing enterprise business processes. These systems are able to automate and implement management controls over tasks from initiation of a process to its closure. Though many application systems can be said to have some level of workflow, they lack the flexibility that enables visual creation and alteration of processes by an enterprise workflow system. More importantly the enterprise workflow allows processes to flow across a number of systems rather than be isolated within a single application.

The drivers for implementing workflow may be to improve both efficiency (by automating and controlling work through a business process), and effectiveness (by monitoring the events work may pass through). Ideally, the design of such systems takes into account a unified analysis of enterprise information content (Rockley, Kostur & Manning, 2003).

The implementation of workflow is often driven from a technology perspective rather then by business process. Questions of flexibility and interoperability become the drivers for the project rather than enhancements to the business processes. The most marked effect of this is the amplification of bottlenecks within the business process when it is run through a workflow system.

Workflow systems provide the integrative framework to support business processes across the enterprise. This is done by providing a capability for the business easily to represent their businesses processes within the workflow system, and by allowing the workflow to call the business functions specific to each system with the required data.

SYSTEM INTEGRATION

Universal Access/Portal

Universal Access/Portal applications are used within enterprises to allow the viewing and interpretation of information from a number of sources, providing a single view of information. These sources of information may include data stored within application databases, data stored in warehouses as well as digital documents and other content stored within document and content management systems. The evolution of Universal Access/Portal applications is to encapsulate not only multi-system data but to provide a single application interface for all enterprise systems.

Corporate portals may be differentiated by application (Collins, 2003) or level (Terra & Gordon,

2003) but they are typically aimed to enhance basic Intranet capability. This may be achieved by provision of facilities such as personalization, metadata construction within enterprise taxonomy, collaborative tools, survey tools, and integration of applications. The integration may extend to external applications for business-business or business-customer transactions, or for derivation of business intelligence by associated external and internal information.

Universal Access/Portal applications include the capability of understanding how the information in each system is related to the business process and the rules by which the data can be combined. They provide different views of information depending on the users needs and allow information to be utilized to greater effect. Just as importantly they provide the controls by which the information capture of the enterprise can be managed to allow the support of both industry and compliance requirements. True on demand reporting of information is provided through dashboards and snapshots rather then the more passive approach of overnight printed reporting.

These applications employ business intelligence to provide the integrative framework to support utilization of information across the enterprise. This framework allows enterprises to expose the applications and business rules as web services to be used by other business systems supporting a service oriented architecture (SOA). Such services may be expanded within larger organizations and outsource companies to offer information management software as a services (SaaS) to smaller companies who may not otherwise be able to implement such an integrative architecture.

Data Warehouse

Many organizations use data-warehouse applications to store archived records from various business systems. In such cases data may be accessible only through a separate user interface, therefore requiring manual steps to reference data to records within the originating and other systems.

Some data warehouse implementations feature Document/Content Management applications rather then purely database applications. Enterprise Report Management (ERM) formerly known as Computer Output to Laser Disk (COLD) software allows the storage of archived data as files and uses the database to store the metadata allowing searching and retrieval of the data file.

The implementation of data repositories within an integrated architecture allows cross referencing and searching of corporate knowledge and information across the enterprise as well as single point of access of information regardless of the repository in which it resides. Integration with Workflow components allows archiving to be done on a transaction basis rather than only batch processing, as well as for archiving of workflow objects as business records.

Enterprise Resource Planning Systems

Given the business impetus towards improving customer service, such as optimization of product to market in what are mostly highly volatile environments, there is a need for management to obtain rapid access to strategic, tactical and operational information. Management information is typically stored in database applications, such as ERP systems, human resource systems, financial systems, and vertical line of business systems. However, information stored in databases often needs to be supplemented by copies of supporting documentation.

Integration between document/content repositories and operational/ administrative systems enables end users to access all information that is relevant to a particular matter. For example,

copies of a contract may be required to support a contract variation, or a supporting invoice may be required to support payment processing.

ENTERPRISE DATA MANAGEMENT

Each of the components within an IDCM application is highly dependent on database technology for the storage of data, including the metadata relating to the various information objects. More recently, applications like Microsoft Office SharePoint Services (MOSS) 2007 are requiring the database to store not only the metadata but also the actual content replacing file servers within the content tier (Figure 1) with multiple database servers.

Software for managing the repositories must accommodate versioning, routing, security and review and approval processes. It should also link to tailored interfaces, be able to deal with different renditions of documents, to maintain associations within complex documents, and to provide for complex queries. Information retrieval development in this area focuses on facilitating indexing and integrating taxonomies (Becker, 2003).

The greater use of Portal functions including Collaboration tools and Knowledge tools, including Wiki stores, has increased both the amount of information being stored and the number of database calls being made. Traditionally these applications used documents to store much of the data but newer implementations have opted to store all information within the database for reasons including simpler searching.

The improvement of information usage increases the number of data calls to the supporting databases and the need for data to be accessible across a wider geographic area. Data supporting document/content management, workflow and other business systems may be required to be managed across a distributed architecture of replicated and cached data stores.

REASONS FOR UTILIZATION

The implementation of systems for managing digital and Web content in silo systems may only solve a tactical business requirement. Essentially, controls over content are implemented. However, silo approaches to the implementation of systems, while they perhaps solve tactical requirements, might not be the most strategic and innovative approach to help solving the key imperatives facing enterprises. Solutions that embrace integrated document and Web content management, combined with universal interface/portal applications, and integration with business systems, may support better opportunities for business process improvement.

With respect to knowledge management, document and Web content management solutions support enterprise knowledge management initiatives, where the knowledge has been made explicit. The strategic approach to managing both data in databases and digital content using an integrative systems architecture, may provide a more cohesive and coherent management of information, with the potential to add more value to knowledge management initiatives (Laugero & Globe, 2002). The use of the universal interface or portal will help to address usability issues with document and content management applications. While these systems contain a wide range of functionality, the end user may not need to invoke much of the functionality, and an enterprise portal style interface approach allows better presentation and management of end user functions and viewing capabilities.

Organizations may find that it is difficult to cost justify digital document management or Web content management applications based purely on information management notions of "better managing enterprise document collections". The investment in the technology may outweigh the perceived benefits. However, the approach of solving strategic management challenges using an

integrative systems architecture to deliver end-to-end business applications, may make it easier to support business justification. At the same time, the enterprise is able to secure a managed repository for documents and Web content to support information policy and planning.

CRITICAL ISSUES

Table 1 summarizes some of the critical issues that enterprises may need to consider when reviewing requirements for an information systems architecture that features integrative document and content management capabilities.

CONCLUSION

The acquisition of enterprise document and Web content management systems might not be justified based only on notions of good information management or contribution to knowledge management. The likelihood of a document and content management project targeted at the enterprise being successful may be enhanced significantly by incorporating the capabilities of these systems into an integrative information systems architecture. This platform may then allow business to initiate projects that deliver a wide range of business process improvement initiatives, all relying on a supporting and consistent enterprise platform, and which provides easy access to authorized users of information. Thus, the platform becomes strategically positioned to support business imperatives in the strategic, tactical and operational hierarchy of an organization, and allow the deployment of innovative and stable end-to-end business solutions.

Table 1. A Summary of critical issues

Business Issues	Technology Issues
Cost The cost of an information systems architecture is likely to be higher than a tactical or silo solution.	**Infrastructure** The infrastructure within the organization may not be adequate for the deployment of the information systems architecture.
Planning The extent of planning required for the acquisition and implementation of a strategic enterprise platform is likely be longer than that required for a tactical solution.	**Systems Integration** The integration between the components of the solutions may be jeopardized if technical specifications are not correctly and thoroughly defined, or if package selection process is not well-managed.
Specifications The extent of specifications required for a strategic enterprise platform is likely to be more extensive than that required for a tactical solution.	**Evolving Technology** The evolving nature of technology, including technology convergence, may impact the long term rollout of the information systems architecture.
Benefits Realization Benefits of a strategic solution may not be realized as early as those for a tactical solution. However, the benefits may be more enduring.	**Security** The nature of the architecture, including integrated components, is likely to involve the development of detailed specifications to ensure seamless access to authorized information.
Lack of Business Unit Buy-In Autonomous business units may not see the benefit of a strategic enterprise solution, focusing instead on specific tactical and operational requirements.	**Disaster Recovery** Storage of large amounts of data made available on demand introduces the need for complex backup procedures and longer recovery times.

REFERENCES

Addey, D., Ellis, J., Suh, P., & Thiemecke, D. (2002). *Content management systems.* Birmingham, UK: glasshaus.

Arnold, S. E. (2003). Content management's new realities. *Online, 27*(1), 36-40.

Asprey, L., & Middleton, M. (2003). *Integrative document and content management: strategies for exploiting enterprise knowledge.* Hershey, PA: Idea Group.

Becker, S. A. (2003). *Effective databases for text & document management*. Hershey PA: IRM Press.

Bielawski, L., & Boyle, J. (1997). *Electronic document management systems: a user centered approach for creating, distributing and managing online publications.* Upper Saddle River, NJ: Prentice-Hall.

Boiko, B. (2002). *Content management bible*. New York: Hungry Minds.

Collins, H. (2003). *Enterprise knowledge portals.* NY: American Management Association.

Laugero, G., & Globe, A. (2002). *Enterprise content services: a practical approach to connecting content management to business strategy.* Boston, MA: Addison-Wesley.

Microsoft (2007) SharePoint Server home page retrieved 14th March, 2008, from http://office.microsoft.com/en-us/sharepointserver/FX100492001033.aspx

Muller, N. J. (1993). *Computerized document imaging systems: technology and applications.* Boston, MA: Artech House.

Robertson, J. (2003). *So, what is a content management system?* Retrieved 14th March, 2008, from http://www.steptwo.com.au/papers/kmc_what/index.html

Rockley, A., Kostur, P., & Manning, S. (2003). *Managing enterprise content: a unified content strategy.* Indianapolis, IN: New Riders.

Terra, J., & Gordon, C. (2003). *Realizing the promise of corporate portals.* Boston, MA: Butterworth-Heinemann.

U.S. Congress (2002, July 30) Public Law 107-204: Sarbanes-Oxley Act of 2002 Retrieved July 16th 2004 from http://frwebgate.access.gpo.gov/cgi-bin/getdoc.cgi?dbname=107_cong_public_laws&docid=f:publ204.107.pdf

Wilkinson, R. Arnold-Moore, T., Fuller, M., Sacks-Davis, R., Thom J., & Zobel, J. (1998). *Document computing: technologies for managing electronic document collections.* Boston, MA: Kluwer Academic Publishers.

KEY TERMS

Document Capture: Registration of an object into a document, image or content repository.

Document Imaging: Scanning and conversion of hardcopy documents to either digital image format or analogue (film).

Document Management: Implements management controls over digital documents via integration with standard desktop authoring tools (word processing, spreadsheets and other tools) and document library functionality. Registers and tracks physical documents.

IDCM: Integrative document and content management.

Portal: User interface to a number of process and information sources.

Recognition Technologies: Technologies such as barcode recognition, optical character recognition (OCR), intelligent character recognition

(ICR), and optical mark recognition (OMR) that facilitate document registration and retrieval.

Web Content Management: Implementation of a managed repository for digital assets such as documents, fragments of documents, images and multimedia that are published to intranet and Internet WWW sites.

Workflow Software: Tools that deal with the automation of business processes in a managed environment.

Chapter LXXIV
Index and Materialized View Selection in Data Warehouses

Kamel Aouiche
Université de Québec à Montréal, Canada

Jérôme Darmont
University of Lyon (ERIC Lyon 2), France

INTRODUCTION

Database management systems (DBMSs) require an administrator whose principal tasks are data management, both at the logical and physical levels, as well as performance optimization. With the wide development of databases and data warehouses, minimizing the administration function is crucial. This function includes the selection of suitable physical structures to improve system performance.

View materialization and indexing are presumably some of the most effective optimization techniques adopted in relational implementations of data warehouses. Materialized views are physical structures that improve data access time by precomputing intermediary results. Therefore, end-user queries can be efficiently processed through data stored in views and do not need to access the original data. Indexes are also physical structures that allow direct data access. They avoid sequential scans and thereby reduce query response time. Nevertheless, these solutions require additional storage space and entail maintenance overhead. The issue is then to select an appropriate configuration of materialized views and indexes that minimizes both query response time and maintenance cost given a limited storage space. This problem is NP hard (Gupta & Mumick, 2005).

The aim of this article is to present an overview of the major families of state-of-the-art index and materialized view selection methods, and to discuss the issues and future trends in data warehouse performance optimization. We particularly focus on data-mining-based heuristics we developed to reduce the selection problem complexity and target the most pertinent candidate indexes and materialized views.

BACKGROUND

Today's commercial relational DBMSs provide integrated tools for automatic physical design. For a given workload, they automatically recom-

mend configurations of indexes and materialized views (Dageville, Das, Dias, Yagoub, Zaït, & Ziauddin, 2004) coupled with data partitioning (Agrawal, Chaudhuri, Kollàr, Marathe, Narasayya, & Syamala, 2004) or table clustering (Zilio et al., 2004). However, these tools depend on the query optimizer and therefore the host DBMS, which renders their adaptation onto other systems intricate. In the remainder of this section, we detail published research about index and materialized view selection.

Index Selection Problem

The index selection problem has been studied for many years in databases (Chaudhuri, Datar, & Narasayya, 2004; Finkelstein, Schkolnick, & Tiberio, 1988), but adaptations to data warehouses are few. In this particular context, research studies may be clustered into two families: algorithms that optimize maintenance cost (Labio, Quass, & Adelberg, 1997) and algorithms that optimize query response time. In both cases, optimization is realized under the storage space constraint. We particularly focus on the second family of approaches, which may be classified depending on how the set of candidate indexes and the final configuration of indexes are built.

The set of candidate indexes may be built manually by the administrator according to his or her expertise of the workload (Choenni, Blanken, & Chang, 1993a, 1993b; Frank, Omiecinski, & Navathe, 1992). This is both subjective and quite hard to achieve when the number of queries is large. In opposition, candidate indexes may also be extracted automatically through a syntactic analysis of the workload (Chaudhuri & Narasayya, 1997; Golfarelli, Rizzi, & Saltarelli, 2002; Valentin, Zuliani, Zilio, Lohman, & Skelley, 2000).

There are several methods for building the final index configuration from the candidate indexes. Ascending methods start from an empty set of indexes (Chaudhuri & Narasayya, 1997; Choenni et al., 1993b; Frank et al., 1992; Kyu-Young, 1987). They increasingly select indexes minimizing workload cost until it does not decrease anymore. Descending methods start with the whole set of candidate indexes and prune indexes until workload cost increases (Choenni et al., 1993a; Kyu-Young, 1987). Classical optimization algorithms have also been used to solve this problem, such as knapsack resolution (Feldman & Reouven, 2003; Gündem, 1999; Ip, Saxton, & Raghavan, 1983; Valentin et al., 2000) and genetic algorithms (Kratika, Ljubic, & Tosic, 2003).

Materialized View Selection Problem

The classical papers about materialized view selection in data warehouses introduce a lattice framework that models and captures ancestor-descendent dependency among aggregate views in a multidimensional context (Baralis, Paraboschi, & Teniente, 1997; Harinarayan, Rajaraman, & Ullman, 1996; Kotidis & Roussopoulos, 1999; Uchiyama, Runapongsa, & Teorey, 1999). This lattice is greedily browsed with the help of cost models to select the best views to materialize. This problem has first been addressed in one data cube, and then extended to multiple cubes (Shukla, Deshpande, & Naughton, 2000). Another theoretical framework, called the and-or view graph, may also be used to capture the relationships between materialized views (Chan, Li, & Feng, 1999; Gupta & Mumick, 2005; Theodoratos & Bouzeghoub, 2000; Valluri, Vadapalli, & Karlapalem, 2002). However, the majority of these solutions are theoretical and not truly scalable.

Another method decomposes data cubes into an indexed hierarchy of wavelet view elements and selects those that minimize the average processing cost of the queries defined on the data cubes (Smith, Li, & Jhingran, 2004). Similarly, the dwarf structure (Sismanis, Deligiannakis, Roussopoulos, & Kotidis, 2002) compresses data cubes, thereby suppressing redundancy to improve maintenance and interrogation costs. These approaches are very interesting, but they mainly focus on computing efficient data cubes by changing their physical design, which is not always convenient in practice.

Yet other approaches detect common subexpressions within workload queries that correspond to intermediary results that are suitable to materialize (Baril & Bellahsene, 2003; Goldstein & Larson, 2001). However, browsing is very costly and these methods are not truly scalable with respect to the number of queries.

Finally, the most recent approaches are workload driven. They syntactically analyze the workload to enumerate relevant candidate views (Agrawal, Chaudhuri, & Narasayya, 2001). By calling the system query optimizer, they greedily build a configuration of the most pertinent views. A real workload is indeed considered as a good starting point to predict future queries.

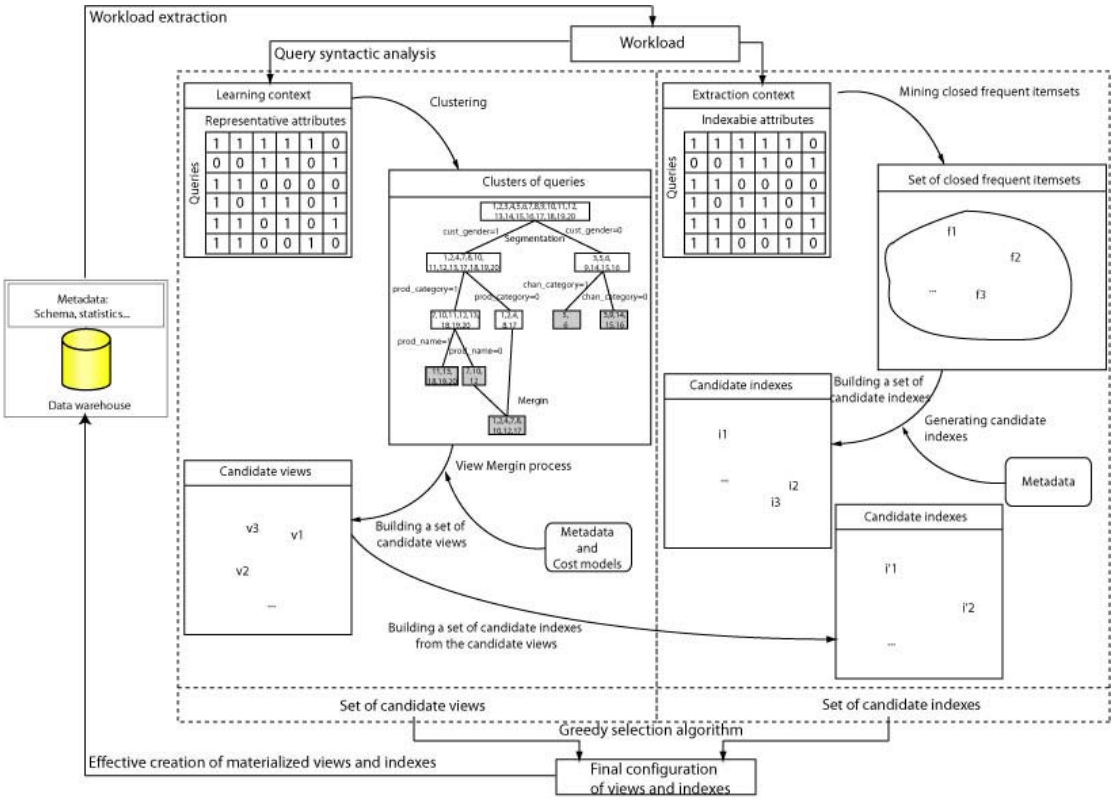
Coupling Index and Materialized View Selection

A few research studies deal with the simultaneous selection of indexes and materialized views. Agrawal et al. (2001) proposed three alternative approaches. The first, MVFIRST, selects materialized views first and then indexes. The second, INDFIRST, selects indexes first and then materialized views. The third, joint enumeration, is claimed by the authors to be the most efficient for workload execution time optimization. It processes indexes, materialized views, and indexes over these views simultaneously.

Bellatreche, Karlapalem, and Schneider (2000) studied the problem of storage space distribution among materialized views and indexes. A set of views and indexes are selected as an initial solution. Then, this solution is iteratively modified to reduce the execution cost by redistributing storage space among indexes and materialized views.

Finally, Rizzi and Saltarelli (2003) a priori determine a trade-off between the storage spaces allotted to indexes and materialized views depending on how queries are defined. Their idea is that view materialization provides the best

Figure 1. Materialized view and index selection system



benefit for queries involving coarse granularity aggregations, while indexing provides the best benefit with queries containing attributes with a high selectivity.

DATA-MINING-BASED INDEX AND MATERIALIZED VIEW SELECTION

Strategy Overview

We advocate for index and materialized view selection strategies that bear the following features:

- **Automatic:** The final configuration of indexes and views should be built automatically.
- **Generic:** The selection strategy should not be dependent on a particular DBMS.
- **Modular:** The selection strategy should be composed of independent modules.
- **Scalable:** The strategy must be able to handle large workloads.

To achieve this goal, we designed a new strategy (Aouiche, Darmont, Boussaïd, & Bentayeb, 2005; Aouiche, Jouve, & Darmont, 2006) that is composed of several modules: a syntactical query analyzer, a data miner, cost models, and an index and materialized view selector (Figure 1). The query analyzer syntactically processes the input workload to extract the most pertinent attributes for indexing and view materialization. It also exploits some knowledge about performance administrative tasks, formalized as if-then rules. For instance, common rules imply the selection of attributes from the "Where" and "Group by" clauses in SQL statements. The output of this process is a so-called query-attribute binary matrix whose lines are the analyzed queries and whose columns are the extracted attributes. The general term of this matrix is set to 1 if an extracted attribute is present in the corresponding query and to 0 otherwise.

The query-attribute matrix constitutes the extraction context for the data miner. This module builds a configuration of candidate indexes and materialized views. It may exploit any data mining technique suiting the data structure to select (indexes or materialized views). For instance, our materialized view selection strategy exploits clustering for building sets of similar queries. The idea of exploiting clustering is motivated by the fact that several queries having a similar syntax may likely be resolved from one materialized view. Hence, workload queries are grouped into clusters that are exploited to build the set of candidate views. Furthermore, these candidate views are merged to resolve multiple queries.

Our index selection strategy exploits another data mining technique, frequent itemset mining, to determine the candidate indexes. Our intuition here is that index utility is strongly correlated to the usage frequency of the corresponding attributes within a given workload, which frequent itemset mining is good at highlighting. Each itemset is analyzed to generate a set of candidate indexes with the help of metadata (primary keys, foreign keys, statistics, etc.). This process for building candidate indexes may also be applied on the candidate materialized views since they are actually tables. Hence, we can build indexes on materialized views to maximize performance improvements.

Finally, the cost model module takes as input the data warehouse metadata, the workload and candidate indexes, and materialized views. It computes the cost, in terms of access and storage cost, of each query in the presence of the candidate indexes and/or views. Since the number of candidates is generally as high as the input workload is large, it is not feasible to materialize them all because of storage space constraints. Hence, our cost models are exploited by the index and view selector to greedily build a final configuration of indexes and materialized views. When simultaneously selecting indexes and materialized views, we exploit specific cost models that allow taking into account the interactions between indexes and materialized views and efficiently sharing storage space.

Discussion

Thanks to its modularity, we have been able to apply our strategy in several cases. For instance, we performed B-tree index selection in a database context as well as bitmap join index selection in a data warehouse. In addition, modularity helps in gradually improving our strategy. A given module may indeed be easily replaced by another more efficient one. For example, it is easy to replace our cost models with more accurate ones, or a data mining algorithm with a more efficient or scalable one.

In opposition to other approaches, particularly those of DBMS vendors, we aimed at remaining as generic and independent from the host DBMS as possible. Our analyzer module indeed processes standard SQL queries, for instance. Our cost models are also mathematical so that they do not depend on a query optimizer. Hence, our strategy may be instantiated within different systems.

Finally, our approach takes into account knowledge (metadata, usage statistics, knowledge extracted from the query workload, the way attributes are queried, etc.) that helps in reducing the selection problem complexity and thus targeting the most pertinent candidate indexes and materialized views. Since this approach is largely based on data mining, we benefit from the active research in this field, which now provides fast and scalable algorithms. Hence, we can process and analyze large workloads.

FUTURE TRENDS

Our strategies are applied on a workload that is extracted from the system during a given period of time. We are thus performing static optimization. Future developments in this domain (both ours and others) should be dynamic and incremental (Kotidis & Roussopoulos, 1999). In our case, studies dealing with dynamic or incremental clustering and frequent itemset mining may be exploited to update the configuration of indexes and materialized views instead of recreating it from scratch. Entropy-based session detection could also be beneficial to determine the best moment to periodically run such a strategy.

In our work, we also only coupled the selection of indexes and materialized views, but the current trend in recent commercial systems is to exploit a mix of several optimization techniques such as buffering, physical clustering, partitioning, and so forth (Agrawal et al., 2004; Dageville et al., 2004; Zilio et al., 2004) to achieve the best performance enhancement. We also aim at integrating the selection of other optimization structures into our strategy.

A tremendous amount of research is currently in progress to help XML- (extensible markup language) native DBMSs in becoming a credible alternative to XML-compatible, relational DBMSs. The majority of XML-native DBMSs indeed present relatively poor performance when the volume of data is very large and queries are complex. However, since XML is gaining importance for representing business data for analytics (Beyer, Chamberlin, Colby, Özcan, Pirahesh, & Xu, 2005), it is crucial to design automatic ways of guaranteeing the best performance of XML data warehouses.

CONCLUSION

The problem of performance optimization has been receiving significant attention since the early days of database research. However, each new class of DBMS (hierarchical, network, relational, object, XML, etc.) or special purpose of database architecture (such as the decision-support data warehouses) invariably gives way to the reformulation or adaptation of existing techniques, and to brand new issues that require original solutions.

REFERENCES

Agrawal, S., Chaudhuri, S., Kollàr, L., Marathe, A. P., Narasayya, V. R., & Syamala, M. (2004). Database tuning advisor for Microsoft SQL Server

2005. *30th International Conference on Very Large Data Bases (VLDB 2004)* (pp. 1110-1121).

Agrawal, S., Chaudhuri, S., & Narasayya, V. (2001). Materialized view and index selection tool for Microsoft SQL Server 2000. *ACM SIGMOD International Conference on Management of Data (SIGMOD 2001)* (p. 608).

Aouiche, K., Darmont, J., Boussaïd, O., & Bentayeb, F. (2005). Automatic selection of bitmap join indexes in data warehouses. In *Lecture notes in computer science: Vol. 3589. 7th International Conference on Data Warehousing and Knowledge Discovery (DaWaK 2005)* (pp. 64-73). Berlin, Germany: Springer.

Aouiche, K., Jouve, P. E., & Darmont, J. (2006). Clustering-based materialized view selection in data warehouses. In *Lecture notes in computer science: Vol. 4152. 10th East-European Conference on Advances in Databases and Information Systems (ADBIS 2006)* (pp. 81-95). Berlin, Germany: Springer.

Baralis, E., Paraboschi, S., & Teniente, E. (1997). Materialized views selection in a multidimensional database. *23rd International Conference on Very Large Data Bases (VLDB 1997)* (pp. 156-165).

Baril, X., & Bellahsene, Z. (2003). Selection of materialized views: A cost-based approach. In *Lecture notes in computer science: Vol. 2681. 15th International Conference on Advanced Information Systems Engineering (CAiSE 2003)* (pp. 665-680). Berlin, Germany: Springer.

Bellatreche, L., Karlapalem, K., & Schneider, M. (2000). On efficient storage space distribution among materialized views and indices in data warehousing environments. *9th International Conference on Information and Knowledge Management (CIKM 2000)* (pp. 397-404).

Beyer, K. S., Chamberlin, D. D., Colby, L. S., Özcan, F., Pirahesh, H., & Xu, Y. (2005). Extending XQuery for analytics. *ACM SIGMOD International Conference on Management of Data (SIGMOD 2005)* (pp. 503-514).

Chan, G. K. Y., Li Q., & Feng, L. (1999). Design and selection of materialized views in a data warehousing environment: A case study. *2nd ACM international workshop on Data warehousing and OLAP (DOLAP 1999)* (pp. 42-47).

Chaudhuri, S., Datar, M., & Narasayya, V. (2004). Index selection for databases: A hardness study and a principled heuristic solution. *IEEE Transactions on Knowledge and Data Engineering, 16*(11), 1313-1323.

Chaudhuri, S., & Narasayya, V. R. (1997). An efficient cost-driven index selection tool for Microsoft SQL server. *23rd International Conference on Very Large Data Bases (VLDB 1994)* (pp. 146-155).

Choenni, S., Blanken, H. M., & Chang, T. (1993a). Index selection in relational databases. *5th International Conference on Computing and Information (ICCI 1993)* (pp. 491-496).

Choenni, S., Blanken, H. M., & Chang, T. (1993b). On the selection of secondary indices in relational databases. *Data Knowledge Engineering, 11*(3), 207-238.

Dageville, B., Das, D., Dias, K., Yagoub, K., Zaït, M., & Ziauddin, M. (2004). Automatic SQL tuning in Oracle 10g. *30th International Conference on Very Large Data Bases (VLDB 2004)* (pp. 1098-1109).

Feldman, Y. A., & Reouven, J. (2003). A knowledge-based approach for index selection in relational databases. *Expert System with Applications, 25*(1), 15-37.

Finkelstein, S. J., Schkolnick, M., & Tiberio, P. (1988). Physical database design for relational databases. *ACM Transactions on Database Systems, 13*(1), 91-128.

Frank, M. R., Omiecinski, E., & Navathe, S. B. (1992). Adaptive and automated index selection in RDBMS. In *Lecture notes in computer science: Vol. 580. 3rd International Conference on Extending Database Technology, (EDBT 1992)* (pp. 277-292). Berlin, Germany: Springer.

Goldstein, J., & Larson, P. A. (2001). Optimizing queries using materialized views: A practical, scalable solution. *ACM SIGMOD International Conference on Management of Data (SIGMOD 2001)* (pp. 331-342).

Golfarelli, M., Rizzi, S., & Saltarelli, E. (2002). Index selection for data warehousing. *CEUR Workshop Proceedings: Vol. 58. 4th International Workshop on Design and Management of Data Warehouses (DMDW 2002)* (pp. 33-42).

Gündem, T. I. (1999). Near optimal multiple choice index selection for relational databases. *Computers & Mathematics with Applications, 37*(2), 111-120.

Gupta, H., & Mumick, I. S. (2005). Selection of views to materialize in a data warehouse. *IEEE Transactions on Knowledge and Data Engineering, 17*(1), 24-43.

Harinarayan, V., Rajaraman, A., & Ullman, J. D. (1996). Implementing data cubes efficiently. *ACM SIGMOD International Conference on Management of Data (SIGMOD 1996)* (pp. 205-216).

Ip, M. Y. L., Saxton, L. V., & Raghavan, V. V. (1983). On the selection of an optimal set of indexes. *IEEE Transactions on Software Engineering, 9*(2), 135-143.

Kotidis, Y., & Roussopoulos, N. (1999). Dynamat: A dynamic view management system for data warehouses. *ACM SIGMOD International Conference on Management of Data (SIGMOD 1999)* (pp. 371-382).

Kratika, J., Ljubic, I., & Tosic, D. (2003). A genetic algorithm for the index selection problem. In *Lecture notes in computer science: Vol. 2611. Applications of Evolutionary Computing (EvoWorkshops 2003)* (pp. 281-291). Berlin, Germany: Springer.

Kyu-Young, W. (1987). Index selection in rational databases. In S. P. Ghosh, Y. Kambayashi, & K. Tanaka (Eds.), *Foundation of data organization* (pp. 497-500). New York: Plenum Publishing.

Labio, W., Quass, D., & Adelberg, B. (1997). Physical database design for data warehouses. *13th International Conference on Data Engineering (ICDE 1997)* (pp. 277-288).

Rizzi, S., & Saltarelli, E. (2003). View materialization vs. indexing: Balancing space constraints in data warehouse design. In *Lecture notes in computer science: Vol. 2681. 15th International Conference on Advanced Information Systems Engineering (CAiSE 2003)* (pp. 502-519). Berlin, Germany: Springer.

Shukla, A., Deshpande, P., & Naughton, J. F. (2000). Materialized view selection for multicube data models. In *Lecture notes in computer science: Vol. 1777. 7th International Conference on Extending Database Technology (EDBT 2000)* (pp. 269-284). Berlin, Germany: Springer.

Sismanis, Y., Deligiannakis, A., Roussopoulos, N., & Kotidis, Y. (2002). Dwarf: Shrinking the petacube. *ACM SIGMOD International Conference on Management of Data (SIGMOD 2002)* (pp. 464-475).

Smith, J. R., Li, C. S., & Jhingran, A. (2004). A wavelet framework for adapting data cube views for OLAP. *IEEE Transactions on Knowledge and Data Engineering, 16*(5), 552-565.

Theodoratos, D., & Bouzeghoub, M. (2000). A general framework for the view selection problem for data warehouse design and evolution. *3rd ACM International Workshop on Data Warehousing and OLAP (DOLAP 2000)* (pp. 1-8).

Uchiyama, H., Runapongsa, K., & Teorey, T. J. (1999). A progressive view materialization algorithm. *2nd International Workshop on Data Warehousing and OLAP (DOLAP 1999)* (pp. 36-41).

Valentin, G., Zuliani, M., Zilio, D., Lohman, G., & Skelley, A. (2000). DB2 advisor: An optimizer smart enough to recommend its own indexes. *16th International Conference on Data Engineering, (ICDE 2000)* (pp. 101-110).

Valluri, S. R., Vadapalli, S., & Karlapalem, K. (2002). View relevance driven materialized view selection in data warehousing environment. *13th Australasian Database Conference (ADC 2002)* (pp. 187-196).

Zilio, D. C., Rao, J., Lightstone, S., Lohman, G. M., Storm, A., Garcia-Arellano, C., et al. (2004). DB2 design advisor: Integrated automatic physical database design. *30th International Conference on Very Large Data Bases (VLDB 2004)* (pp. 1087-1097).

KEY TERMS

Data Cube: Data modeled and viewed in a multidimensional space.

Data Mining: The nontrivial extraction of implicit, previously unknown, and potentially useful information from data.

Granularity: The aggregation level within a dimension hierarchy.

Index: Physical data structure that allows direct (vs. sequential) access to data.

Materialized View: Physical data structure that improves data access time by precomputing intermediary results.

Online Analytical Processing (OLAP): An approach for processing decision-support, analytical queries that are dimensional in nature.

Selectivity: The portion of accessed tuples that are effectively selected by a query.

Workload: Set of queries that are executed over a given database or data warehouse.

Chapter LXXV
Synopsis Data Structures for Representing, Querying, and Mining Data Streams

Alfredo Cuzzocrea
University of Calabria, Italy

INTRODUCTION

Data-stream query processing and mining is an emerging challenge for the database research community. This issue has recently gained the attention from the academic as well as the industrial world. Data streams are continuously flooding data produced by untraditional information sources. They are generated by a growing number of data entities, among all we recall performance measurements in network monitoring and traffic management systems, call details records in telecommunication systems, transactions in retail chains, ATM operations, log records generated by Web servers, sensor network data, RFID- (radio frequency identification) based readings, and so forth.

The most distinctive characteristics of data streams are (a) massive volumes of data (e.g., terabytes) and (b) tuples arriving rapidly; the latter make DBMS- (database management system) inspired computational models, which are typically memory bounded, inappropriate for efficiently processing such kinds of data. More specifically, as regards the model used to represent and perform computation over data streams, we can identify the following widely accepted properties characterizing data streams (Babcock, Babu, Datar, Motawani, & Widom, 2002). First, data items of the stream arrive online; typically a time stamp is associated with each, and, as a consequence, the reading of a stream can be modeled as a tuple of kind $\langle id, val, ts \rangle$ such that *id* is the identifier of the reading (e.g., the RFID tag), *val* is the value of the reading (e.g., the bar code of the product identified by the RFID tag), and *ts* is the time stamp of the data item (e.g., the time instant in which the value *val* was produced). Second, the stream processor has no control over the order in which data items arrive (and, thus, over the order in which data items can be processed). Next, the data stream is potentially unbounded. Finally, once an item of the data stream has been processed, it must be discarded so there is no possibility of it being reprocessing more times. For what regards queries, there are two distinct classes of queries that are of interest for extracting useful informa-

tion and knowledge from data streams (Babcock et al., 2002): (a) one-time queries, which are evaluated once over a point-in-time snapshot of the stream (e.g., an aggregate operator, such as SUM and COUNT, computed over the collection of items belonging to the target data stream and having a time stamp contained within a given time interval $[t_i, t_j]$, with $t_i < t_j$ and t_j smaller than the current time t), and (b) continuous queries, which produce a stream of answers over time, reflecting the evolution of the target data stream; in other words, a continuous query Q with frequency $\frac{1}{T_Q}$ produces, at each time t, an answer $A_t(Q)$ computed by considering the items of the stream whose time stamp is contained within the interval $[t\text{-}T_Q, t]$, with T_Q being an input parameter of Q. It should be noted that, while one-time queries are meaningful evolutions of conventional DBMS queries targeted at the particular application context (i.e., data-stream query processing), continuous queries represent a novel and very interesting class of queries that allow us to think of new scenarios of continuous, useful information and knowledge delivery beyond traditional client-server computational schemes, and also pose important challenges for the database and data warehouse research community. In order to efficiently support continuous query answering, new models and algorithms able to fit the novel requirements dictated by the computational model underlying such queries are needed.

Contrary to the above-described characteristics, data-stream-based applications and systems require processing queries and mining patterns over continuously evolving data in real time, thus involving the need for innovative models and algorithms for representing, querying, and mining data streams. As a consequence, there is a stringent need for designing and developing the so-called data-stream management system (DSMS), that is, a system that efficiently supports representation, indexing, query, and mining functionalities over data streams by overcoming the limitations of traditional DBMS.

Similar to querying data streams, mining data streams poses new challenges beyond the actual capabilities of data mining tools, which were designed and developed for mining massive, persistent amounts of data. Typical data mining tasks include association mining (e.g., discovering association rules), classification, and clustering; these techniques aim at finding interesting patterns, regularities, and anomalies in data. Data mining is a traditional discipline that has attracted the attention of a plethora of researchers so that in literature there exists an impressive number of data mining techniques and algorithms, also applied to breaking real-life scenarios such as fraud detection, disaster prevention, and so forth. Unfortunately, all these techniques cannot be exploited for data-stream mining issues directly; this is because they address disk-resident data sets and usually make several passes over data. Contrary to these assumptions, as said previously, streams cannot be stored persistently, and only single-pass algorithms can be applied over them. On the other hand, mining data streams in order to extract useful information and knowledge from them is an exciting line of research that will take over the scene during the coming years. Application-wise, an emerging context is that of RFID-based systems, which have a remarkable impact in leading application scenarios such as supply chain management systems, medical information systems, ambient intelligence systems, and so forth.

As a unifying view of the research challenges we address in this article, we can see the problem of efficiently querying data streams as a basic problem for the more general one concerning mining data streams. In fact, typically, stream mining techniques process data within a fixed time window over the stream (e.g., the last n items, with $n > 0$), and data and patterns to be processed are accessed via meaningful queries over such a bulk of data. In other words, we can think of the problem of querying data streams as the core layer of the general problem of mining data streams, which, indeed, is properly focused on the application layer of the target topic, that is, how to extract useful information and knowledge from streams.

A very popular way of processing data streams is maintaining in memory a synopsis of the stream, that is, a succinct, bounded representation of the unbounded stream. Usually, this synopsis is built and maintained with respect to a fixed time window over the stream, which, in turn, considers a compressed portion of the stream, thus retrieving approximate answers. A pertinent problem in this respect is how to define the width of the window, or how to discern between relevant and irrelevant information in order to improve the quality of the mining task. On the other hand, being that data-stream query and mining tasks are of a trendy, explorative nature, approximate answers perfectly fit the goals and requirements of analysis tools for data streams.

Given this computational model, a significant problem in data-stream research is how to construct in memory the stream synopsis in order to efficiently support query and mining tasks over data streams. It is a matter to note that this line of research is heavily influenced by previous efforts in the context of database data warehouse compression techniques, which, similar to the techniques we investigate here, aim at reducing data size to improve the performances of query and mining algorithms over massive amounts of data sets (e.g., corporate databases, data cubes, etc).

Following these considerations, in this article we provide an overview of state-of-the-art synopsis data structures for data streams by putting in evidence benefits and limitations of each of them in efficiently supporting representation, query, and mining tasks over data streams.

BACKGROUND

To understand the background of data-stream processing research, we needed to start with a comparative analysis between traditional DBMS and next-generation DSMS, trying to put in evidence similarities and differences (Koudas & Srivastava, 2005).

Without any loss of generality, DBMS and DSMS can be compared with respect to the following entities composing their respective representation and query layers. In traditional DBMS, models are represented in terms of persistent relations among data, whereas in novel DSMS, they are represented in terms of transient relations among data. In DBMS, relations are represented and managed as bags of tuples, whereas in DSMS, relations are represented and managed as sequences of tuples. Regarding data update, in DBMS, updates happen in forms of modifications of data, while in DSMS, updates happen in forms of appends of new tuples to the stream. Next, queries in DBMS are reasonably complex and tend to identify what data populate the database, while queries in DSMS are extremely sophisticated and tend to support advanced analysis over data (e.g., trend analysis, pattern detection, prediction, etc.). In DBMS, queries are transient; that is, they change depending on the particular application (e.g., in a sales database, a customer could first ask for the parts having a price under a given threshold, and then for the sale points located in a given region). DSMS queries, on the other hand, are persistent; that is, they do not change over time. For example, a data-stream client application could define a fixed-range query $Q = \langle [T_i, T_j], [P_i, P_j] \rangle$ (Ho, Agrawal, Megiddo, & Srikant, 1997) retrieving the sum of the sales of a corporate company during a certain interval of time (i.e., $[T_i, T_j]$) and concerning a certain class of products (i.e., $[P_i, P_j]$); this query is then posed to the corporate data-stream server and maintained unchanged over time. Referring to query answers, DBMS clients are typically interested in exact answers over a set of tuples, whereas, DSMS clients admit approximate answers as streams are potentially unbounded so that they cannot be managed with limited memory. On the other hand, approximate answers generally suffice to typical goals of data-stream analysis tools that, usually, are interested in studying trends of data streams rather than retrieving exact answers over fixed time windows. Queries in DBMS are evaluated in arbitrary ways, and multistep query algorithms are usually employed, whereas in DSMS, queries must be evaluated in a single pass; that is, each

tuple of the stream can be examined at most once. Finally, regarding the query plan, in DBMS, the query plan is fixed once determined by the query optimizer, while in DSMS, due to the fact that the nature of streams is rapidly changing (e.g., usually, the rate of streams changes continuously), the query plan must be adaptive depending on the current features of the stream.

SYNOPSIS DATA STRUCTURES FOR DATA-STREAM PROCESSING

In the data-stream processing research field, a popular solution that aims at mitigating high-computational requirements posed by such a kind of data is represented by synopsis data structures. These initiatives propose computing a succinct representation of the unbounded stream, and then performing query answering (also, as said above, in support of data-stream mining algorithms) against such a (succinct) representation instead of the original stream. It is a matter to note that these ideas have been inherited by previous research efforts done in database and data warehouse compression, which, however, is still an active research area. Synopsis data streams produce approximate answers, of course, and finding deterministic and probabilistic bounds over them is an exciting research challenge (Cuzzocrea 2005b; Cuzzocrea & Wang, in press; Garofalakis & Gibbons, 2002; Garofalakis & Kumar, 2004). Another distinctive characteristic of this way of processing data streams is that algorithms founding on it apply in a single-pass mode, that is, by processing each item of the data stream at most once without having the chance of processing the stream many times. This feature is in strident contrast with traditional knowledge extraction methodologies, which instead process data many times in order to progressively refine the quality of results. It poses novel frontiers to be investigated; as an example, it should be noted that these single-pass algorithms for data streams could be applied to support information and knowledge extraction from (massive) terabyte databases, which is a research topic still waiting for a breaking solution. Notice that accessing massive databases is time consuming and resource intensive so that single-pass algorithms would be particularly useful in such a field.

To provide a comprehensive, rational overview of synopsis data structures for data streams, first we need to define a general classification of them. From the analysis of their characteristics, synopsis for data streams can be classified into the following classes (Garofalakis, Gehrke, & Rastogi, 2002):

- **Basic techniques:** These techniques are those coming from the database and data warehouse compression solutions and consist of roughly applying such solutions to the stream or a consistent window of it, thus generating a synopsis stream with respect to the evaluating (with some degree of approximation) queries.
- **Sketch-based computation techniques:** These techniques are evolutions of the previous ones where data summaries, which can be intended as further specializations of synopsis data structures, are built from the sketch by processing the stream via (elegant) statistical models, and under a space bound constraint are tighter than the previous case.
- **Advanced techniques:** These techniques represent the (current) last frontier in data-stream processing research by moving the attention toward advanced topics such as (a) distinct value estimation, which aims at finding the number of distinct values in the target stream, (b) sliding-window-based computation, which purports to use time windows that slide over the stream in order to capture and process useful data of the stream (e.g., maintaining the maximum value over the last *n* items of the stream), and (c) advanced analysis tools over data streams, which aim at exporting consolidated analysis techniques such as OLAP within the data-stream context.

BASIC TECHNIQUES

Among the basic techniques for computing synopsis over data streams, the most relevant ones are sampling, histograms, and wavelets. As said above, all these techniques are inherited from the database and data warehouse compression research area, which has reached a significant degree of maturity during the last decade.

Sampling. Sampling is still the simplest yet effective technique for databases and data cubes approximation (Acharya, Gibbons, & Poosala, 1999; Acharya, Gibbons, Poosala, & Ramaswamy, 1999; Babcock, Chaudhuri, & Das, 2003; Chaudhuri, Das, Datar, Motwani, & Rastogi, 2001; Ganti, Lee, & Ramakrishnan, 2000; Gibbons & Matias, 1998; Hellerstein, Haas, & Wang, 1997). Concerning the data-stream processing context, the main idea of sampling is noticing that, very often, a small number of samples extracted from the stream represents the original stream very well. This assumption is further corroborated if one thinks of real-life applications such as sensor-network data processing systems. Consider, in fact, the readings coming from a sensor network monitoring environmental parameters (e.g., temperature, pressure, humidity, etc.): in this case, it is well known that sensor environmental readings are quite regular; that is, they range uniformly over a certain interval of values (e.g., temperature in Mediterranean regions during a fixed season) so that in the output stream of items we can find regular patterns and trends, and, as a consequence, sampling techniques are able to summarize the stream effectively because each item of the stream is an accurate representative item of a (quite) long sequence of neighboring items. Implementation-wise, sampling is codified in terms of random sampling; that is, samples are extracted from the stream according to a uniform distribution ranging on a given interval $[a, b]$, with $a < b$. With respect to theoretical issues, sampling a stream of data corresponds to build an estimator (Papoulis, 1994) of queries against that stream. A relevant problem in this matter is trying to build an unbiased estimator (Papoulis), that is, an estimator such that the expected value of the answer (i.e., the value retrieved from the original stream) is the actual value of the answer (i.e., the value retrieved from the sample stream), thus minimizing the distance between the estimation of the observed parameter and its exact value. To give other hints, building an unbiased estimator of a given data domain or distribution (like that originated by a data stream) is a nontrivial engagement involving several aspects related to probabilistic and statistical formal models mainly (Papoulis). Another nontrivial issue is how to guarantee that approximate answers are bounded, meaning their values are very close to the values of the (respective) exact answers (note that this aspect is relevant given that the user-client application does not know the values of exact answers when querying the synopsis). To face off this issue, current proposals adopt the so-called tail inequalities (Papoulis), which are a significant research result coming from the theoretical statistical field. Given a random variable X, tail inequalities define bounds for the estimated value of an observed parameter on X (e.g., the instance of X, or a function, i.e., another random variable, defined on the instance of X such as *sum*[X] or *avg*[X]), thus indicating to us the probability that such an estimated value deviates far from its expectation. From the literature, we have several tail inequalities that allow us to compute such bounds. Letting X be the target random variable with (a) estimated value X_e for an observed parameter of interest (for the sake of simplicity, we consider as parameter of interest just the instance of X, denoted by $\overline{X}$), and (b) variance σ^2, for each $\varepsilon > 0$ we have, according to the Markov's inequality (Papoulis), we get the following.

$$P(\overline{X} \geq \varepsilon) \leq \frac{X_e}{\varepsilon} \tag{1}$$

According to Chebyshev's inequality (Papoulis, 1994),

$$P(\left|\overline{X} - X_e\right| \geq X_e \cdot \varepsilon) \leq \frac{\sigma_X^2}{X_e^2 \cdot \varepsilon^2}. \tag{2}$$

If we instead consider advanced combinations of random variables, such as sums of random variables, more sophisticated probabilistic and statistical tools are available that can be successfully applied to the problem of retrieving formal bounds for data-stream (approximate) query answers. Let $I_X = \{X_1, X_2, \ldots, X_M\}$ be a set of independent random variables such that $0 \leq X_i \leq r$, with i belonging to the range [1:M], let $r > 0$, and let $X_e = \frac{1}{M} \cdot \sum_i \overline{X}_i$ (i.e., the sample mean of the instance of I_X, $I_e = \{\overline{X}_1, \overline{X}_2, ..., \overline{X}_M\}$) be the estimated value to be observed; according to Hoeffding's inequality (Hoeffding, 1963), for each $\varepsilon > 0$ we have,

$$P(\left|\overline{X} - X_e\right| \geq \varepsilon) \leq 2 \cdot e^{\frac{-2M\varepsilon^2}{r^2}}. \qquad (3)$$

This relevant theoretical result has been successfully applied in order to provide probabilistic bounds for approximate answers to conventional OLAP queries against ROLAP data cubes in the context of online aggregation systems (Hellerstein et al., 1997), and range aggregate queries against MOLAP data cubes in the context of advanced, high-performance OLAP engines (Cuzzocrea, 2006).

Reservoir sampling (Vitter, 1985) proposes maintaining a sample s (i.e., a synopsis) of fixed-size M by progressively adding items from the stream with probability $\frac{M}{n}$, such that n is the current number of items of the stream. When a new item is added to the sample, a victim is evicted from the (current) sample randomly. The novelty of this proposal consists of the approach according to which instead of flipping one item at time, the new item is added after skipping a certain number of items in the sample, the number being analytically determined. In the concise sampling proposal (Gibbons & Matias, 1998), duplicates in the sample are stored by means of pairs of a kind, ⟨*value*, *count*⟩, such that *count* counts the number of occurrences of the item with value *value* in the stream. This approach has the positive aspect of boosting the sample size as duplicates are stored by incrementing the parameter *count* simply. Fix the sample size to M and a threshold T, and have each new item added in the sample with probability $\frac{1}{T}$; if the sample size exceeds M, then (a) a new threshold $T' > T$ is selected, (b) each item of the sample with probability $\frac{1-T}{T'}$ is evicted from the sample, and finally, (c) the new value of the threshold is set to T', and, as a consequence, each new item is added with probability $\frac{1}{T'}$.

Histograms. Histograms are compressed data structures with a "statistical spirit," focused on approximating a data domain or a data distribution. In our application context, histograms are used to approximate the frequency distribution of items in the target stream: In this vest, histograms provide another class of synopsis data structures for data-stream processing. Histograms have a long history in the field of selectivity estimation for relational queries within DBMS query optimizers (Kooi, 1980; Piatetsky-Shapiro & Connell, 1984). In this field, histograms are used to summarize the distribution of values of attributes within a relation; at query time, the selectivity of input queries is evaluated against the histogram beforehand to define the optimal query plan (Ioannidis & Poosala, 1999). A related, relevant problem is how to compute multidimensional histograms (Gibbons, Matias, & Poosala, 1997; Poosala & Ioannidis, 1997; Poosala, Ioannidis, Haas, & Shekita, 1996), that is, histograms defined on multiple attributes of a given relation. After this initial goal, histograms were successfully applied to estimate range queries in OLAP (Poosala & Ganti, 1999).

A histogram consists of a set of buckets defined by a partitioned representation of the input data domain or distribution. Buckets are in turn defined over a set of data items of such domain or distribution and store an aggregate of information that summarizes the nature of these

items. Very often, this information is computed via popular SQL (standard query language) aggregate operators like SUM, COUNT, AVG, and so forth. A plethora of histograms have been proposed in literature, each of them focused on covering a particular requirement of the compression task (e.g., space bound, low query error, geometrical or quantitative uniformity of the bucket-based representation, etc.). Poosala (1997) proposes an elegant taxonomy of histogram-based compression techniques that is widely accepted as the reference work in this field.

For what concerns histogram-based techniques for data-stream compression, the most suitable histograms in this respect are equidepth (Muralikrishna & DeWitt, 1998) and *v*-optimal (Ioannidis & Poosala, 1995; Jagadish, Koudas, Muthukrishnan, Poosala, Sevcik, & Suel, 1998) histograms. The criterion of the equidepth histogram is creating buckets such that the number of items within them is the same as much as possible. Given an *n*-dimensional data domain *D*, the equidepth histogram $H_{E\text{-}D}(D)$ is built as follows: (*a*) Fix an ordering of the *n* dimensions $d_0, d_1, \ldots, d_{n-1}$, (b) set $\alpha \approx n^{\text{th}}$ root of the desired number of buckets, (c) initialize $H_{E\text{-}D}(D)$ to the input data distribution of *D*, (d) for each *i* in {0, 1, …, *n*-1}, split each bucket in $H_{E\text{-}D}(D)$ in α equidepth partitions along d_i, and finally (e) return resulting buckets to $H_{E\text{-}D}(D)$. This technique presents some limitations. Fixing α and a dimension ordering can result in poor partitions, and, consequently, there could be a limited level of "bucketization." The criterion of the *v*-optimal histogram is instead creating buckets such that the frequency variance within them is minimized; in other words, the problem addressed by the *v*-optimal approach can be formulated as follows:

$$minimize \sum_b \sum_i \left(f(b(i)) - \frac{AGG(b)}{\|b\|} \right)^2, \quad (4)$$

such that (*i*) $f(b(i))$ is the frequency of the item $b(i)$ of the bucket *b*, $AGG(b)$ is the value of the aggregate information stored in *b*, and $\|b\|$ is the selectivity (i.e., the volume) of *b*. In doing this, given an *n*-dimensional data domain *D*, the *v*-optimal histogram $H_{V\text{-}O}(D)$ is built by using a spanning-based approach that finds, among all the possible ones, the best bucket-based partition of *D* that satisfies Equation 4. Obviously, this approach is computationally inefficient on massive-in-size data domains: As a consequence, in the context of database and data warehouse compression techniques, several alternative greedy solutions have been proposed with success (e.g., Poosala & Ioannidis, 1997).

A pertinent problem is how to query a histogram. In this case, since buckets store aggregate information, retrieving the value of queries involving intrabucket data can result in relevant (query) errors as aggregate information causes the loss of details. This problem has been faced off according to various solutions with different fortunes. As an example, linear interpolation is very often used; however, it has been demonstrated that linear interpolation works well when data are uniformly distributed (Cuzzocrea, 2005a), or, in other words, when the continuous value assumption (CVA) holds (Colliat, 1996). For skewed (i.e., asymmetric) data distributions, different solutions have been proposed, such as outliers indexing and management (Chaudhuri et al., 2001; Cuzzocrea, Wang, & Matrangolo, 2004). Beyond these techniques inherited from database and data warehouse compression solutions, the most successful way of applying histogram-based approaches to the specific goal of compressing (time windows of) data streams consists of adopting sampling techniques to guide the histogram construction phase; that is, histograms are computed over samples of the target stream instead of the original stream. In this regard, relevant initiatives are those of Chaudhuri, Motwani, and Narasayya (1998), who investigate how to efficiently construct histograms via sampling in such a way as to obtain unbiased estimators for (approximate) queries, and Gibbons et al. (1997), who study how to incrementally maintain a histogram via progressive sampling.

Wavelets. Wavelets (Stollnitz, Derose, & Salesin, 1996) are a mathematical transformation

that defines a hierarchical decomposition of functions (representing signals or data distributions) into a set of coefficients. They were originally applied in the field of image and signal processing. Recent studies have shown the applicability of wavelets to selectivity estimation (Matias, Vitter, & Wang, 1998), as well as to the approximation of both specific forms of query (like range queries; Vitter, Wang, & Iyer, 1998) and "general" queries (using join operators; Chakrabarti, Garofalakis, Rastogi, & Shim, 2000) over data cubes. It has been shown that wavelet-based techniques improve the histogram-based ones in the summarization of multidimensional data (Vitter et al.), and thus they have been used for approximate query answering (Chakrabarti et al.). Concerning the data-stream context, wavelets are mainly used as a support for the construction of histograms more accurate than those provided by traditional techniques, presented above, thus achieving a combined approach that successfully exploits the positive aspects of both the techniques. The resulting data structures are known under the wavelet-based histograms term first proposed in Matias et al. (1998). The key idea of such an initiative is to use a compact subset of wavelet coefficients for approximating the input data distribution. This approach has provided good results in range query selectivity estimation issues, and has outperformed the performances of sampling and histograms (Matias et al.). The problem of the dynamic maintenance of wavelet-based histograms, which assumes a leading role in data-stream processing, is instead investigated in Matias, Vitter, and Wang (2000), where it is recognized that updates in a single distribution value can affect the values of many wavelet coefficients through propagations like paths to the root of the decomposition tree; as a consequence, a dynamic technique in which, as the distribution changes, the most significant (e.g., largest) coefficients are updated consequentially is proposed. Finally, recent proposals have investigated the issue of finding error guarantees for wavelet-based synopsis (Garofalakis & Gibbons, 2002; Garofalakis & Kumar, 2004), with a similar spirit of what was done for histogram-based synopsis (Cuzzocrea, 2005b; Cuzzocrea & Wang, in press).

SKETCH-BASED COMPUTATION TECHNIQUES

Sketch-based computation techniques overcome limitations of basic techniques: Indeed, as said previously, the latter are re-adaptations of results developed in different contexts, that is, database and data warehouse compression approaches. These limitations can be briefly summarized as follows: (a) One-dimensional histograms are not able to capture correlations among data (which are instead very common in real-life data streams), (b) samples are very poor in supporting the approximate evaluation of join queries since sampling tends to lose relationships among data (e.g., see Acharya, Gibbons, Poosala, et al., 1999), and (c) multidimensional histograms and wavelets are more precise than previous ones, but they require multiple passes to build the synopsis; this is inappropriate for streaming data, as stream processors cannot process items many times.

The key idea of sketch-based computation techniques is building data summaries from streams, and using summaries at query and mining time instead of the original data. The benefit involved in such an approach is two-fold: From one side, the complexity required by these techniques is $O(\log n)$ in space, with n being the target window of stream items; from another side, summaries can be computed via consolidated probabilistic and statistical techniques that allow us to infer nice properties on both the representation and the query and mining level, such as probabilistic guarantees over the degree of approximation of the retrieved answers.

Alon, Matias, and Szegedy (1996) pose the basis for computing randomized sketch synopsis for streams. In this proposal, the summary is computed as an inner product between two vectors: F, which is the distribution vector containing f_i items that report the frequency of values within the stream (i.e., $F[i] = f_i$), and ξ, which is a vector of random values ξ_i extracted from an appropriate (input) distribution (i.e., $\xi[i] = \xi_i$). This summary is very simple to compute as it is only needed to add a new ξ_i item of ξ whenever the i^{th} value is seen in

the stream; furthermore, such a summary requires a very small amount of space to be stored, and ξ_i items can be easily obtained by using well-known pseudorandom generators. In addition to this, the statistical nature of this approach combined with probabilistic models built on families of ad hoc devised independent random variables allow tunable probabilistic guarantees to be achieved.

Dobra, Garofalakis, Gehrke, and Rastogi (2002) start from the results of Alon, Gibbons, Matias, and Szegedy (1999) to address the relevant problem of computing sketches for stream joins and multijoins, which have an important impact on real-life data-stream-based applications (e.g., distributed measurement systems). The key intuition of this approach is exploiting coarse statistics on the data stream to meaningfully partition the join-attribute space in such a way as to improve the quality of error guarantees on answering stream joins, with traditional schemes for singleton stream queries (e.g., Hoeffding's bounds) being inefficient for the former class of queries. This approach performs well provably (Dobra et al.), and its most relevant features are the following: (a) Coarse statistics on the stream are collected during the initial pass only, and (b) given the partitioned representation of the join-attribute space A_S, the sketches are built independently for each partition of A_S, so that the estimation of a given stream join query Q can be obtained by linearly combining the estimations of Q computed against each singleton partition sketch, and, similarly, the variance of Q (useful to study the selectivity of Q within a stream processor) can be obtained by linearly combining the variances of Q computed against each singleton partition sketch.

Another way of computing data summaries for streams is exploiting compression techniques, which have been successfully applied in various application fields related to database and data warehouse research, such as the compression of massive databases, Web and XML (extensible markup language) repositories, and data cubes. A relevant proposal following this line is presented in Cuzzocrea, Furfaro, Masciari, Saccà, and Sirangelo (2004), where a framework for supporting aggregate query answering over sensor-network data, which are a significant and very popular case of data streams, is presented. In sensor networks, the amount of data produced by sensors is very large and grows continuously, and queries need to be evaluated quickly in order to extract useful information and knowledge from sensor readings and perform a timely reaction to the physical world. In Cuzzocrea et al., sensor-network readings are collected in an up-to-date snapshot of the monitored world continuously as time passes, and this snapshot is then compressed via a two-dimensional representation scheme that allows us to efficiently support range query evaluation defined on top of an SQL aggregate operator having a historical fashion. This has several benefits for sensor-network data analysis. For instance, a climate disaster-prevention system would benefit from the availability of continuous information on atmospheric conditions in the last hour. Similarly, a network congestion detection system would be able to prevent network failures by exploiting the knowledge of network traffic during the last minutes.

In more detail, the proposal in Cuzzocrea, Furfaro, et al. (2004) is based on a two-dimensional, hierarchical summarization of data streams, called quad-tree window (QTW), embedded into a flexible indexing structure. This structure is also compressed, meaning that it is updated continuously as new sensor readings arrive, and when the available storage space is not enough to host new data, some space is released by compressing the oldest data. Adopting this approach, the recent knowledge on the system is represented with more detail than the old one. Note that the recent knowledge is usually more relevant to extract for the context of applications on sensor data. Specifically, data streams are collected and organized into a two-dimensional array (i.e., the QTW), where the first dimension corresponds to the set of sensors and the other dimension corresponds to the time. In turn, the time is divided into intervals Δt-wide and each cell $\langle s_i, \Delta t_j \rangle$ of the array represents the sum of all the readings generated by the source s_i during the time interval Δt_j.

The time granularity is the critical parameter for such an aggregation data structure. That is, the coarser the granularity, the lower the accuracy of the representation. Thus, time granularity needs to be chosen accurately, depending on the particular application domain monitored by the sensor network. Using this data representation, an aggregate query $Q = \langle \{s_i,\ldots,s_j\},[t_{start},\ldots,t_{end}]\rangle$ on summarized data streams is defined by two ranges: $R_S = [s_i:s_j]$ and $R_T = [t_{start}:t_{end}]$; the first one is that on the sensor dimension and the second one is that on the time dimension. The (approximate) answer to Q, denoted by $A(Q)$, is obtained by applying the aggregate operator required by Q on the readings produced by the sensors belonging to the range R_S and during the time interval defined by R_T, and making use of linear interpolation for those aggregate domains that overlap Q. Specifically, the hierarchical data structure presented in Cuzzocrea, Furfaro, et al. (2004) is particularly suitable for the aggregate operator SUM, but deriving answers for other aggregate operators (like COUNT and AVG) is straightforward.

ADVANCED TECHNIQUES

Advanced techniques represent the latest result for synopsis data-stream structures and are the natural evolution of the above-described techniques.

Finding the number of distinct items in a stream of readings with domain $[r_i, r_j]$, such that $r_j > r_i$, is a very important problem in data-stream processing, which assumes a critical role in specialized domains such as network management systems, where finding duplicated IP (Internet protocol) addresses is very useful to prevent the network from unauthorized accesses and attacks. To face off this problem, there exist several approaches. Domingos and Hulten (2000) use uniform samples collected from the stream, and then devise an appropriate unbiased estimator to detect duplicates with respect to the sample stream. Flajolet and Martin (1985) and Alon et al. (1996) use single-pass algorithms and hash functions to map stream items to bit positions in a bitmap. Gibbons (2001) extends the latter initiatives in order to handle distinct-aware query predicates. The sliding window model, which is based on computing (sliding) windows of interesting and relevant portions of the target data stream by considering the last n items, and then meaningfully using such windows to support approximate query answers over the stream, is typically used to maintain statistics-based information and patterns describing the stream items whose time stamps are contained within the target window. As an example, maintaining the maximum value over the last n items of the stream is a popular yet simple way of storing statistics for streaming data. It has been demonstrated that this computational model for streams gives good performance in specialized domains where the recent data have more value with respect to the old data, such as sensor-network and stock data processing. A successful approach following the sliding window model is given in Dasu, Johnson, Muthukrishnan, and Shkapenyuk (2002), where several classes of histogram-based statistics (specifically, founding on the aggregate operator COUNT) over sliding windows are computed and used to evaluated bounded approximate answers with very low space and time complexities.

More recently, novel advanced techniques for data-stream processing appeared. Zhu and Shasha (2002) propose adopting the discrete Fourier transform (DFT) to maintain statistics over sliding windows defined on readings of thousands of data streams for online monitoring and decision-making purposes. To this end, summaries of data streams are employed, similar to previous experiences in this field. Experimental results in Zhu and Shasha (2002) show that DFT scales well on massive data streams, and, above all, provides good I/O (input/output) performance. In some sense, DFT defines a sort of improved synopsis for data streams, with solid theoretical foundations that also allow nice mathematical transformations that could be useful to improve the overall quality of analysis tools over data streams. Aggarwal, Han, Wang, and Yu (2003) focus on the problem of clustering data streams, which is of relevant interest for a large family of practical

data-stream applications. Contrary to classical application-centered approaches, Aggarwal et al. propose dividing the overall clustering process in multiple computational components that interact between summary statistics computed over the target stream. These statistics are obtained via the traditional microclustering technique, which, thanks to its nice additivity property, meaningfully couples with the problem of computing data-stream clusters. A relevant problem is just how to compute high-quality statistics capable of making the clustering successful in opposition to the problematic issue of dealing with rapidly evolving data streams (which is the proper application context of this proposal); to this end, authors define an innovative pyramidal time-frame model that provides an effective trade-off between the storage requirements of representing-in-memory clusters and the ability of recalling statistics from different time horizons. Finally, in Cai, Clutter, Pape, Han, Welge, and Auvil (2004), the authors describe the demonstration of the system MAIDS (Mining Alarming Incidents from Data Streams), which integrates several OLAP methodologies and technologies such as multidimensional and multiresolution visions of data, cuboid lattices, dimensional hierarchies, and so forth to analyze and mine data streams on the basis of their intrinsic multidimensional nature.

CONCLUSION

Data streams pose new challenges for the database and data warehouse research community, with novelties that were unrecognized in DBMS research. Furthermore, emerging applications, such as environmental parameter monitoring systems, high-performance distributed measurement systems, sensor networks, and RFID-based applications, demand more and more for the development of the so-called DSMS, a data-stream management system able to efficiently support representation, indexing, and query functionalities over data streams with similar capabilities achieved within modern DBMS. A very efficient way of handling data streams is computing synopsis data structures over them, and using such structures to support query and mining tasks. A plethora of synopsis techniques for data streams has been proposed during the last years, each of them focused on capturing specialized characteristics of the stream under constraints of different nature (e.g., space bound, low query error, accuracy of answers, etc.). According to these considerations, this article has provided an overview of state-of-the-art techniques for representing, querying, and mining data streams, thus posing the basis for further research in this field.

FUTURE TRENDS

A lot of research needs to be developed in the context of data-stream processing. In fact, the particular characteristics of data streams make DBMS technology inappropriate for efficiently supporting query and mining functionalities over data streams. Among the widely recognized directions of research in this filed, here we want to highlight the following: (a) devising more precise accuracy bounds for approximate query answers over data streams, beyond the capabilities of the above-described schemes, (b) moving from approximate aggregate query answering to set-valued query answering, which is particularly difficult over rapidly evolving data streams, (c) devising schemes for multiple data-stream query processing, with particular emphasis on join queries over different streams, (d) adding Boolean and more advanced predicates within data-stream query processors, (e) providing support for blocking query operators over data streams, (f) setting up an appropriate stream query algebra similar to the DBMS relational algebra, (g) devising efficient memory management schemes for data-stream query processing, (h) designing novel architectures for the integration of data-stream processing systems and databases, (i) studying the relevant problem of indexing data streams as a critical component of the DSMS, and (j) developing advanced analysis methodologies for

data streams, such as filtering, association rules, and correlations discovery, that will allow us to improve the quality of next-generation data-stream mining tools.

REFERENCES

Acharya, S., Gibbons, P. B., & Poosala, V. (1999). AQUA: A fast decision support system using approximate query answers. *Proceedings of the 25th International Conference on Very Large Data Bases* (pp. 754-757).

Acharya, S., Gibbons, P. B., Poosala, V., & Ramaswamy, S. (1999). Join synopses for approximate query answering. *Proceedings of the 1999 ACM International Conference on Management of Data* (pp. 275-286).

Aggarwal, C., Han, J., Wang, J., & Yu, P. (2003). A framework for clustering evolving data streams. *Proceedings of the 29th International Conference on Very Large Databases* (pp. 81-92).

Alon, N., Gibbons, P. B., Matias, Y., & Szegedy, M. (1999). Tracking join and self-join sizes in limited storage. *Proceedings of the 18th ACM International Symposium on Principles of Database Systems* (pp. 10-20).

Alon, N., Matias, Y., & Szegedy, M. (1996). The space complexity of approximating the frequency moments. *Proceedings of the 28th ACM International Symposium on Theory of Computing* (pp. 20-29).

Babcock, B., Babu, S., Datar, M., Motawani, R., & Widom, J. (2002). Models and issues in data stream systems. *Proceedings of the 21st ACM International Symposium on Principles of Database Systems* (pp. 1-16).

Babcock, B., Chaudhuri, S., & Das, G. (2003). Dynamic sample selection for approximate query answers. *Proceedings of the 2003 ACM International Conference on Management of Data* (pp. 539-550).

Cai, Y. D., Clutter, D., Pape, G., Han, J., Welge, M., & Auvil, L. (2004). MAIDS: Mining alarming incidents from data streams. *Proceedings of the 2004 ACM International Conference on Management of Data* (pp. 919-920).

Chakrabarti, K., Garofalakis, M., Rastogi, R., & Shim, K. (2000). Approximate query processing using wavelets. *Proceedings of the 26th International Conference on Very Large Data Bases* (pp. 111-122).

Chaudhuri, S., Das, G., Datar, M., Motwani, R., & Rastogi, R. (2001). Overcoming limitations of sampling for aggregation queries. *Proceedings of the 17th IEEE International Conference on Data Engineering* (pp. 534-542).

Chaudhuri, S., Motwani, R., & Narasayya, V. (1998). Random sampling for histogram construction: How much is enough? *Proceedings of the 1998 ACM International Conference on Management of Data* (pp. 436-447).

Colliat, G. (1996). OLAP, relational, and multidimensional database systems. *SIGMOD Record, 25*(3), 64-69.

Cuzzocrea, A. (2005a). Overcoming limitations of approximate query answering in OLAP. *Proceedings of the 9th IEEE International Conference on Database Engineering and Applications* (pp. 200-209).

Cuzzocrea, A. (2005b). Providing probabilistically-bounded approximate answers to non-holistic aggregate range queries in OLAP. *Proceedings of the 8th ACM International Working Group on Data Warehousing and OLAP* (pp. 97-106).

Cuzzocrea, A. (2006). Improving range-sum query evaluation on data cubes via polynomial approximation. *Data & Knowledge Engineering, 56*(2), 85-121.

Cuzzocrea, A., Furfaro, F., Masciari, E., Saccà, D., & Sirangelo, C. (2004). Approximate query answering on sensor network data streams. In A. Stefanidis & S. Nittel (Eds.), *Sensor-based distribution geocomputing* (pp. 53-72). CRC Press.

Cuzzocrea, A., & Wang, W. (in press). Approximate range-sum query answering on data cubes with probabilistic guarantees. *Journal of Intelligent Information Systems.*

Cuzzocrea, A., Wang, W., & Matrangolo, U. (2004). Answering approximate range aggregate queries on OLAP data cubes with probabilistic guarantees. In *Lecture notes in computer science: Vol. 3181. Proceedings of the 6th International Conference on Data Warehousing and Knowledge Discovery* (pp. 97-107). Springer-Verlag.

Dasu, T., Johnson, T., Muthukrishnan, S., & Shkapenyuk, V. (2002). Mining database structure or how to build a data quality browser. *Proceedings of the 2002 ACM International Conference on Management of Data* (pp. 61-72).

Dobra, A., Garofalakis, M., Gehrke, J., & Rastogi, R. (2002). Processing complex aggregate queries over data streams. *Proceedings of the 2002 ACM International Conference on Management of Data* (pp. 61-72).

Domingos, P., & Hulten, G. (2000). Mining high-speed data streams. *Proceedings of the 2000 ACM International Conference on Knowledge Discovery from Data* (pp. 31-42).

Flajolet, P., & Martin, G. N. (1985). Probabilistic counting algorithms for data base applications. *Journal of Computer and Systems Science, 31*(2), 20-34.

Ganti, V., Lee, M., & Ramakrishnan, R. (2000). ICICLES: Self-tuning samples for approximate query answering. *Proceedings of the 26th International Conference on Very Large Data Bases* (pp. 176-187).

Garofalakis, M. N., Gehrke, J., & Rastogi, R. (2002). Querying and mining data streams: You only get one look. *Proceedings of the 28th International Conference on Very Large Data Bases* (p. 635).

Garofalakis, M. N., & Gibbons, P. B. (2002). Wavelet synopses with error guarantees. *Proceedings of the 2002 ACM International Conference on Management of Data* (pp. 476-487).

Garofalakis, M. N., & Kumar, A. (2004). Deterministic wavelet thresholding for maximum-error metrics. *Proceedings of the 23rd ACM International Symposium on Principles of Database Systems* (pp. 166-176).

Gibbons, P. B. (2001). Distinct sampling for highly-accurate answers to distinct values queries and event reports. *Proceedings of the 27th International Conference on Very Large Data Bases* (pp. 13-24).

Gibbons, P. B., & Matias, Y. (1998). New sampling-based summary statistics for improving approximate query answers. *Proceedings of the 1998 ACM International Conference on Management of Data* (pp. 331-342).

Gibbons, P. B., Matias, Y., & Poosala, V. (1997). Fast incremental maintenance of approximate histograms. *Proceedings of the 23rd International Conference on Very Large Data Bases* (pp. 466-475).

Hellerstein, J. M., Haas, P. J., & Wang, H. J. (1997). Online aggregation. *Proceedings of the 1997 ACM International Conference on Management of Data* (pp. 171-182).

Ho, C.-T., Agrawal, R., Megiddo, N., & Srikant, R. (1997). Range queries in OLAP data cubes. *Proceedings of the 1997 ACM International Conference on Management of Data* (pp. 73-88).

Hoeffding, W. (1963). Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association, 58*(301), 13-30.

Ioannidis, Y. E., & Poosala, V. (1995). Balancing histogram optimality and practicality for query result size estimation. *Proceedings of the 1995 ACM International Conference on Management of Data* (pp. 233-244).

Ioannidis, Y. E., & Poosala, V. (1999). Histogram-based approximation of set-valued query answers. *Proceedings of the 25th International Conference on Very Large Data Bases* (pp. 174-185).

Jagadish, H. V., Koudas, N., Muthukrishnan, S., Poosala, V., Sevcik, K., & Suel, T. (1998). Optimal histograms with quality guarantees. *Proceedings of the 24th International Conference on Very Large Data Bases* (pp. 275-286).

Kooi, R. P. (1980). *The optimization of queries in relational databases.* Unpublished doctoral dissertation, CWR University.

Koudas, N., & Srivastava, D. (2005). Data stream query processing: A tutorial. *Proceedings of the 29th International Conference on Very Large Data Bases* (p. 1149).

Matias, Y., Vitter, J. S., & Wang, M. (1998). Wavelet-based histograms for selectivity estimation. *Proceedings of the 1998 ACM International Conference on Management of Data* (pp. 448-459).

Matias, Y., Vitter, J. S., & Wang, M. (2000). Dynamic maintenance of wavelet-based histograms. *Proceedings of the 26th International Conference on Very Large Data Bases* (pp. 101-110).

Muralikrishna, M., & DeWitt, D. J. (1998). Equi-depth histograms for estimating selectivity factors for multi-dimensional queries. *Proceedings of the 1988 ACM International Conference on Management of Data* (pp. 28-36).

Papoulis, A. (1994). *Probability, random variables, and stochastic processes* (2nd ed.). McGraw-Hill.

Piatetsky-Shapiro, G., & Connell, C. (1984). Accurate estimation of the number of tuples satisfying a condition. *Proceedings of the 1984 ACM International Conference on Management of Data* (pp. 256-266).

Poosala, V. (1997). *Histogram-based estimation techniques in database systems.* Unpublished doctoral dissertation, University of Wisconsin, WI.

Poosala, V., & Ganti, V. (1999). Fast approximate answers to aggregate queries on a data cube. *Proceedings of the 11th IEEE International Conference on Statistical and Scientific Database Management* (pp. 24-33).

Poosala, V., & Ioannidis, Y. E. (1997). Selectivity estimation without the attribute value independence assumption. *Proceedings of the 23rd International Conference on Very Large Databases* (pp. 486-495).

Poosala, V., Ioannidis, Y. E., Haas, P. J., & Shekita, E. (1996). Improved histograms for selectivity estimation of range predicates. *Proceedings of the 1996 ACM International Conference on Management of Data* (pp. 294-305).

Stollnitz, E. J., Derose, T. D., & Salesin, D. H. (1996). *Wavelets for computer graphics.* Morgan Kauffmann.

Vitter, J. S. (1985). Random sampling with a reservoir. *ACM Transactions on Mathematical Software, 11*(1), 37-57.

Vitter, J. S., Wang, M., & Iyer, B. (1998). Data cube approximation and histograms via wavelets. *Proceedings of the 7th ACM International Conference on Information and Knowledge Management* (pp. 96-104).

Zhu, Y., & Shasha, D. (2002). StatStream: Statistical monitoring of thousands of data streams in real time. *Proceedings of the 28th International Conference on Very Large Databases* (pp. 358-369).

KEY TERMS

Continuous Query: Query defined over the flooding data stream and producing in output a stream of answers.

Data Stream: An unconventional information source that produces unbounded, continuously flooding data.

Data-Stream Pattern: A repetitive, regular sequence of items of the data stream.

On-Time Query: Query defined over a fixed temporal snapshot of the data stream.

Predicate of a Query: The condition expressed by the query in its *where* clause.

Query Optimizer: The DBMS component devoted to optimize, based on relational algebra rules, the input query in order to find the optimal query plan with respect to spatial and temporal constraints.

Reading: The point value produced by a stream source.

Selectivity of a Query: The number of tuples involved by evaluating the query.

Time Stamp: The time instant related to a stream reading and reporting the (absolute) time in which the reading was produced.

Time Window: A fixed interval of time when the data stream is processed for query and mining purposes.

Chapter LXXVI
GR-OLAP:
Online Analytical Processing of Grid Monitoring Information

Julien Gossa
LIRIS–INSA Lyon, France

Sandro Bimonte
LIRIS–INSA Lyon, France

INTRODUCTION

The Grid is an emerging solution for sharing resources through a network. It is meant to manage heterogeneous resources in world-scale multi-institutional networks. Grid resources monitoring and network monitoring are very active research areas with actually efficient solutions. Unfortunately, these solutions are limited in terms of analysis of the gathered data. Our proposition is to use data warehouse (DW) and online analytical processing (OLAP) technologies on Grid monitoring information. This allows new complex analyses of crucial importance for Grid users' everyday tasks. Unfortunately, the implementation raises several challenging issues.

This article is organized as follows. First, we introduce concepts of DW, OLAP, and Grids, and we discuss recent advances in Grid monitoring as well as the needs and usage of the Grid users. Then we present our conceptual and implementation solutions. Finally, we discuss our main contribution and point out the future works.

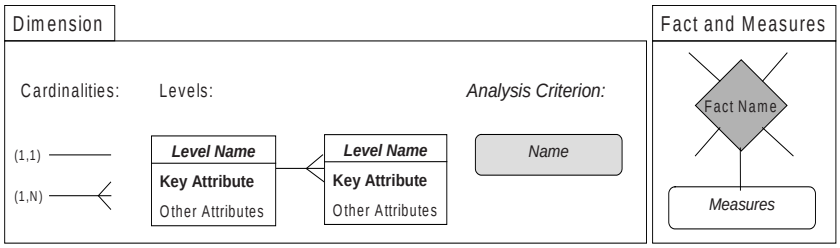
BACKGROUND: DATA WAREHOUSE & OLAP

Definition and Usage

DW in addition to OLAP technologies intend to be an innovative decision support for business intelligence and knowledge discovery. It has now become a leading topic in business organizations as well as in the research community. The main motivation is to take benefits from the enormous amount of data in distributed and heterogeneous databases to enhance data analysis and decision making (Kimball & Ross, 2002).

A DW is a subject-oriented, integrated, nonvolatile, and time-variant collection of data stored in a single site repository and collected from multiple

Figure 1. MultiDimEr conceptual multidimensional model



sources (Inmon, 1996). Information in the DW is organized following a multidimensional model in order to allow precomputation and fast access to summarized data in support of management's decisions. This multidimensional model organizes data in analysis axes called dimensions. Analyzed subjects or facts are characterized by metrics called measures. Dimensions can be organized following hierarchy schemas, thus allowing navigation through different levels of detail of analysis.

An OLAP server calculates and optimizes the hypercube, that is, the set of fact values for all combinations of dimension instances (called members). In order to optimize accesses to the data, query results are precalculated in the form of aggregates. This allows the decision makers to explore the different dimensions at different granularities. This analysis process is conducted by navigating into the multidimensional cube through some OLAP operators (roll-up, drill-down, slice, rotate, etc.).

Finally, interactive user interfaces (OLAP clients) have been developed to support knowledge discovery, promoting the iterative nature of the analysis process. OLAP clients visually represent the multidimensional structure of the hypercube and formulize multidimensional queries. The most adopted data presentation paradigm is the pivot table, a 2-D spreadsheet with associated subtotals and totals. It supports complex data by nesting several dimensions on the *x*- or *y*-axis and displaying data on multiple pages.

The three-tier architecture composed of a DW, an OLAP server, and an OLAP client effectively allows multidimensional analysis.

In the following, we will use the conceptual multidimensional model MultiDimEr (Malinowski & Zimányi, 2006) in order to describe our proposal. The details of the model are presented in Figure 1.

BACKGROUND: THE GRID AND ITS MONITORING

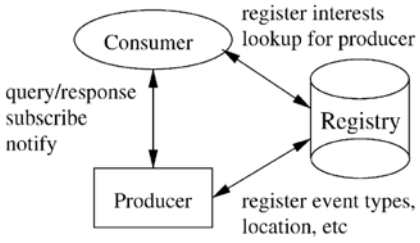
The term *the Grid* was coined in the mid 1990s to denote a paradigm of distributed computing infrastructure for advanced science and engineering (Foster & Kesselman, 1994).

A Glimpse at the Grid Computing

The real and specific problem that underlies the Grid concept can be summarized as "coordinated resource sharing and problem solving in dynamic, multi-institutional virtual organizations" (Foster, Kesselman, & Tuecke, 2001). This sharing concerns direct access to resources (e.g., computers or hosts, software, data, etc.) as required by a range of collaborative problem-solving and resource brokering strategies emerging in industry, science, and engineering. It needs high control: Resource providers and consumers must define clearly and carefully what is shared, who is allowed to share what, and the conditions of the sharing. A set of individuals and/or institutions defined by such sharing rules form what is called a virtual organization (VO). An example of VO is the application service providers, storage service providers, cycle providers, and consultants engaged by a car manufacturer to perform scenario evaluation during planning for a new factory.

In such an information system, the monitoring definitively takes a crucial place.

Figure 2. GMA logical schema



A Glimpse at the Grid Monitoring

The Grid is a large-scale, highly heterogeneous distributed system that supports scattered communities to form virtual organizations in order to collaborate for the realization of common goals. Consequently, users must be supported in finding and keeping track of resources of interest. This is the main purpose of Grid information services (GISs) (Czajkowski, Fitzgerald, & Foster, 2001). They systematically collect information regarding the current status of Grid resources, a process known as monitoring.

In addition to information services, monitoring is also crucial in a variety of cases such as scheduling, resources brokering, data replication, accounting, performance analysis, and optimization of distributed systems, individual applications, self-tuning applications, and so forth (Balaton, Kacsuk, & Podhorszki, 2001).

As the Grid is multi-institutional (i.e., composed of several sites under different authorities), administration-level information (like hosts description, physical and logical topology, IP [Internet protocol] addressing plan, routing information, etc.) is not available for anyone and from anywhere in the Grid. Consequently, to support their decision making, users and applications have to rely on user-level available data gathered by monitoring tools.

Recent developments have led to effective tools providing the indexing of hosts, data, and services available on the Grid. The most advanced monitoring solution is the Grid monitoring architecture (GMA; Balaton et al., 2001). It responds to the main constraint of Grid context, which is the users' and resources' dynamicity. GMA is based on a consumer-producer pattern as shown in Figure 2. Resources register with the monitor every time they are reachable. Users or applications make push or pull requests to the monitor to get references to the desired resources. GMA is hierarchical with at least two levels: The first is associated with the VO and the second is local to traditional organizations.

There are several GMA-compliant Grid resource monitors. They differ by the kind of entities they manage. MDS (Czajkowski et al., 2001) and R-GMA (Cooke, Gray, & Nutt, 2004) focus on monitoring physical resources like CPU and storage, while others like SCALEA-G (Truong & Fahringer, 2004) aim at managing

Table 1. The classical metrics of the Grid

CPUSpeed	host	static	MHz	speed of the CPUs of the host
CPUFree	host	dynamic	%	percentage of availability of the CPUs
Mem	host	static	Mb	total amount of RAM memory space of the host
MemFree	host	dynamic	%	percentage of availability of RAM
Disk	host	static	Mb	total amount of RAM disk space of the host
DiskFree	host	dynamic	%	percentage of availability of disk space
ServiceSet	host	static	set	set of description of the services run on the host
Latency	link	dynamic	ms	data propagation time from one host to one other
Bandwidth	link	dynamic	Mb/s	amount of data that can be sent in a period

software resources too. They differ also by some implementation considerations, especially the storage format and query language. In R-GMA, the storage is implemented like a virtual relational database queried using an SQL- (structured query language) like language, whereas SCALEA-G is based on XML (extensible markup language) and Xquery, and MDS stores resource information in an LDAP-like directory.

These systems focus on host (and sometimes application) monitoring only. Complementary tools like the Network Weather Service (NWS; Wolski, Spring, & Hayes, 1999) provide monitoring information about the network conditions, mainly about the bandwidth and the latency between two given endpoints. NWS is based on a LDAP-like directory and on a GMA-like three-level logical architecture.

Clearly, hosts and network monitoring services are complementary to obtain a full view of the Grid.

The identification of relevant metrics and measurement methods for the Grid environment has been made by the Network Measurements Working Group of the Global Grid Forum (Lowekamp, Tierney, & Cottrell, 2004). Metrics can be related to either host or link between two hosts. They can be either static when they are declared or dynamic when they need to be measured frequently. Table 1 shows the metrics we deal with.

Please note that the links metrics are not necessarily symmetrical: Latency and bandwidth uplink and downlink are not equal.

Finally, users are sometimes interested in more complex metrics called compound metrics. They are built from the raw metrics presented in Table 1. For example, CPUSpeed or CPUFree independently are not sufficient. Users prefer to use *CPUavailability* = *CPUSpeed* × *CPUFree*, which is more representative of the actual computing capacity of a CPU.

User and Usage of the Grid

Three kinds of human actors work on the Grid: end users, service developers, and administrators. They differ by their tasks on the Grid as well as their usages and needs of Grid monitoring.

End Users of the Grid

They are the customers of the available services: They only submit tasks to the Grid. Thus, they should not be concerned with monitoring. Ideally, they only have to describe the task they want to be processed and wait for the result. The Grid middleware should do the rest.

Services Developers for the Grid

They develop the services used by the end users of the Grid and thus need some monitoring information. For instance, knowing the capabilities of the resources they have access to is necessary to dimension their services: There is no need to build a service working on terabytes of data if the infrastructure cannot support it.

Administrators of the Grid

They are responsible for the infrastructure underlying the Grid. As the Grid is multi-institutional, each administrator is in charge of one given part of the Grid only. Thus, they have no access to administrator-level information all across the Grid. Nevertheless, they have to make decisions about the infrastructure everyday. Which sites have the most important capacities? Which sites need to be upgraded? What are the most used

Figure 3. An example of Grid infrastructure

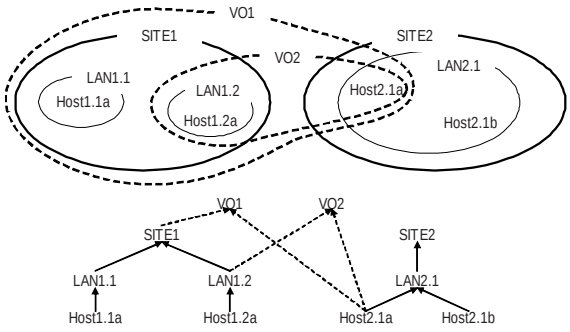
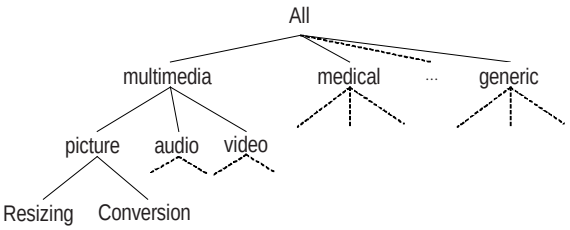


Figure 4. A sample of services classification tree



parts of the Grid? Regarding the software, where are the services related to one given application executed? What kind of services is executed on one given part of the Grid? This task needs advanced knowledge about the Grid condition and topology. This kind of user is the most concerned with our application.

Based on these observations, we can identify three key dimensions for the analysis of the Grid infrastructure.

The Time

Whereas monitoring information upon the last few minutes is enough to make operational decisions, sometimes Grid users need to know the evolution of the whole Grid by month since its creation, or the condition of a particular site minute by minute for a given week to analyze a past incident.

Users are interested in minute-scale to Grid-life-scale aggregations: the minimum, maximum, and average of hosts metrics and links metrics. Moreover they need to consider human usage (with weekdays, weekends, or ferial days for instance).

The Infrastructure (or Space)

As Grids are composed of several sites under different administrations, Grid users sometimes need accurate information about a particular location of the Grid, and even about a given single host. E contrario, they sometimes need global information in order to plan global evolutions, for instance.

At the lowest level, Grids are composed of single hosts. Hosts are gathered in LANs (local area networks). LANs are gathered in sites. The VOs are subsets of the physical architecture. One VO can contain several sites, LANs, and single hosts. An example of such an infrastructure is shown in Figure 3. In addition to hosts, the infrastructure is composed of links representing the connections between hosts.

Users are interested in host-scale to VO-scale aggregations: the minimum, maximum, and average of dynamic links metrics, and the minimum, maximum, average, and sum of dynamic host metrics.

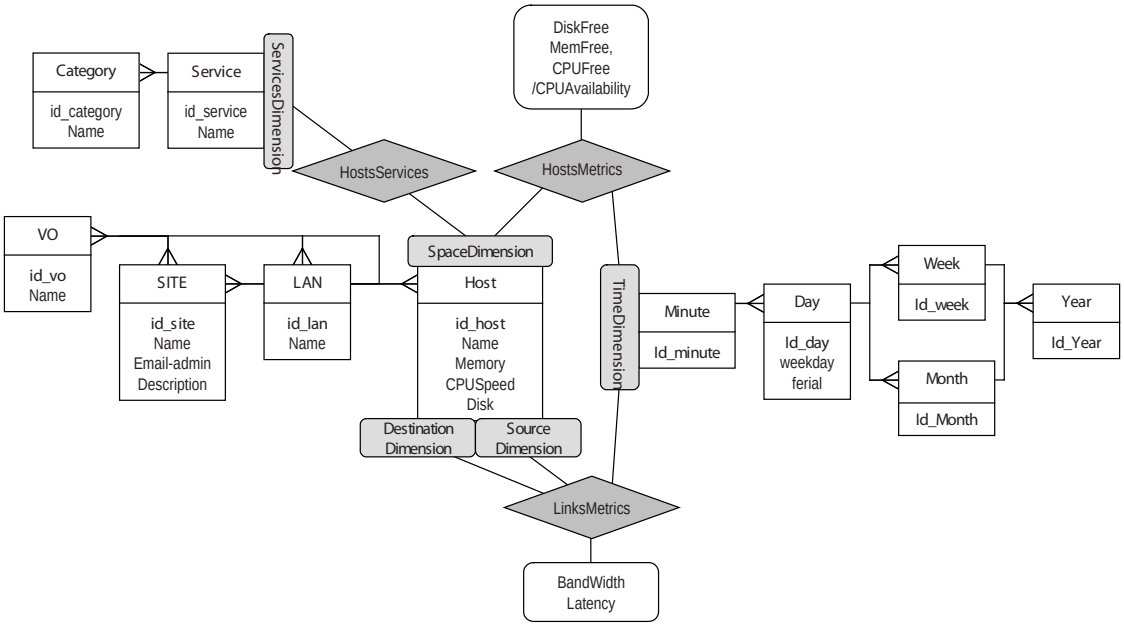
The Services

The applications on a Grid are usually called Grid services or simply services and are based on Web services technology. This aspect is important because services represent the final usefulness of the Grid. Having global as well as fine-grained information about their distribution is very important for Grid developers and administrators. Services can be thematically classified thanks to taxonomy. Unfortunately, such classifications are very hard to find in a complete and ready-to-be-used state because they are expensive professional tools and thus kept unpublished. Nevertheless, we can use a trivial sample classification tree as a proof of concept as shown in Figure 4.

Grid Monitoring Tools Limitations

Existing monitoring solutions lack analysis capabilities to support Grid users' needs. Indeed, the querying is restricted to the most detailed levels of analysis hierarchies only. Thus, the users cannot have a global vision of the status of the Grid and their analysis capabilities are limited. Indeed they concern the current conditions only. For instance, they support requests like this: What are the current CPU speed and load for each host of the grid? On the contrary, by analyzing measures with hierarchical analysis dimensions, the administrator could detect, for example, the

Figure 5. GR-OLAP conceptual multidimensional model



period for which some resources are underused, and so he or she can have a global vision of the Grid behavior.

Moreover, the interfaces of existing monitoring systems are often limited to APIs (application programming interfaces) and are not suitable to support any decision-making process.

Indeed, they show data only in a textual way without any interactive form of graphic or advanced tabular representation. By this way, querying and analyzing Grid monitoring data lacks in interactivity and user friendliness, which are opposite the basis for decision-making processes.

Furthermore, similarities between the multidimensional analysis model and the presented dimensions are evident: DW technologies might be used to store the complex hierarchical services, time, and infrastructure data and OLAP clients to gain an insight into the data, allowing the discovery of unknown phenomena, patterns, and data relationships through interactive pivot tables and graphic displays.

In this way, the Grid monitoring will benefit from the well-known and experienced analysis capabilities of DW and OLAP technologies, permitting the user of the Grid to avoid instinctively handmade analyses suffering of inaccuracy and limitation in terms of complexity.

MAIN THRUST: GR-OLAP

Our proposition is not to build another Grid monitoring solution, but rather to use the data gathered by existing ones to populate a DW and use OL models and tools for analysis. We first have to build a multidimensional model adapted to grid monitoring data. As described in the following section, the design of this hypercube raises challenging issues such as the design of nonstrict and noncovering hierarchies, the use of nonadditive or semiadditive measures, sets of facts leading to a star constellation schema, the design of a shared conforming dimension, and so forth, The accurate design of the hypercube will lead to enhanced analysis capabilities fitting actual needs of Grid users.

Conceptual Multidimensional Model

The conceptual representation of the GR-OLAP application is shown in Figure 5. Its dimensions and facts are described bellow.

Dimensions

SpaceDimension
This dimension is a noncovering and nonstrict hierarchy with several levels: host, LAN, site, and VO. It is nonstrict as many-to-many relationships between members of different levels can subsist; one host, LAN, and/or site can belong to several VOs. In this kind of hierarchy, members can skip some levels. For example, the host level can skip the LAN and site levels, and the LAN level can skip the site level. An example of members of this hierarchy is shown in Figure 3. Host *Host21a* is directly connected to the *VO1* and *VO2*; *LAN12* is directly connected to *VO2*.

TimeDimension
This multiple dimension presents two alternative hierarchies representing the two classical calendar subdivisions: minute, day, week, and year; and minute, day, month, and year. The user has to select between them to navigate in the cube.

ServiceDimension
This dimension is a simple hierarchy as shown Figure 4.

Fact Tables

HostsMetrics
HostsMetrics contains the host monitoring data: CPUFree, MemFree, DiskFree as described in Table 1, and the derived measure CPUAvailability. Its dimensions are the SpaceDimension and TimeDimension. The measures are aggregated with *average*, *minimum*, and *maximum* operators along the TimeDimension and with *sum*, *average*, *minimum*, and *maximum* operators along the SpaceDimension (measures are semiadditive).

The static metrics (Table 1) are attributes of host members. Modeling these attributes is mandatory as they can be used to query the multidimensional model and to calculate derived measures like CPUAvailability.

With this fact table, the following are possible.

- Developers can check if one VO can support their applications, and administrators can check the distribution of the resource capacities and loads.
  Display total DiskFree for each VO on 08-2006 with highlighted minimums.
- Administrators and end users can find out at which periods the resources are underused and thus recommended to execute large tasks.
  Display total CPUAvailability for SITE1 for each weekday of 2006 with highlighted maximums?
- Administrators can evaluate the evolution of the architecture.
  Display the average MemFree on SITE1 for each week of the whole Grid life.

LinksMetrics
Links metrics contain the link-monitoring metrics measurements: latency and bandwidth.

Its dimensions are the TimeDimension, SourceDimension, and DestinationDimension. The latter two dimensions represent the same hierarchy as *SpaceDimension*. Exploiting this hierarchy allows querying the multidimensional model about links between different hosts and between groups of hosts (LANs and sites).

The measures are aggregated with *average*, *minimum*, and *maximum* along the TimeDimension and *average*, *minimum*, and *maximum* along the SpaceDimension.

With this fact table, the following are possible.

- One can compare the internal communication capability of the different LANs.
  Display average bandwidth inside each LAN on 08-2006.

- One can evaluate how the fickleness of the connection between two sites evolves in time.
  Display maximum and minimum latency between the SITE2 and the SITE1 for each month of 2006.
- One can select which LAN is the best to store a large data set used by a given host (if its own LAN does not have enough free disk space for instance).
  Display average bandwidth between Host21a and each LAN for 24-08-2006 with highlighted minimum and maximum.

Figure 6. HostsMetrics implementation

GR-OLAP Application

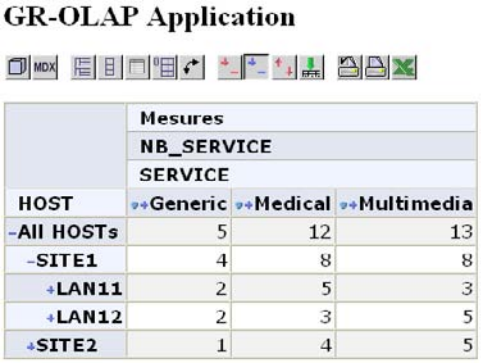
		Mesures			
TimeDimension	SpaceDimension	CPUFREE	MEMFREE	DISKFREE	CPUAVAILABILITY
-All TimeDimensions	+PHY	4.77	5,018.53	1,437,780.30	44,832.11
	+VO1	3.82	4,897.63	1,399,978.51	33,271.40
	+VO2	2.83	3,989.84	604,999.40	13,322.47
-06	+PHY	4.77	5,018.53	1,437,780.30	44,832.11
	+VO1	3.82	4,897.63	1,399,978.51	33,271.40
	+VO2	2.83	3,989.84	604,999.40	13,322.47
+08-06	+PHY	4.76	5,016.96	1,437,664.46	44,777.79
	-VO1	3.82	4,895.93	1,399,880.99	33,220.34
	-SITE1	2.88	4,774.88	1,362,115.70	23,002.64
	+LAN11	0.99	907.81	795,611.57	3,945.29
	+LAN12	1.89	3,867.08	566,504.13	7,556.03
	-vSITE1	0.94	121.05	37,765.29	660.17
	-vLAN21	0.94	121.05	37,765.29	660.17
	HOST21a	0.94	121.05	37,765.29	660.17
	+VO2	2.83	3,988.12	604,269.42	13,310.90
-09-06	+PHY	4.78	5,022.62	1,438,081.72	44,973.44
	+VO1	3.84	4,902.05	1,400,232.26	33,404.26
	+VO2	2.84	3,994.32	606,898.92	13,352.55
+01-09-06	+PHY	4.81	5,005.06	1,440,516.67	45,231.62
	+VO1	3.86	4,884.71	1,402,741.67	33,621.88
	+VO2	2.85	4,005.94	608,200.00	13,403.81
+02-09-06	+PHY	4.76	5,041.36	1,435,484.44	44,698.04
	+VO1	3.81	4,920.56	1,397,555.56	33,172.13
	+VO2	2.83	3,981.93	605,511.11	13,297.87

Figure 7. LinksMetrics implementation

GR-OLAP Application

		Mesures							
		AVG_LATENCY		AVG_BANDWIDTH		MIN_BANDWIDTH		MAX_LATENCY	
		GRIDD		GRIDD		GRIDD		GRIDD	
TimeDimension	GRIDS	+SITE1	+SITE2	+SITE1	+SITE2	+SITE1	+SITE2	+SITE1	+SITE2
-08-06	+SITE1	1.24	6.01	297.62	7.48	50.0	5.0	1.98	7.92
	-SITE2	5.99	1.48	7.47	74.53	5.0	50.0	7.92	1.98
	+LAN21	5.99	1.48	7.47	74.53	5.0	50.0	7.92	1.98
+25-08-06	+SITE1	1.17	6.13	331.83	7.26	52.0	5.0	1.98	7.6
	-SITE2	6.24	1.59	7.74	65.75	5.1	51.0	7.68	1.78
	+LAN21	6.24	1.59	7.74	65.75	5.1	51.0	7.68	1.78
+26-08-06	+SITE1	1.25	6.04	298.45	7.39	50.0	5.0	1.98	7.92
	-SITE2	6.02	1.54	7.45	74.79	5.0	50.0	7.92	1.98
	+LAN21	6.02	1.54	7.45	74.79	5.0	50.0	7.92	1.98
+27-08-06	+SITE1	1.25	6.03	298.83	7.40	50.0	5.0	1.98	7.92
	-SITE2	5.99	1.48	7.46	73.01	5.0	51.0	7.92	1.98
	+LAN21	5.99	1.48	7.46	73.01	5.0	51.0	7.92	1.98
+28-08-06	+SITE1	1.23	5.99	292.63	7.38	50.0	5.0	1.98	7.92
	-SITE2	5.93	1.46	7.37	74.53	5.0	50.0	7.92	1.96
	+LAN21	5.93	1.46	7.37	74.53	5.0	50.0	7.92	1.96
+29-08-06	+SITE1	1.24	6.04	296.69	7.66	50.0	5.0	1.98	7.92
	-SITE2	5.93	1.46	7.64	75.97	5.0	50.0	7.92	1.98
	+LAN21	5.93	1.46	7.64	75.97	5.0	50.0	7.92	1.98
+30-08-06	+SITE1	1.23	5.94	300.05	7.59	50.0	5.0	1.98	7.92
	-SITE2	6.07	1.47	7.42	74.73	5.0	50.0	7.92	1.98
	+LAN21	6.07	1.47	7.42	74.73	5.0	50.0	7.92	1.98

Figure 8. HostsServices implementation



HostServices

HostsServices represents the set of services executed on hosts. It is used either to select a set of hosts executing some services (or categories), or to find what services (and categories) are executed on a set of hosts.

Its dimensions are the SpaceDimension and ServicesDimension. This is a "fact-less" fact, thus the aggregation function is *COUNT*.

With this fact table, the following are possible:

- Developers can find out which sites have particular interest in medical purposes.
  Display the number of medical services executed on each site.
- Administrators can check the distribution of the interests of VOs.
  Display the number of services executed on VO1 for each category.

Navigation in Fact Tables

Furthermore, the main interest of our solution is to manipulate all the facts together. Each fact table covers one aspect of the multidimensional querying. As shown in Figure 5, HostsMetrics and LinksMetrics share common dimensions. The application of the drill-across operator and the common dimension members allow the relation of facts, for instance, as in the following:

- *Retrieve the number of multimedia services and the total CPUAvailability, MemFree, and DiskFree for each LAN and on 08-2006 with highlighted maximums.*
- *Retrieve the number of medical services and the total DiskFree for each LAN on 08-2006 with highlighted maximums.*
- *Retrieve the latency and bandwidth between the LANs of the precedent queries on 08-2006 with highlighted minimum and maximum.*

These three queries allow us to find out which are the most recommended LANs to participate in on a new VO where medical services will produce large data sets and send them to multimedia services for highly resource-consuming processing.

It is important to note that the presented queries are crucial for Grid users in their everyday tasks but impossible with current monitoring systems.

Implementation of the GR-OLAP DW

The GR-OLAP multidimensional application has been implemented using Oracle as a DBMS (database management system), Mondrian (*Mondrian*, n.d.) as an OLAP server, and JPivot (*JPivot*, n.d.) as an OLAP Web client.

To populate the DW, we have generated a sample of monitoring data based on real monitoring data gathered by NWS on our test Grid, which is composed of six highly heterogeneous hosts distributed over three distant sites. The data have been replicated with random changes to simulate a long-term monitoring.

Actually, populating our DW with real data from a real monitoring system is not an issue. Indeed, the GMA standard described previously is absolutely compatible with the ETL process. From a logical architecture point of view, several heterogeneous sources (called producers in the GMA) can be easily used thanks to the GMA's subscription mechanism: Our DW is a standard GMA consumer (Figure 2). From the interoperability point of view, the query languages of monitoring solutions are based on common

standards (SQL, XML, or LDAP): They are easy to integrate. As a matter of fact, our solution can be deployed with no more effort than any other DWs and OLAP solutions.

In the following paragraphs, we present three different screenshots of the JPivot pivot table showing multidimensional Grid monitoring analysis capabilities of our DW. The toolbar provides interactive triggering of OLAP operators (roll-up, slice, etc.).

Figure 6 shows a multidimensional query on the HostsMetrics fact. The user can navigate in the SpaceDimension using the physical structure of the Grid (PHY member) or the several VOs (i.e., *VO1*). The noncovering SpaceDimension has been implemented with placeholders (i.e., *vLAN21*) to eliminate skipped levels. Moreover, the nonstrictness at the VO level implies inaccuracy issues: multiple counting and repartition of measure values (Kimball & Ross, 2002). Fortunately, aggregations between different VOs are irrelevant and the measures associated with one host participate entirely with the aggregations at the VO level.

For instance, Figure 6 shows that the global computing capability of *VO1* is twice the one of *VO2*.

Figure 7 shows the result of a multidimensional query on the LinksMetrics fact: measures associated with connections between sites (*SITE1*, *SITE2*), and between sites (*SITE1*, *SITE2*) and LAN (*LAN21*) by month and day.

For instance, one can see that the network of *SITE1* is faster than the one of *SITE2*.

Finally in Figure 8 we show an example of a query on the HostsServices fact.

For instance, one can see that *SITE1* runs as many medical as multimedia services, but their distribution differs at the LAN level.

Evaluation of the Disk Space Needed by GR-OLAP DW

HostsMetrics and LinksMetrics are the only tables concerned with frequent insertions and thus are the only ones that may present size issues.

Let h be the number of monitored hosts, d the delay between two measures, and p the period of time concerned with the stored data.

If the length of the host identifier is 4 bytes (like IPv4 address), and CPUFree, MemoryFree, DiskFree, latency, and bandwidth are encoded in 4 bytes (like standard integer and float), then the cumulative size of the two tables is

$$s = \left[h \times sizeof(PointsMetrics) + h \times (h-1) \times sizeof(LinksMetrics)\right] \times \frac{p}{d}$$
$$= \left[h \times (4 \times 4) + h \times (h-1) \times (3 \times 4)\right] \times \frac{p}{d}$$

with h=1000 hosts, p=1 month, and d=5 minutes ($s \approx 100$GB).

The main issue is that the size of LinksMetrics grows in $O(h^2)$. Even without considering all the tables, the aggregations, and the specificities of DBMS, the size might be an issue on a year scale.

First, a frequency of 5 minutes is generally useless and should be limited to specific cases; hour-scale or day-scale frequencies are more realistic. Second, the lowest level data can be dropped after a given delay: hour-scale data of the past months are probably of no use anymore. So, this size can be kept reasonable without affecting the results' relevancy and accuracy.

Other performance aspects, like maintenance overhead, are not specific to our application. Thus, we can rely on the DW solution to handle them.

DISCUSSION AND FUTURE TRENDS

First, the main limitation of our solution is related to the dynamicity of dimensions. Actually, sites, LANs, and hosts can appear or disappear, can change regarding attributes or structure, and can join and leave VOs at any time. This dynamicity affects the schema of the space dimension and users are interested in the traces of these modifications. Unfortunately, on-the-fly modifications of dimensions are not easy to handle. The same issue applies to the ServiceDimension. This implies the

investigation of temporal modeling aspects in our future works.

Second, it would be interesting to include end users' monitoring aspect in our application. Who uses what and when? What is the typical user profile of some services? This means that measures are not numeric values but complex objects as described in Bimonte, Tchounikine, and Miquel (2005).

Third, interactive maps and graphic displays are the main instruments to support a real and effective spatiotemporal analysis process. They not only reveal spatiotemporal trends or relationships, but they also stimulate users' thinking processes. Indeed, a cartographic representation of the multidimensional data permits the visualization of measures on maps and the discovery of spatiotemporal correlations between facts and dimension members. For these reasons, a potential perspective is to navigate in the GR-OLAP hypercube using a spatial OLAP client like GéOLAP (Bimonte et al., 2005).

Fourth, the main limitation of our client is that the different pivot tables are not interconnected: Users cannot navigate between facts using the interface of JPivot only, but must use MDX or SQL. An extension of JPivot supporting navigations in all the fact tables and providing drill-across operators will be provided in the next version. Moreover, we plan to integrate classical grid monitoring tools like WebMDS (Czajkowski et al., 2001) in order to create a complete Grid monitoring decision support application.

Moreover, Grid monitoring data are real time. So, as ETL and querying in real-time DWs are open research issues nowadays, investigations of these aspects are promising research trends for our DW solution.

Finally, we will investigate how our application can be used in automatic decision making, for instance, in services deployment problems for autonomous and self-healing networks. Such problems are more complex than basic operational ones. Their solving cannot rely on the basic monitoring data provided by classical monitoring tools: They need complex views upon the state and life cycle of the resources

CONCLUSION

We have presented how OLAP technologies can be applied to Grid monitoring. This novel application increases the number and complexity of possible queries on monitoring information. We have shown that current monitoring solutions are too limited to satisfy Grid users in their everyday tasks. Using our application in addition clearly enhances the analysis capabilities on monitoring data. Whereas the conceptual model is rather complex and the current DW+OLAP solutions do not fully cover the peculiarities of our application, we have shown the feasibility and rationality of our solution by an implementation that is compatible with current monitoring architectures. Finally, this work opens several interesting perspectives for both scientific research purposes and actual Grid users' activity.

ACKNOWLEDGMENT

The authors would like to thank Anne Tchounikine, Maryvonne Miquel, and Lionel Brunie for their great help.

REFERENCES

Balaton, Z., Kacsuk, P., & Podhorszki, N. (2001). *Use cases and the proposed grid monitoring architecture* (Tech. Rep. No. LDS-1/2001). Hungary: Hungarian Academy of Sciences, Computer and Automation Research Institute.

Bimonte, S., Tchounikine, A., & Miquel, M. (2005). Towards a spatial multidimensional model. *Proceedings of the 8th International Workshop on Data Warehousing and OLAP* (pp. 39-46).

Cooke, A. W., Gray, A., & Nutt, W. (2004). The relational grid monitoring architecture: Mediating information about the grid. *Journal of Grid Computing, 2*(4), 323-339.

Czajkowski, S., Fitzgerald, K., & Foster, I. (2001). Grid information services for distributed resource sharing. *Proceedings of the 10th IEEE International Symposium on High-Performance Distributed Computing* (pp. 181-194).

Foster, I., & Kesselman, C. (Eds.). (1994). *The grid: Blueprint for a new computing infrastructure.* San Francisco: Morgan Kaufmann.

Foster, I., Kesselman, C., & Tuecke, S. (2001). The anatomy of the grid: Enabling scalable virtual organizations. *International Journal of Supercomputer Applications, 15*(3), 200-222.

Inmon, W. H. (1996). The DW and data mining. *Communications of ACM, 39*(11), 49-50.

JPivot. (n.d.). Retrieved July 10, 2006, from http//jpivot.sourceforge.net

Kimball, R., & Ross, M. (2002). *The data warehouse toolkit* (2nd ed.). New York: John Wiley and Sons, Inc.

Lowekamp, B., Tierney, B., & Cottrell, L. (2004). *A hierarchy of network performance characteristics for grid applications and services* (Global Grid Forum Proposed Recommendation No. GFD-R.023).

Malinowski, E., & Zimányi, E. (2006). Hierarchies in a multidimensional model: From conceptual modeling to logical representation. *Data and Knowledge Engineering, 59*(2), 348-377.

Mondrian. (n.d.). Retrieved July 10, 2006, from http://mondrian.pentaho.org

Truong, H., & Fahringer, T. (2004). SCALEA-G: A unified monitoring and performance analysis system for the grid. *European Across Grids Conference* (pp. 202-211).

Wolski, R., Spring, N. T., & Hayes, J. (1999). The network weather service: A distributed resource performance forecasting service for metacomputing. *Future Generation Computer Systems, 15*(5-6), 757-768.

KEY TERMS

Data Warehouse: A database application that collects, integrates, and stores data with the aim of producing accurate and timely management of information and support for analysis techniques.

Decision Support System: A computerized system for decision-making support through the estimation, the evaluation, and/or the comparison of alternatives.

The Grid: A large-scale, highly heterogeneous distributed system that supports scattered communities to form virtual organizations sharing resources.

Metric: A (monitoring) metric is a quantity related to the performance and reliability of the Internet.

Monitoring: The process of collecting information regarding the current status of shared resources.

Online Analytical Processing (OLAP): An approach to quickly provide the answer to analytical queries that are dimensional in nature.

Virtual Organization: A temporary network of companies, suppliers, customers, or employees linked by information and communications technologies with the purpose of solving a specific problem.

Chapter LXXVII
A Pagination Method for Indexes in Metric Databases

Ana Valeria Villegas
Universidad Nacional de San Luis, Argentina

Carina Mabel Ruano
Universidad Nacional de San Luis, Argentina

Norma Edith Herrera
Universidad Nacional de San Luis, Argentina

INTRODUCTION

Searching for database elements that are close or similar to a given query element is a problem that has a vast number of applications in many branches of computer science, and it is known as proximity search or similarity search. This type of search is a natural extension of the exact searching due to the fact that the databases have included the ability to store new data types such as images, sound, text, and so forth.

Similarity search in nontraditional databases can be formalized using the metric space model (Baeza Yates, 1997; Chavez, Navarro, Baeza-Yates, & Marroquín, 2001; Chávez, & Figueroa, 2004). A metric space is a pair (X, d) where X is a universe of objects and $d: X \times X \rightarrow R^+$ is a distance function that quantifies the similarities among the elements in X. This function d satisfies the properties required to be a distance function:

- $\forall\ x, y \in X,\ d(x,y) \geq 0$ (positiveness)
- $\forall\ x, y \in X,\ d(x,y) = d(y,x)$ (symmetry)
- $\forall\ x, y, z \in X,\ d(x,y) \leq d(x,z) + d(z,y)$ (triangle inequality)

A finite subset $U \subseteq X$, which will be called a database, is the set of objects where the search takes place.

A typical query in a metric database to retrieve similar objects is the range query, denoted by $(q, r)_d$. Given a query $q \in X$ and a tolerance radius r, a range query retrieves all elements from the database that are within a distance r to q; that is $(q, r)_d = \{ x / x \in U \wedge d(x, q) \leq r \}$.

Range queries can be trivially answered by examining the entire database U. Unfortunately, this is generally very costly in real applications. In order to avoid this situation, the database is preprocessed by using an indexing algorithm. An indexing algorithm is an off-line procedure whose

aim is to build a data structure or index designed to save distance computations at query time.

The metric spaces indexing algorithms are based on, first, partitioning the space into equivalence classes and, second, a subsequent indexation of each class. Afterward, at query time, some of these classes can be discarded using the index and an exhaustive search takes place only on the remaining classes. The difference among the existing indexing algorithms is the way they build up the equivalence relation. Basically, two groups can be distinguished: pivot-based algorithms and local partitioning algorithms.

Pivot-Based Algorithms

The idea behind the pivot-based algorithms is as follows. At indexing time, k pivots $\{p_1, p_2,..., p_k\}$ are selected from X and each database element u is mapped to a k-dimensional vector, which represents the respective distances to the pivots, that is, $\phi(u) = (d(u,p_1), d(u,p_2) ,..., d(u,p_k))$. When a range query $(q, r)_d$ is issued, the triangle inequality is used together with the pivots in order to filter out elements in the database, without actually evaluating each distance to the query q. To do this, $\phi(q) = (d(q,p_1), d(q,p_2) ,..., d(q,p_k))$ is computed; if $|d(q,p_i)-d(u,p_i)| > r$ holds for any pivot p_i, then, by the triangle inequality, we know that $d(q,u)>r$ without actually evaluating $d(q,u)$. Only those elements that cannot be discarded using this rule are actually compared against q.

Local Partition Algorithms

In this case, the idea is to divide the database in zones as compact as possible. A set of centers $\{c_1,c_2, \ldots,c_k\}$ is chosen and each element in the database is associated with its closest center c_i. The set of the elements closer to the center c_i than to any other center forms the class $[c_i]$. There are many possible criteria to discard classes when the index is used to answer a query; the most popular ones are the following:

1. **Hyperplane criterion:** This is the most basic one and the one that best expresses the idea of compact partition. Basically, if c is the center of the class $[q]$ (i.e., the center closest to q), then the query ball does not intersect $[c_i]$ if $d(q,c)+r < d(q,c_i)-r$. In other words, if the query ball does not intersect the hyperplane that divides both its closest center c and c_i, then the ball is totally outside the class of c_i.
2. **Covering radius criterion:** This tries to bound the class $[c_i]$ by considering a ball centered at c_i that contains all the elements of U that lie in that class. The covering radius of c for database U is $cr(c) = max_{u\in[c]\cap U} d(c,u)$. Therefore, we can discard $[c_i]$ if $d(q,c_i)-r > cr(c_i)$.

Regardless of the indexing algorithm used, the total time to evaluate a query (similarity search) can be split as T = *# of distances evaluations* x *complexity (d)* + *CPU extra time* + *I/O time*, and we would like to minimize T. Based on the assumption that evaluating d is so costly, most of the research on metric spaces has focused on algorithms to discard elements reducing the number of distance evaluations, paying no attention to I/O cost; one exception is the M tree designed specifically for secondary storage. However, if the index does not fit into main memory, the I/O cost plays an important role and, consequently, the performance criteria will be both the number of distance evaluations and the number of disk page reads (I/O cost).

We begin presenting a brief explanation of indexes on secondary storage. After that, we introduce our proposal in detail and give an application example. Finally, the conclusions are presented.

INDEXES IN SECONDARY STORAGE

The organization of the memory of the most computer systems is based on a hierarchy of memory

levels. In this memory hierarchy, each level has its own cost and performance characteristics. Accessing the data in the lowest levels (registers, cache, and main memory) is much faster than accessing them in the highest levels (secondary storage).

The secondary storage is assumed to be partitioned into transfer blocks, called disk pages, each of which contains *p* atomic items. We call *p* the disk page size and we use the term disk access or I/O operation for a disk page reading or writing operation (Vitter, 2001).

For those applications processing large data sets on secondary storage, the number of disk accesses required at query time is an important factor that directly influences the overall performance. For this reason, one of the most important aspects with respect to the index design is whether the main memory has space enough to contain the whole index. A typical example is the hashing data structure; if it fit into main memory, the number of slots per bucket must be 1 or else the bucket size is determined by the disk page size, that is, as many slots as can contain in a page. Another example is the search tree; when it is used in the main memory, it achieves a good performance with arity 2 (Ramez & Shamkant, 2002; Ullman, 1988;). However, when the search tree is kept in secondary storage, a disk page is used for storing each node of the tree, increasing the arity as much as necessary. The widely known B tree (Comer, 1979) has established the basis for the development of a great number of efficient indexes in secondary storage, such as B^+ tree for relational databases (Ullman, 1988), R tree for spatial databases (Guttman, 1984), string B tree for text databases (Ferragina and Grossi, 1999), and M tree for metric databases (Ciaccia, Patella, & Zezula, 1997), among others.

In general, it can be seen that most proposals for secondary storage indexes consist of adapting the base unit of a data structure (a bucket in hashing or a node in a tree) to a disk page.

We propose a strategy for metric databases whose index and/or data do not fit into main memory. In order to minimize the number of disk accesses, instead of modifying the index, we adapt the metric database, partitioning it according to the main memory sizes such that the index of each part fits into main memory. Then, a search is carried out by separately seeking in each index, which may be done both in the main memory and in parallel. In addition, because this technique does not imply that there is no modifying in the index structure, it can be used with any metric index that has a good performance in main memory.

HANDLING METRIC INDEXES IN SECONDARY STORAGE

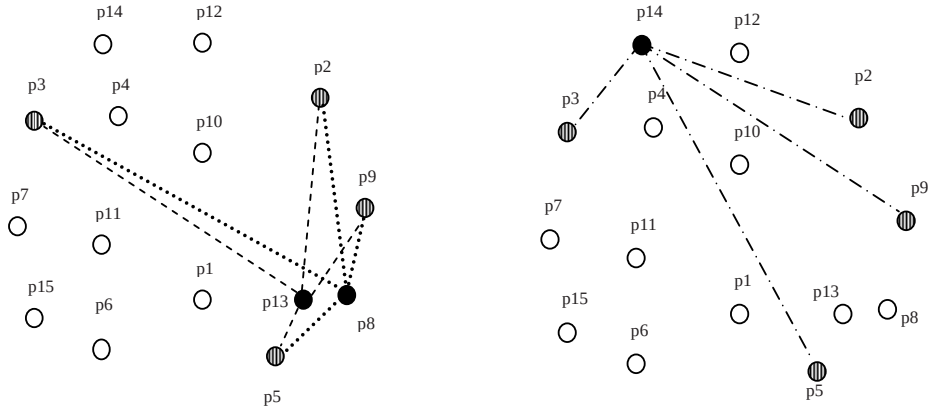
As mentioned above, this strategy partitions the metric database in such way that the index of each part fits into the main memory; that is, each part is indexed separately from the rest. A proximity query is carried out as follows:

1. the index of the part is loaded
2. the index is searched to determine which classes may be relevant to the query
3. if there are relevant classes, the corresponding part of the metric database is loaded into main memory and the elements belonging to this class are directly compared against the query
4. The steps 1 to 3 are repeated for each part.

Since searching in each of the parts is independent of the others, so can it be done in parallel by distributing the index of the parts on different nodes in a network.

The database can be trivially partitioned at random. However, this would not be a good choice since it is highly probable that the query has an answer element in each part, then all parts of the database must be loaded in main memory to answer it. In order to avoid this situation, we divide the database by grouping similar elements. Thus, we reduce the amount of parts loaded in the main memory. That is, in each part either there is a large percentage of elements similar to the query, or the whole part should be discarded from the index search, which can greatly reduce the I/O accesses performed at query time.

Figure 1. Permutations generated by the permuting elements p5, p9, p2, p3. For p13 and p8 the permutation is p5,p9,p2,p3. For p14 the permutation is p3,p2,p9,p5.



The main difficulty in this process lies in how to decide whether two elements are similar or not. To solve this problem, we have designed the partitioned LCS technique, denoted by PLCS, that allows us to split the database by using the LCS (longest common subsequence) distance. The similarity among the elements grouped in a part is given by the closeness of such elements to a preselected set of objects.

PLCS Algorithm

This process uses the LCS distance function to determine the similarity among the elements. Given two strings, this function returns the length of the longest common subsequence between them. For example, given the alphabet $\sum$={*a, b, c, d, e, f, g, h, i*} and two strings α=*aabcdeh* and β=*abfgdih*, then *LCS*(α, β)=4 where *abdh* is the longest common subsequence.

This process consists of identifying two groups: *t_acceptable* (the set of elements that are similar according to a tolerance *t*) and *t_non-acceptable*. The *t_acceptable* group is partitioned at random since their elements are similar. Then, this process is applied to *t_non-acceptable*.

An outline of PLCS algorithm follows.

1. *z* elements (called permuting elements) are randomly selected from the database.
2. For each element *x* in the database (including the permuting ones), a vector v_x (called permutation) of *z* components is generated. The permuting elements are kept ordered according to the distance to *x*. That is, if v_{x_i} is an element found in the i^{th} position of v_x, then $d(x,v_{x_i}) \leq d(x, v_{x_{(i+1)}})$ for $i = 1 \ldots z\text{-}1$.
3. A permutation is randomly chosen as the canonical one (denoted by v_c).
4. For each element *x*, if $LCS(v_x, v_c) \geq t$, then *x* is added to the set of the *t_acceptable;* otherwise, it becomes part of the *t_non-acceptable* set.
5. The *t_acceptable* set is partitioned at random. The number of elements in each part is determined by the number of elements that fit into a single disk page.
6. The elements of the *t_acceptable* set that found no location (Step 5) together with the *t_non-acceptable* ones form a new group. This new group is submitted to the above steps again by choosing new *z* permuting elements different from those already used.

This process determines the similarity between two elements with respect to the permuting ones; that is, two elements are considered to be similar

if they see *t* or more permuting elements in the same order. Figure 1 illustrates these ideas. In this example, four permuting elements have been chosen, p_3, p_2, p_9, and p_5, and the permutations for the points p_{13}, p_8, and p_{14} are computed. Notice that both p_{13} and p_8 generate the same permutation $v_{p14} = v_{p8} = (p_5, p_9, p_2, p_3)$, and consequently $LCS(v_{p13}, v_{p5}) = 4$. Nevertheless, if we look at point p_{14}, its permutation is $v_{p14} = (p_3,p_2,p_9, p_5)$ and, therefore, $LCS(v_{p13}, v_{p14}) = 1$, meaning that p_{13} is not similar to p_{14}.

It is possible that, at some time, a set of very different elements remains and it cannot be classified with the *z* permuting elements; in other words, the *t_acceptable* set becomes empty or has so few elements that new partitions cannot be generated. This group of elements, called outliers, is partitioned at random.

Notice that this algorithm has three parameters: the number of parts that are generated with each set of *t_acceptable*, the tolerance *t*, and the amount of permuting elements *z*. These parameters must be specified when PLCS is used; therefore, a preprocessing of the database is needed in order to determine the most appropriate values for them.

AN APPLICATION EXAMPLE

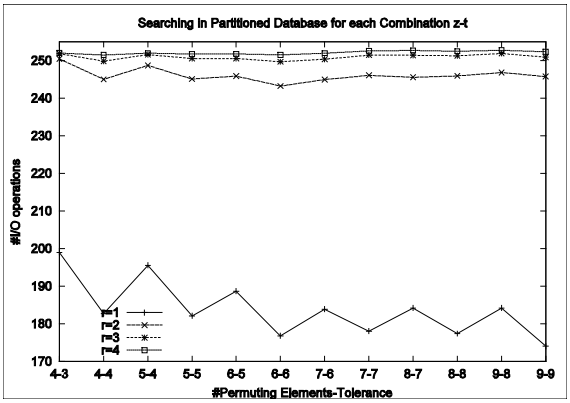
In the following we will show the application of the PLCS technique to a particular metric database in order to determine the efficiency of this method when solving similarity searches.

The experiments were performed over a Spanish dictionary using the edit distance (also called Leveshtein distance) as the distance function. This function is discrete and computes the minimum number of character insertions, deletions, and replacements needed to make two strings equal. Each part generated by PLCS will be indexed with a pivot-based structure (Chavez & Figueroa, 2004).

As a first step, we aimed to determine the most appropriate values for the algorithm parameters. To achieve this objective, we make a previous experimentation using a batch of 500 queries randomly taken from the database according to the following premises.

- As regards the number of parts generated with each set of *t_acceptable*, we will consider two options: One is to generate the greatest possible number of parts, and the other is to generate only two parts in order to allow the elements to form a group considering different reference points.

Figure 2. I/O operations for PLCS strategy when the amount of parts that can be generated with t_acceptable group is not limited. On the left part are I/O operations of each combination z-t for a range search with radius r= 1, 2, 3 and 4. On the right is the number of parts produced from the set of outliers.



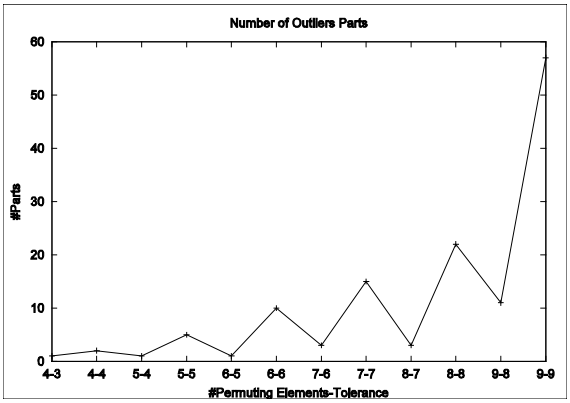


Figure 3. I/O operations for PLCS strategy when the amount of parts that can be generated with t_acceptable group is limited

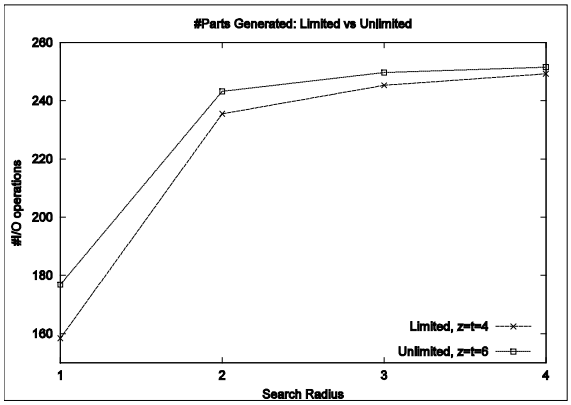


Figure 4. I/O operations for PLCS strategy when the amount of parts that can be generated with t_acceptable group is limited vs. I/O operations when the amount of parts that can be generated is not limited

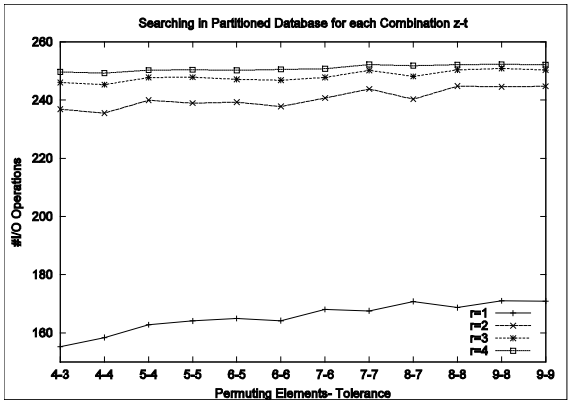


Figure 5. I/O operations when partitioning the database with PLCS strategy vs. I/O operations when partitioning does not take place

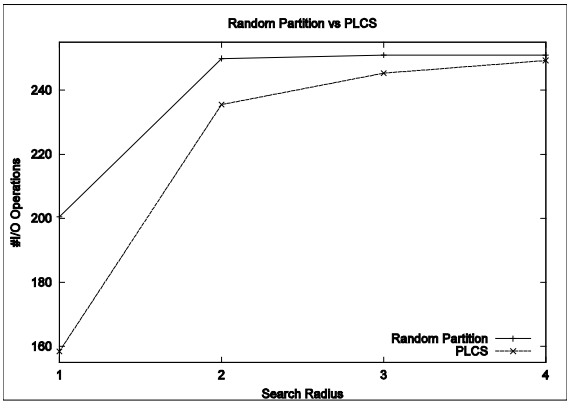
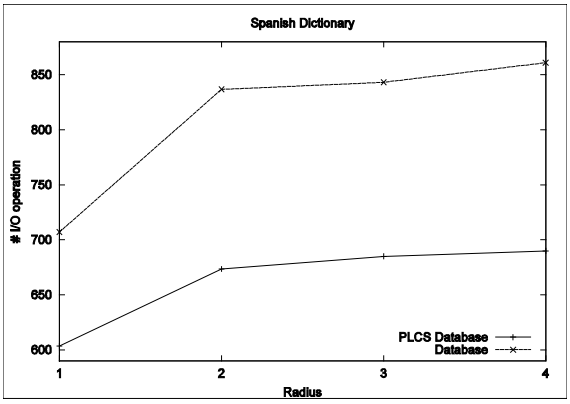


Figure 6. I/O operations when partitioning the database with PLCS strategy vs. I/O operations when the database is randomly partitioned



- With respect to *t* and *z*, to choose the value of *t* close to that of *z* is more representative than to choose its values independently. For example, choosing t=3 does not mean the same when *z*=4 as when *z*=9. In the first case, for two elements to be similar, we are requiring them to see three out of four permuting elements in the same order. In the second case, we require three out of nine, implying that two nonsimilar elements belong to the same part. Therefore, we will prove what happens when *t* varies depending on the value of *z*, that is, *z*=4, 5, …, 9 and *t*= *z*, *z*-1.
- Finally, we have to establish an upper bound in the number of attempts to partitioning a group before it is considered a set of outliers. By means of experimentation we have been able to determine that 500 iterations is a safe limit.

Figure 2 shows the results obtained for the different values of *z* and *t* when the number of parts generated with each *t_acceptable* set is unlimited. The left part of the figure shows the number of I/O operations of each combination *z-t* for a range search with radius r= 1, 2, 3, and 4. Since a best combination for all the radius values considered does not exist, we chose the closest combination to the optimal one in all the cases (*z*=6 and *t*=6) for future comparisons.

Note that if we compared *t=z* and *t=z-1* for each *z* value, it always occurs that the most competitive results are obtained in *t=z*. This behavior is due to the elements grouped with *t=z* being really similar because they see all the permuting elements in the same order (recall the example displayed in Figure 1 in the previous section). Furthermore, as *z* increases, the performance of the technique improves (while more permuting elements are used, more reference points exist to corroborate whether two elements are really similar).

However, it is important to note that increase in the z value produces an increment in the cardinality of the outlier set generated. Figure 2 (on the right) shows the number of parts produced from the set of outliers, where it can be clearly seen how *z* influences in this aspect.

Now, we consider the results when the amount of parts generated with each set of *t_acceptable* is 2. From these results, we can conclude that the most advisable combination is *z=t*=4 (see Figure 3).

Note that, in this case, the values for *z* and *t* where a better performance is achieved are lower than the values in the previous one (i.e., not limiting the amount of parts). This is due to the fact that limiting the number of parts to be generated causes a few parts to depend on the same reference points (the permuting ones), preventing the classification to be biased by the first *z* permuting. This same behavior was obtained in the previous case by increasing the LCS distance to its maximum value (*t=z*), thus getting each set of *t_acceptable* to have a few elements and, consequently, to generate few parts**.**

Based on the above we can conclude the following.

- When the amount of parts generated with each *t_acceptable* set is unlimited, the best combination is *z=t=6*.
- When the amount of parts generated with each *t_acceptable* set is limited to 2, the best combination is *z=t=4*.

Figure 4 shows the behavior of these two points for the Spanish dictionary as the search radii increases. It can be inferred that the best option is to limit the amount of parts generated with each *t_acceptable* set to 2, fixing *z=4* and *t=4*.

It is important to remark that the tuning of the parameter values is necessary with each class of metric database on which PLCS will be applied. Although this process implies more effort at indexing time, it achieves a reduction in the I/O operations at query time. This can be observed in Figure 5, which shows the amount of I/O operations when partitioning the database with PLCS vs. the amount of I/O operations when partitioning does not take place, that is, leaving the process of index partitioning to the operating system. The results shown are the average of 500 range query with radii 1 to 4.

Finally, it is important to note that the efficiency of PLCS lies on the appropriate division of the space. Figure 6 shows the number of disk accesses made by PLCS vs. the number of disk accesses made when the database is randomly partitioned.

CONCLUSION

In this work, a strategy for metric databases whose index and/or data do not fit into main memory has been introduced. To do this, instead of modifying the index, we adapt the metric database, partitioning it according to the main memory size such that the index of each part fit into main memory. Then, a search is carried out by separately seeking in each index, which may be done both in the main memory.

This strategy has two main advantages.

- Since searching in each of the parts is independent of the others, it can be done in parallel by distributing the index of the parts on different nodes in a network.
- Because this technique does not imply that there will be no modifying in the index structure, it can be used with any metric index that has a good performance in the main memory.

We have designed the partitioned LCS technique that allows us to split the database by using the longest common subsequence distance to determine the similarity among the elements. This process consists of identifying two groups: *t_acceptable* (the set of elements that are similar according to a tolerance *t*) and *t_non-acceptable*. The *t_acceptable* group is partitioned at random since their elements are similar. Then, this process is applied to *t_non-acceptable*.

The experiments to evaluate the partitioned LCS technique were performed over word dictionaries using the edit distance. Hence, we conclude the following.

- For the Spanish dictionary, the most appropriate values for the algorithm parameters are
  - the number of parts that are generated with each set of *t_acceptable*=2,
  - the tolerance *t*=4,
  - and the amount of permuting elements *z*=4.
- The LCS technique reduced the amount of disk accesses made by 21% with respect to when the database is randomly partitioned.
- The LCS technique reduced the amount of disk accesses made with respect to when partitioning does not take place, that is, leaving the process of index partitioning to the operating system.

REFERENCES

Baeza Yates, R. (1997). Searching: An algorithm tour. In A. Kent & Williams (Eds.), *Encyclopedia of computer science* (Vol. 37, pp. 331-359). Marcel Dekker Inc.

Chavez, E., Navarro, G., Baeza-Yates, R., & Marroquín, J. (2001). Searching in metric spaces. *ACM Computing Surveys, 33*(3), 273-321.

Chavez, E., & Figueroa, K. (2004). Faster proximity searching in metric data. In *Lecture notes in computer science: Vol. 2972. Proceedings of MICAI 2004.* México: Springer.

Ciaccia, P., Patella, M., & Zezula, P. (1997). M-tree: An efficient access method for similarity search in metric spaces. *Proceedings of the 23rd Conference on Very Large Databases* (pp. 426-435).

Comer, D. (1979). The ubiquitous B-tree. *ACM Computing Surveys, 11*(2), 121-137.

Ferragina, P., & Grossi, R. (1999). The string B-tree: A new data structure for string search in external memory and its applications. *Journal of the ACM, 46*(2), 236-280.

Guttman, A. (1984). R-trees: A dynamic index structure for spatial searching. *Proceedings of the ACM SIGMOD International Conference on Management of Data* (pp. 47-57).

Ramez, E., & Shamkant, N. (2002). *Fundamentos de sistemas de bases de datos.* Addison Wesley.

Ullman, J. (1988). *Principles of database and knowledge: Base systems* (Vol. 1). New York: Computer Science Press, Inc.

Vitter, J. (2001). External memory algorithms and data structure: Dealing with massive data. *ACM Computing Surveys, 33*(2), 209-271.

KEY TERMS

Disk Access: A disk page reading or writing operation.

Disk Page: A fixed-size portion of secondary storage corresponding to the amount of data transferred in a single access.

Indexing Algorithm: A procedure to build beforehand a data structure or index designed to speed up searches.

I/O Cost: Number of disk page reads or writes.

LCS Distance: A distance function that applies to chains and returns the length of the common maximum subsequence between two chains.

Metric Database: A finite subset of a metric space.

Metric Space: A universe of objects together with a distance function that measures the similarities between the elements in the universe.

Pagination Method: Method to store a data structure on disk pages in order to control the number of disk accesses to the secondary storage during search.

Similarity Query: Searching for database elements that are close or similar to a given query object.

Chapter LXXVIII
SWIFT: A Distributed Real Time Commit Protocol

Udai Shanker
M.M.M. Eng. College, India

Manoj Misra
IIT Roorkee, India

Anil K. Sarje
IIT Roorkee, India

INTRODUCTION

Many applications such as military tracking, medical monitoring, stock arbitrage system, network management, aircraft control, factory automation, and so forth that depend heavily on database technology for the proper storage and retrieval of data located at different remote sites have certain timing constraints associated with them. Such applications introduce the need for distributed real-time database systems (DRTDBS) [Ramamritham, 1993]. The implementation of DRTDBS is difficult due to the conflicting requirements of maintaining data consistency and meeting distributed transaction's deadlines. The difficulty comes from the unpredictability of the transactions' response times [Huang, 1991]. Due to the distributed nature of the transactions and in presence of other sources of unpredictability such as data access conflicts, uneven distribution of transactions over the sites, variable local CPU scheduling time, communication delay, failure of coordinator and cohort's sites, and so forth, it is not easy to meet the deadline of all transactions in DRTDBS [Kao & Garcia – Monila, 1995]. The unpredictability in the commitment phase makes it more serious because the blocking time of the waiting cohorts due to execute-commit conflict may become longer. Hence, due to unique characteristics of the committing transactions and unpredictability in the commitment process, design of an efficient commit protocol is an important issue that affects the performance of DRTDBS [Shanker, Misra & Sarje, 2006d].

BACKGROUND

The Two Phase Commit (2PC) is still one of the most commonly used protocols in the study of DRTDBS. Most of the existing commit protocols proposed in the literature, such as presumed commit (PC) and presumed abort (PA) [Haritsa, Ramamritham & Gupta, 2000] are based on it. Soparkar et al. [Nandit, Levy. Korth & Silberschatz, 1994] proposed a protocol that allows individual sites to unilaterally commit. If it is later found that the decision is not consistent globally then compensation transactions are executed to rectify errors. The problem with this approach is that many actions are irreversible in nature. The 2PC based optimistic commit protocol (OPT) [Gupta, Haritsa & Ramamritham, 1997] for real-time databases try to improve system concurrency by allowing executing transactions to borrow data from the transactions in their commit stage. This creates dependencies among transactions. If a transaction depends on other transactions, it is not allowed to start commit processing and is blocked until the transactions, on which it depends, have committed. The blocked committing transaction may include a chain of dependencies as other executing transactions may have data conflicts with it. Enhancement has been made in the Permits Reading of Modified Prepared-Data for Timeliness (PROMPT) commit protocol, which allows executing transactions to borrow data in a controlled manner only from the healthy transactions in their commit phase [Haritsa, Ramamritham & Gupta, 2000]. However, it does not consider the type of dependencies between two transactions. The abort of a lending transaction aborts all the transactions dependent on it. The impact of buffer space and admission control is also not studied. In case of sequential transaction execution model, the borrower is blocked for sending the workdone message and the next cohort can not be activated at other site for its execution. It will be held up till the lender completes. If its sibling is activated at another site anyway, the cohort at this new site will not get the result of previous site because previous cohort has been blocked for sending of workdone message due to being borrower [Shanker, Misra, Sarje & Shisondia, 2006c]. In shadow PROMPT, a cohort forks of a replica of the transaction without considering the type of dependency, called a shadow, whenever it borrows a data page.

The deadline-driven conflict resolution (DDCR) commit protocol maintains three copies of each modified data item (before, after and further) for resolving execute-commit conflicts [Lam, Pang, Son & Cao, 1999]. This not only creates additional workload on the system but also has priority inversion problems. Based on the concepts of above protocols [Lam, Pang, Son & Cao, 1999; Haritsa, Ramamritham & Gupta, 2000], Biao Qin and Y. Liu proposed a protocol Double Space (2SC) [Qin & Liu, 2003] which classifies the dependencies between lender and borrower into two types; commit and abort. The abort of a lending transaction only forces transactions in its abort dependency set to abort. The transactions in the commit dependency set of the aborted lending transaction continue as normal. However, 2SC creates inconsistency in case of write-write conflicts [Shanker, 2006e]. The protocols [Lam, Pang, Son & Cao, 1999; Qin & Liu, 2003] use blind write model whereas PROMPT uses update model.

SWIFT

A static two phase locking [Lam, Hung & Son, 1997; Lam, 1994] with higher priority (S2PL-HP) based distributed real time commit protocol named as SWIFT has been proposed.

Basic Idea of Protocol

A commit protocol can improve transaction success percentage by reducing the commit duration for each transaction, causing locks to be released

sooner resulting in reduction of the wait time of other transactions, or allowing ordered sharing of the locks.

The ideas discussed below are based on the factors listed above.

(a) The execution of a transaction may be delayed due to resources (CPU and disk) or data contentions. The optimization proposed in this work is based on the following observations:

1. The data contention is the main cause of delay in a transaction's execution.
2. A cohort may wait due to data contention only during its locking activity, i.e., in locking phase.

We, therefore, propose to divide the execution phase of the transaction into two parts:

1. Locking phase, and
2. Processing phase

The execution of a cohort is carried out according to the following steps:

1. During the locking phase, the transaction locks the data items.
2. Just before the start of processing phase, the cohort sends a Workstarted message to its coordinator. Then, it is executed.
3. After the receipt of WORKSTARTED messages from all its cohorts, the coordinator sends VOTE REQ message to all its cohorts at time t calculated as follows:

$$t = \max \{t_i + processing\_time_i\} - T_{com}$$

where,

t_i = Arrival time of WORKSTARTED message from $cohort_i$ $processing\_time_i$ = Processing time needed by the $cohort_i$

T_{com} = Communication Delay from one site to another

(b) If cohort T_i utilizes dirty data items already locked by other cohorts, one of the following three types of dependencies may arise.

1. T_i may be commit dependent on other cohorts.
2. T_i may be abort dependent on other cohorts.
3. T_i may be both commit dependent as well as abort dependent on other cohorts.

Let us consider the case where transaction T_2 (borrower) acquires all locks and enters its processing phase before its lender transaction T_1 receives the global decision. If T_2 completes the processing before T_1 receives the global decision, existing commit protocols, including 2SC and PROMPT, block the borrower from sending WORKDONE message until the lender commits or aborts. We propose to allow a commit dependent borrower to send WORKSTARTED message as abort of the lender never aborts the borrower. Hence, one of the following two decisions is taken based on the type of dependencies when the cohort is about to enter its processing phase after getting all the required locks.

1. T_2 is allowed to send WORKSTARTED message to its coordinator if it is only commit dependent on other cohorts. This is free from cascaded aborts because abort of T_1 (lender) does not cause T_2 (borrower) to abort.
2. T_2 is not allowed to send WORKSTARTED message to its coordinator if it is abort dependent on other cohorts. So, the coordinator cannot initiate commit processing. Instead, it has to wait until either T_1 receives its global decisions or its own deadline expires, whichever occurs earlier.

(c) The cohort sends a Yes Vote message in response to its coordinator's VOTE REQ message only when its dependencies are removed and it has finished its processing. If it is still dependent on any cohort or has not finished its processing, Yes Vote message is deferred. The borrower sends the deferred Yes Vote message after the completion of processing and removal of the dependencies. This may be either due to abort or commit of the lender.

(d) The CPU scheduling algorithm for the cohorts is described below. Cohort processing is done on the basis of CPU availability and cohort's priority.

1. If two cohorts are ready to run on the same processor, the higher priority cohort is scheduled first.
2. While in locking period, if a higher priority cohort (T_h) arrives, the lower priority cohort (T_L) aborts. T_L releases all its locked data items.
3. If a higher priority cohort (T_h) arrives during the processing phase of T_L it aborts the lower priority transaction (T_L). The aborted cohort/transaction releases all its locked data items.

SWIFT is based on redefined dependencies [Shanker, 2006e] that are created when a lock holding cohort lends its locked data to some other cohorts for reading or updating. The WORKSTARTED message is sent just before the start of processing phase of the cohort in place of sending Workdone message at the end of processing phase [Shanker et al. 2005b]. This improves performance of the system by overlapping the transmission time of WORKSTARTED message with the processing time of the cohorts. SWIFT also reduces the time needed for commit processing by allowing commit dependent only borrower to send its WORKSTARTED message instead of being blocked [Shanker et al. 2005a]. To ensure non-violation of the ACID properties, checking of completion of processing and the removal of dependency of cohort are done before sending the YES Vote message in SWIFT [Shanker et al. 2006a; Shanker et al. 2006b]. The important point of SWIFT is that the required modifications are local to each site and do not require inter-site communications so free from message overhead.

The performances of SWIFT has been compared with the 2SC and PROMPT for the scenario where communication delays are negligible and when they are large by simulating a distributed real time database system consisting of n sites both for the main memory resident and the disk resident databases. Results of simulation show a performance improvement of the order of 5% - 10% in Transaction Miss Percentage. The performance of SWIFT has also been analyzed for partial read-only optimization, which minimizes intersite message traffic, execute-commit conflicts and log writes consequently resulting in a better response time. The effect of partial read only optimization has been studied both for the main memory and the disk resident databases at communication delay of 0ms and 100ms. The performance improvement in transaction Miss Percentage varies from 1% to 5%. The impact of permitting the communication between the cohorts of the same transaction (sibling) in SWIFT has also been analyzed both for the main memory and the disk resident database at communication delay of 0ms as well as 100 ms. The cohort sends the abort messages directly to its siblings as well as its coordinator. A little improvement in transaction Miss Percentage was observed, i.e., up to 3%. A study has also been done to see the percentage of transactions that miss their deadline during processing phase in comparison to the total transaction miss percentage. The results show that this protocol is beneficial in case of main memory resident database with sufficient communication delay.

FUTURE TRENDS

Following are some suggestions to extend this work [Shanker, Misra & Sarje, 2001; Shanker, Misra & Sarje, 2007].

- More work is needed to explore the impact of communication among cohorts of the same transaction (siblings) on the overall system performance.
- Our performance studies are based on the assumption that there is no replication. Hence, a study of relative performance of the topic discussed here deserves a further look under assumption of replicated data.
- The integration and the performance evaluation of proposed commit protocol with 1PC and 3PC protocols.
- Although tremendous research efforts have been reported in the hard real time systems in dealing with hard real time constraints, very little work has been reported in hard real time database systems. So, the performance of SWIFT can be evaluated for hard real time constrained transactions.

CONCLUSION

In a large network, communication and queuing delays become a bottleneck [Ulusoy, 1992]. In SWIFT, the cohorts send WORKSTARTED message just before the start of their processing in place of sending WORKDONE message. This overlaps the message transmission time with the cohort's processing time and reduces the overall transaction's completion time. The borrower is also allowed to send WORKSTARTED message if the dependency between the borrower and its lenders is commit dependency only. It is free from cascaded aborts and borrower with only commit dependency is not aborted in case its lenders abort. This reduces the blocking period of the borrower. The simulation results show that the gain in performance can be achieved at low and moderate loads. A suitable modification in distributed real time commit protocol at the time of sending Yes Vote message has been made to ensure atomicity. The important point to note here is that the new protocol's features are local to each site and do not require inter-site communications. Also, the proposed optimization can be integrated easily with any other protocol based on 2PC.

REFERENCES

Gupta, R., Haritsa, J. R., & Ramamritham, K. (1997). More optimism about real-time distributed commit processing (Technical Report TR – 97 – 04). *Database System Lab, Supercomputer Education and Research Centre, I.I.Sc. Bangalore, India.*

Haritsa, J. R., Ramamritham, K., & Gupta, R. (2000). The PROMPT real-time commit protocol. *IEEE Transactions on Parallel and Distributed Systems, 11*(2), 160-181.

Huang, J. (1991). Real time transaction processing: Design, implementation and performance evaluation. *PhD thesis, University of Massachusetts.*

Kao, B., & Garcia – Monila, H. (1995). An overview of real-time database systems. *Advances in Real-Time Systems,* 463-486.

Lam, K. Y., Hung, S. L., & Son, S. H. (1997). On using real-time static locking protocols for distributed real-time databases. *Real-Time Systems, 13,* 141-166.

Lam, K. Y. (1994). Concurrency control in distributed real-time database systems. *PhD Thesis, City University of Hong Kong, Hong Kong.*

Lam, K. Y., Pang, C., Son, S. H., & Cao, J. (1999). Resolving executing committing conflicts in distributed real-time database systems. *Journals of Computer, 42*(8), 674-692.

Nandit, S., Levy, E., Korth, H. F., & Silberschatz, A. (1994). Adaptive commitment for real-time distributed transaction. *3rd International Conference on Information and Knowledge Management,* 187-194. *Gaithersburg, MD.*

Qin, B., & Liu, Y. (2003). High performance distributed real-time commit protocol. *Journal of Systems and Software, 68*(2), 145-152. *Elsevier Scie*nce Inc

Ramamritham, K. (1993). Real-time databases. Distributed and parallel databases. *Special Issue: Research Topics in Distributed and Parallel Databases, 1*(2), 199-226.

Shanker, U. (2006e). Some performance issues in distributed real-time batabase systems. *PhD Thesis, Department of Electronics & Computer Engineering, Indian Institute of Technology Roorkee, India.*

Shanker, U., Misra, M., & Sarje, A. K. (2001). Hard real-time distributed database systems: Future directions. *All India Seminar on Recent Trends in Computer Communication Networks, Department of Electronics & Computer Engineering,* 172-177. *Indian Institute of Technology Roorkee, India.*

Shanker, U., Misra, M., & Sarje, A. K. (2005a). Dependency sensitive distributed commit protocol. *8th International Conference on Information Technology (CIT05),* 41-46, *Bhubaneswar, India.*

Shanker, U., Misra, M., & Sarje, A. K. (2005b). Optimizing distributed real-time transaction processing during execution phase. *3rd International Conference on Computer Application (ICCA2005), University of Computer Studies,* 1-7. *Yangon, Myanmar.*

Shanker, U., Misra, M., & Sarje, A. K. (2006a). SWIFT-A new real time commit protocol. *International Journal of Distributed and Parallel Databases, 20*(1), 29-56. Springer Verlag

Shanker, U., Misra, M., & Sarje, A. K. (2006b). OCP-the optimistic commit protocol. *17th Australasian Database Conference (ADC2006), 49,* 193-202. *Hobart, Tasmania, Australia.*

Shanker, U., Misra, M., & Sarje, A. K. (2006d). Some performance issues in distributed real time database systems. *VLDB PhD Workshop (2006), The Convention and Exhibition Center (COEX).* Seoul, Korea.

Shanker, U., Misra, M., & Sarje, A. K. (2007). Distributed real time database systems: Background and literature review. *International Journal of Distributed and Parallel Databases. Springer Verlag* (accepted).

Shanker, U., Misra, M., Sarje, A. K. & Shisondia, R. (2006c). Dependency sensitive shadow SWIFT. *10th International Database Applications and Engineering Symposium (IDEAS 06),* 373-276. Delhi, India.

Ulusoy, O. (1992). Concurrency control in real-time database systems. *PhD Thesis, Department of Computer Science, University of Illinois, Urbana-Champaign.*

KEY TERMS

Atomic Commit Protocols: Ensures that either all the effects of the transaction persist or none of them persist despite of the site or communication link failures and loss of messages.

Distributed Real Time Database Systems: Collection of multiple, logically inter-related databases distributed over a computer network where transactions have explicit timing constraints, usually in the form of deadlines.

Firm Real Time Transaction: A firm real time transaction loses its value after its deadline expires.

WORKDONE/PREPARED Message: This message is send by cohort after completion of processing.

WORKSTARTED Messages: The execution phase of a cohort is divided into two parts locking phase and processing phase, and in place of WORKDONE message, a WORKSTARTED message is sent just before the start of processing phase of the cohort.

YES VOTE Message: When a cohort receives a VOTE REQ, it responds by sending a yes or no vote message (YES or NO VOTE, YES VOTE also known as PREPARED message) to the coordinator.

Chapter LXXIX
MECP: A Memory Efficient Real Time Commit Protocol

Udai Shanker
M.M.M. Eng. College, India

Manoj Misra
IIT Roorkee, India

Anil K. Sarje
IIT Roorkee, India

INTRODUCTION

Important data base system resources are the data items that can be viewed as logical resource, and CPU, disks and the main memory which are physical resources [Garcia-Molina et al. 1992]. Though, the cost of main memory is dropping rapidly and its size is increasing, the size of database is also increasing very rapidly. In real time applications, where the databases are of limited size or are growing at a slower rate than memory capacities are growing, they can be kept in the main memory. However, there are many real time applications, which handle large amount of data and require support of an intensive transaction processing. The amount of data they store is too large (and too expensive) to be stored in the main memory. Examples include telephone switching, satellite image data, radar tracking, media servers, computer aided manufacturing etc. [Ramamritham, 1993]. In these cases, the database can not be accommodated in the main memory easily. Hence, many of these types of database systems are disk resident. The buffer space in the main memory is used to store the execution code, copies of files, data pages and any temporary objects produced. The buffer manager controls the main memory and the availability of main memory space affects transaction's response time [Garcia-Molina et al. 1992]. Before starting the execution of a transaction, buffer is allocated for the transaction. When the memory is running low, a transaction may be blocked from execution. The amount of memory available in the system thus limits the number of

concurrently executable transactions. In the large-scale real time database systems, the execution of the transaction will be significantly slowed down, if available main memory is low. When the total maximum memory requirement of the admitted transactions exceeds the available memory, distributed real time database systems (DRTDBS) must decide how much memory should be given to each transaction. This decision must also take into account the transactions' timing requirements to ensure that the transactions receive their required resources in time to meet their deadlines. In addition, the effectiveness of memory allocation in reducing individual transaction's response time should be considered, so as to make the best use of the available memory. Therefore, it is important for the database designer to develop memory efficient protocols, so that more number of transactions can be executed concurrently at any time instant. In this work, design of a distributed commit protocol which optimizes memory usage has been presented.

BACKGROUND

The development of commit protocols for the traditional database systems has been an area of extensive research in the past decade. However, in case of distributed real time commit protocols, very little amount of the work has been reported in the literature. The real time commit protocol Permits Reading of Modified Prepared-Data for Timeliness (PROMPT) and deadline-driven conflict resolution (DDCR) commit protocol were proposed by Gupta et al. and Lam et al. respectively [Lam et al. 1999; Haritsa et al. 2000]. Based on the concepts of PROMPT and DDCR, Biao Qin and Yunsheng Liu proposed a new commit protocol double space commit (2SC) [Qin et al. 2003]. All the above protocols consume considerable amount of main memory for maintaining the intermediate temporary records created during the execution of transactions. In PROMPT, the lender maintains extra data structures to record the types of dependencies of its borrowers and DDCR uses more than one copy of the data items (i.e. before, after and further).

All of this not only requires extra memory but also creates additional workload on the system. Furthermore, the locking scheme used by PROMPT and 2SC protocols specifies the lock held by the lender only and these protocols either use read-before-write model or write only (blind write) model. The effect of using both models collectively has not been investigated in any previous work. So, another significant difference between our work and the works reviewed above is that we have considered both blind write (a read is not performed before the data item is written) as well as update (read-before-write). A blind-write model is not unrealistic [Burger et al. 1997] and it occurs in real life information processing for example, recording and editing new telephone numbers, opening new accounts, changing addresses etc. There are also many applications such as banking, intelligent network services database etc. where we need write without ever read model.

MECP

Here, a memory efficient commit protocol (MECP) has been proposed after redefining all kind of dependencies that may arise by allowing a committing cohort to lend its data to an executing cohort under both update (read-before-write) model and blind write (write not ever read) model. A new locking scheme has also been proposed for MECP, in which a lock not only shows the lock obtained by the lender but also the lock obtained by the borrower [Shanker et al. 2005a].

A database is considered as being made up of a set of data items associated with lock variable. The proposed memory efficient commit protocol (MECP) uses a new locking scheme which reduces the need for large number of temporary

intermediate records and thus relieves the system from additional workload.

NEW LOCKING SCHEME

A transaction (or a cohort) can lock a data item using a read/write (blind)/update lock depending on the operation that it needs to perform. In MECP, the write operation is categorized into two types: blind write and update. Subsequent occurrences of "write" should be treated as "blind write" in this work.

In addition to lock information, a flag is also attached with each data item. The flag is set to any one of the following three modes when a cohort locks a data item at the time of its arrival at a site.

Mode 1
If a cohort wants to use a data item and it is not locked by any other cohort, it sets the flag of data item in Mode 1.

Mode 2
If a cohort T_2 wants to write/update a data item read by another cohort T_1 in its committed phase, it changes the flag of the data item from Mode 1 to Mode 2. T_2 is now not allowed to commit until T_1 commits. However, if T_1 aborts, T_2 does not abort.

Mode 3
If a cohort T_2 wants to read/write/update an uncommitted data item written by another cohort T_1, it converts the flag of the data item from Mode 1 to Mode 3. Here, T_2 is not allowed to commit until T_1 commits and if T_1 aborts, T_2 also aborts.

These Modes are required to maintain the ACID properties of the transaction. If a data item is already locked and its Mode is either 2 or 3. Then, other lock requesting cohorts are not permitted to lock that data item and they will be blocked. Each site S_i maintains a Borrower_List that contains the following information.

Borrower_List (S_i) :{(T_j, D) |T_j is a borrower and has locked the dirty data D}

Data Access Conflicts Resolving Strategies

Let T_1 be a cohort in commit phase holding a lock on data item (x) and T_2 be the cohort requesting a lock on the same data item (x). The data item (x) is in mode 1 (as T_1 has locked it). Hence, there are six possible cases of data conflict.

Case 1: Read-Write OR Update Conflict
If cohort T_2 requests a Write OR Update Lock while cohort T_1 is holding a Read Lock, the flag associated with the data item is set in Mode 2 from Mode 1.

Case 2: Write-Write Conflict
If cohort T_2 requests a Write lock while cohort T_1 is holding the Write Lock, the flag associated with data item is set in Mode 3 from Mode 1.

Case 3: Update-Update Conflict
If both locks are Update - Locks, then the flag associated with data, item is set in Mode 3 from Mode 1.

Case 4: Update-Write Conflict
If cohort T_2 requests a Write Lock while cohort T_1 is holding an Update Lock, flag associated with data item is set in Mode 3 from Mode 1.

Case 5: Write-Update Conflict
If cohort T_2 requests an Update Lock while cohort T_1 is holding Write Lock, flag associated with data item is set in Mode 3 from Mode 1.

Case 6: Write OR Update-Read Conflict
If cohort T_2 requests a Read Lock while cohort T_1 is holding a Write OR Update Lock, then flag

associated with data item is set in Mode 3 from Mode 1.

Mechanics of Interaction between Lender and Borrower Cohorts

When transaction T2 accesses a data already locked by transaction T1, following possible scenarios may arise.

Scenario 1: T_1 receives decision before T_2 has completed its local data processing:

If global decision is to commit, T_1 commits.
All cohorts using the data items locked by T_1 whose flag is either in Mode 2 or in Mode 3 will execute as usual.
The flags of the data items will change from Mode 2 or Mode 3 to Mode 1.

If the global decision is to abort, T_1 aborts.
All cohorts using the data items locked by T_1 whose flag is in Mode 2 will execute as usual. Flags of the data items will change from Mode 2 to Mode 1.
All cohorts, using the data items locked by T_1 whose flag is in Mode 3 will abort. Flags of the data items will change from Mode 3 to 0.

The cohorts dependent on data set of T_1 will be deleted from the Borrower_List.

Scenario 2: T_2 completes data processing before, T_1 receives global decision:

T_2 does not send WORKDONE message.
T_2 is blocked and It has to wait, until

1. Case1: either T1 receives its global decisions, or
2. Case2: its own deadline expires,

whichever occurs earlier.

In case 1, the system will execute as in the scenario 1, whereas in case 2, T2 will be killed and will be removed from the Borrower_List.

Scenario 3: T_2 aborts before T_1 receives decision:

In this situation, T_2's updates are undone and T_2 will be removed from the Borrower_List.

MEMORY OPTIMIZATION

It is assumed that the number of data items in the database at each site is N. The amount of memory required for maintaining the record of data items lent by a single cohort is computed and compared with 2SC below.

Case 1: Memory Requirement in 2SC
At least a flag is required corresponding to every data item to show its locking status. The minimum memory required to keep this information for all data items is N/8 bytes (a flag needs at least single bit storage). Further, each site maintains a list of lenders, and each lender maintains two lists: commit dependent cohorts and abort dependent cohorts with dirty data used by them. This can be implemented in two ways.

1. Using a linear list, or
2. with a linked list

In the first case, there may be a lot of wastage of memory in maintaining the records of dependency information due to high level of dynamism in the conflicts.

In the second case, a dependency list has to be maintained which contains the identity (id) of committing cohorts (lenders) that have lent their modified data to newly arrived cohorts. Each lender in this dependency list also maintains two

ALGORITHM

On the basis of the above discussion, the complete pseudo code of the protocol is given as below.

```
if (T1 receives global decision before T2 ends execution) then
 {
One: if (T1's global decision is to commit) then
          {
          T1 commits;
All cohorts using the data items in Mode 2 or 3 locked by T1 will execute as usual;
          Flags, either in Mode 2 or 3, of data items locked by T1 are set to Mode 1.
          The cohorts dependent on T1 will be deleted from the Borrower_List;
           }
else   //  T1's global decision is to abort
          {
          T1 aborts;
All cohorts using the data items with Mode 2 flag and locked by T1 will execute as usual. Flag will change
from Mode 2 to Mode 1of data items locked by T1;
All cohorts using the data items with Mode 3 flag and locked by T1 will abort. Flag in Mode 3 of data
items locked by T1 is set to 0.
The cohorts dependent on data set of T1 will be deleted from the Borrower_List;
           }
}
else
{
if ( T2 ends execution  before T1 receives global decision )
          {
                    T2's workdone message is blocked;
                    T2 waits for next event/message;
                              Switch (type of next event/message)
                              {
                              Case 1: if (T2 misses deadline)
                              {
                                        Undo the computation of T2;
                                        Abort T2;
                                        Delete T2 from the Borrower_List;
                              }
                              Case 2: if (T1 commits/aborts)
                                        GoTo One;
                               }
          }
else  // T2 is aborted by higher transaction before T1 receives decision
          {
                    Undo the computation of T2;
                    Abort T2;
                    Delete T2 from the Borrower_List;
          }
}
```

lists to record ids of abort and commit dependent cohorts with dirty data items utilized by them. The memory required for keeping the record of data items lent by a single cohort is computed below. Let us assume that, on an average, each lender has p nodes in commit dependency list and q nodes in abort dependency list. The total memory (M) required by each lender is given as:

M=M1+ (p+q)*M2, where

M1 Memory required for a node of the dependency list corresponding to the lending cohort. It is 14 bytes (2 bytes for id of the lender, 4 bytes for the

M2 address of commit dependent cohort list, 4 bytes for the address of abort dependent cohort list and 4 bytes for

N address of the next node)
Memory required by one node of the list of commit/abort dependent cohorts. It is 8 bytes. (8 bytes=2 bytes for borrower cohort id + 2 bytes for dirty data + 4 bytes for address of the next node)
No. of data items that is lent by a cohort=p+q

Case 2: Memory Requirement in Proposed Scheme (MECP)

Memory required by MECP is given below.

Minimum two bits are needed to record the Modes of every data item at a site. So the total memory required for all data items is 2*N/8 bytes. The proposed protocol maintains a single Borrower_List for keeping the information of borrowers and the dirty data used by them. This requires 8 bytes of memory (2 bytes for borrower id + 2 bytes for dirty data + 4 bytes for address of the next node) for one node of the Borrower_List. Total number of nodes in this Borrower_List corresponding to one lender will be N_d.

Total Memory Requirement (per lender in MECP) = $2*N/8 + 8*N_d$.

Compared to case 1, there is an additional need of N/8 bytes at each site to keep the information about the Mode of every data item. On the other hand if there are N_L lenders at any instant of time, the additional memory required is $14*N_L$ bytes in case 1 as compared to case 2 (see in table 1). With the increase in the transaction arrival rate and transaction size, there are more chances of conflicts resulting in more number of dependencies and more lenders. Although, it seems that initially more number of bytes are needed for keeping the Mode information of data items, the proposed protocol competes with 2SC at high transaction arrival rate.

The MECP will be beneficial, if the following condition holds.

$$(2*N/8 + 8*N_d* N_L) < (N/8 + N_L*(14+8*N_d)), \text{ or}$$
$$N_L > N/112$$

The proposed protocol is memory efficient only, if the number of lenders at an instant is more than the number of data items/112 at a site [Shanker, 2006b]; otherwise, it consumes the

Table 1. Memory Requirement of 2SC and MECP

Commit Protocol	flags at each site (Memory in bytes)	Single lender (Memory in bytes)
2SC	N/8	$14+8*N_d$
MECP	2*N/8	$8*N_d$

same amount of memory and will be at par with 2SC and PROMPT.

The MECP improves the system performance due to the following reasons [Shanker et al. 2006a].

1. By reducing the memory requirements needed by the system as compared to 2SC. Thus, this protocol is suited to real time applications where there is a scarcity of the available memory [Kayan et al. 1999].
2. The new locking scheme ensures that a borrower can't be a lender simultaneously at the same site. This relieves the system from the burden of checking that a borrower is not trying to lend as compared with PROMPT and 2SC.
3. The new locking scheme ensures that the same data can not be used by another borrower simultaneously as compared to PROMPT and 2SC, where there is a need for checking this.

The performance of MECP is compared with PROMPT and 2SC and is better with these commit protocols in term of Miss Percentage of the transaction, and also it reduces the memory requirement to a great extent [Shanker et al. 2005b] depending on number of lenders at an instant. This makes it suitable for data intensive applications with high transaction arrival rate where system's main memory size is a bottleneck.

FUTURE WORKS

Following are some suggestions to extend this work [Shanker, 2001; Shanker, 2007].

1. Our work can be extended for Mobile DTRDBS, where memory space, power and communication bandwidth is a bottleneck. The MECP will be well suited to hand held devices and possibility of its use for commit procedure can be explored.
2. Our performance studies are based on the assumption that there is no replication. Hence, a study of relative performance of the topic discussed here deserves a further look under assumption of replicated data.
3. The fault tolerance and the reliability are highly desirable in many real time applications because in these applications, continued operation under catastrophic failure and quick recovery from failure is very crucial. These aspects may also be dealt.
4. Our research work can also be extended for grid database systems.

CONCLUSION

Here, a new distributed real time commit protocol (MECP)has been presented that uses a new locking scheme. The new locking scheme ensures that a borrower can't be a lender simultaneously at the same site and the same data can not be used by another borrower simultaneously as compared to PROMPT and 2SC, where there is a need for checking this. It not only optimizes the storage cost but also considers blind and update type writes collectively for DRTDBS. The simulation results show that the protocol performs better than PROMPT and 2SC commit protocols. It is well suited to data intensive applications where transaction arrival rate is high and the sizes of transactions are large.

REFERENCES

Shanker, U., Misra, M., & Sarje, A. K. (2007). Distributed Real Time Database Systems: Background and Literature Review. International Journal of Distributed and Parallel Databases, Springer Verlag (under second review).

Shanker, U., Misra, M., & Sarje, A. K. (2006a). Some Performance Issues in Distributed Real Time Database Systems. VLDB PhD Workshop (2006), The Convention and Exhibition Center (COEX), Seoul, Korea.

Shanker, U. (2006b). Some Performance Issues in Distributed Real Time Database Systems. PhD Thesis, Department of Electronics & Computer Engineering, Indian Institute of Technology Roorkee, India.

Shanker, U., Misra, M., & Sarje, A. K. (2005a). A Memory Efficient Fast Distributed Real Time Commit Protocol. 7th International Workshop on Distributed Computing (IWDC 2005), Indian Institute of Technology Kharagpur, India, 500-505.

Shanker, U., Misra, M., & Sarje, A. K. (2005b). The MEWS Distributed Real Time Commit Protocol. WSEAS Transactions on COMPUTERS, 4(7), 777-786.

Shanker, U., Misra, M., & Sarje, A. K. (2001). Hard Real-Time Distributed Database Systems: Future Directions. All India Seminar on Recent Trends in Computer Communication Networks, Department of Electronics & Computer Engineering, Indian Institute of Technology Roorkee, India, 172-177.

Garcia – Molina, H., & Salem, K. (1992). **Main Memory Database Systems: An Overview.** IEEE Transactions on Knowledge and Data Engineering, 4(6), 509-516.

Burger, Albert., Kumar, Vijay & Hines, Mary Lou. (1997). Performance of Multiversion and Distributed Two - Phase Locking Concurrency Control Mechanisms in Distributed Databases. International Journal of Information Sciences, 1-2, 129-152.

Kayan, Ersan, & Ulusoy, O. (1999). An Evaluation of Real Time Transaction Management Issues in Mobile Database Systems. Computer Journal, 42(6).

Ramamritham, K. (1993). Real-time Databases. Distributed and Parallel Databases. Special Issue: Research Topics in Distributed and Parallel Databases, 1(2), 199-226.

Haritsa, J. R., Ramamritham, K. & Gupta, R. (2000). The PROMPT Real-time Commit Protocol. IEEE Transactions on Parallel and Distributed Systems, 11(2), 160-181.

Lam, K.Y., Pang, C., Son, S. H. & Cao, J. (1999). Resolving Executing -Committing Conflicts in Distributed Real - time Database Systems. Journals of Computer, 42(8), 674-692.

Qin, B., & Liu, Y. (2003). High Performance Distributed Real-time Commit Protocol. Journal of Systems and Software, Elsevier Science Inc., 68(2), 145-152.

KEY TERMS

Atomic Commit Protocols: Ensures that either all the effects of the transaction persist or none of them persist despite of the site or communication link failures and loss of messages.

Blind Write: A read is not performed before the data item is written.

Borrower: The transaction using modified but uncommitted data of lenders.

Distributed Real Time Database Systems: Collection of multiple, logically inter-related databases distributed over a computer network where transactions have explicit timing constraints, usually in the form of deadlines.

Firm Real Time Transaction: A firm real time transaction loses its value after its deadline expires.

Lender: The transaction that has lent their modified data to newly arrived cohorts

Update: Read before write.

Chapter LXXX
Self-Tuning Database Management Systems

Camilo Porto Nunes
Federal University of Campina Grande, Brazil

Cláudio de Souza Baptista
Federal University of Campina Grande, Brazil

Marcus Costa Sampaio
Federal University of Campina Grande, Brazil

INTRODUCTION

Computing systems have become more complex and there is a plethora of systems in heterogeneous and autonomous platforms, from mainframes to mobile devices, which need to interoperate and lack effective management.

This complexity has demanded huge investments to enable these systems to work properly. It is necessary to invest on software acquisition and installation: management, administration, and update. These costs compound the *Total Cost of Ownership* (TCO), which tends to increase exponentially according to the software complexity.

Information Technology (IT) focuses mainly on providing information services in order to achieve simplicity, agility, large access to information, and competitivity. **Database Management Systems** (**DBMS**) are part of the IT infrastructure in large, medium, and even small enterprises.

According to LIGHTSTONE (2003), the complexity of a DBMS enhances the difficulty in administrating it, as there are too many tasks a Database Administrator (DBA) should consider. Thus, more people are needed to be trained to efficiently manage this software. As a result, the maintenance costs increase and the problem is not properly solved. Performance, time-to-market, throughput, robustness, availability, security, are some concerns that a **DBA** should consider.

Suppose that, due to an unexpected event (e.g., too many clients accessing the DBMS), the average load in a database server changes quickly and the query response times may become unacceptable. The **DBA** must, rapidly (a) detect the problem (unacceptable response time); (b) find the root cause

of the problem (e.g., too many swap operations); and (c) tune the server to maintain response time goal (e.g., to increase memory size).

These three activities—symptom detection, diagnosis, and tuning—if executed manually by a DBA, can take a long time. Worse still, the DBA could be absent by the time the symptom arises.

On the other hand, **autonomic computing** has become an important research theme, aiming to reduce the manual efforts of administration of complex systems. Regarding DBMSs, they should be capable to react automatically to negative events, under the form of adaptive continuos tuning (LIGHTSTONE, 2003). For example, when detecting a load change, with risk of degrading the global performance of the system, a DBMS could automatically adjust some query execution plans; change its memory pool configuration; or still adjust data on disk (through index creation, data clustering, table partition, and so on) in order to maintain its overall performance. DBMSs with some capabilities of adaptive continuous tuning are called **self-tuning** ones.

This chapter addresses the issue of self-tuning DBMS. In the remainder of the chapter we present a background on this topic, followed by a discussion focusing on performance, indexing, and memory issues. Then, we highlight future trends and conclude the chapter.

BACKGROUND

Manual **DBMS tuning** requires a collection and analysis of some performance metrics by the DBA. Examples of such metrics include hit ratio in data cache, number of I/O executed in a given table, etc. After collection and analysis of these metrics, the **DBA** may decide whether it is necessary to adjust the DBMS. If an intervention is due, then the **DBA** needs to know which components are likely to be adjusted and what are the impacts of such adjustments.

In order to adjust the DBMS to improve performance, the **DBA** may act in several components such as memory buffers, index creation, SQL tuning, etc. In this chapter we focus on index and memory tuning, due to their importance in the overall performance of the DBMS.

Memory buffers consist of memory areas which are used for a given purpose. DBMS uses them, for example, to store the query execution plans, for both data and SQL statement caching, as an area for data sorting, and for storing session data. The tuning of these buffers may improve the performance of the submitted queries. For example, by increasing the memory buffer size, the number of physical I/O operations may be reduced, which results in performance gains.

Index maintenance is one of the most important tasks in DBMS administration. Hence, it deserves special attention due to its importance in the whole tuning process. An **index** may help the execution of a query submitted to the DBMS by reducing the number of accesses to disk. Nonetheless, during data update, insertion and deletion, an index may decrease the DBMS performance, as index reorganization may be necessary in such operations.

Once DBMS tuning involves many parameters, components and metrics, it is very difficult for humans to efficiently complete this task. Therefore, there has been a great effort from both academia and industry in providing automatic solutions for database administration.

Automatic Diagnosis of Performance Problems

The main motivation to automate the task of diagnosing performance problems concerns the complexity and importance for the subsequent decision making to solve the problem. The diagnosis, when executed manually, usually depends on DBA's experience, because of the great number of metrics involved. Then it is difficult and ex-

pensive to proceed with analysis and correlation of such metrics in order to find the root cause of the problem.

Some related works (BERNOIT, 2005; DAGEVILLE, 2006) are based on two postulates: (1) it should be possible to define units of measure for different performance metrics which are either compatible or additive; and (2) both the many phases of processing user requests in charge of DBMS components and the many database resources used may be disposed in a hierarchy, so that drill-down / roll-up operations may be executed.

DBTime is an important concept in tuning, as it measures the time spent by a DBMS when processing user requests. Nonetheless, it does not include time spent in the intervening layers such as the network and middle tiers. *DBTime* includes tasks such as *connecting to the DBMS*, *parsing SQL statements*, *optimizing SQL statements*, *executing SQL statements* and *fetching query results.* The meaning of this concept is the measurement of the total amount of work done by the DBMS, and the *DBTime* rate is the DBMS load average. Figure 1 illustrates this concept.

Notice that DBMS (sub)components like 'Query Parser' and 'Query Optimizer' consume DBMS resources that include both hardware resources like CPU and I/O devices and DBMS resources like index, join algorithms, etc. The question to be answered is: in a given time interval, which DBMS components and hardware resources are being overloaded, in average?

To answer this question, the hierarchy of tasks for processing user requests as well as the underlying hierarchy of resources should be explored. A DBMS then could be modeled like an oriented graph, the nodes of which are either tasks or resources. More precisely, the root node represents the DBMS with its total *DBTime* rate; the second level represents the users connected to the DBMS during the time interval, with their portions of consumed *DBTime* rate; the other intermediary nodes represent either tasks or resources, with their respective percentual of the DBMS *DBTime* rate; finally, the leaf nodes are always atomic resources. Figure 2 presents such graph.

Considering that the DBMS has been correctly and extensively instrumented to obtain precise timing information — saying in another way, all components and resources have been correctly monitored —, the automatic detection and diagnosis of a performance problem, with the help of a *DBTime* graph consists of two steps: (1) to identify the leaf node(s) having a proportion of DBMS *DBTime* above a specified threshold; and (2) to roll up the oriented graph from the leaf node(s) with problematic *DBTime*. These identified nodes could indicate the root cause of the performance problem; soon after, by rolling up the

Figure 1. DBTime

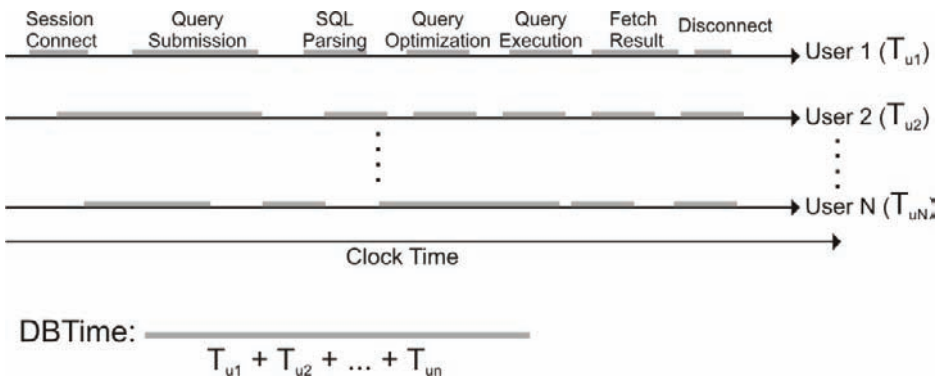
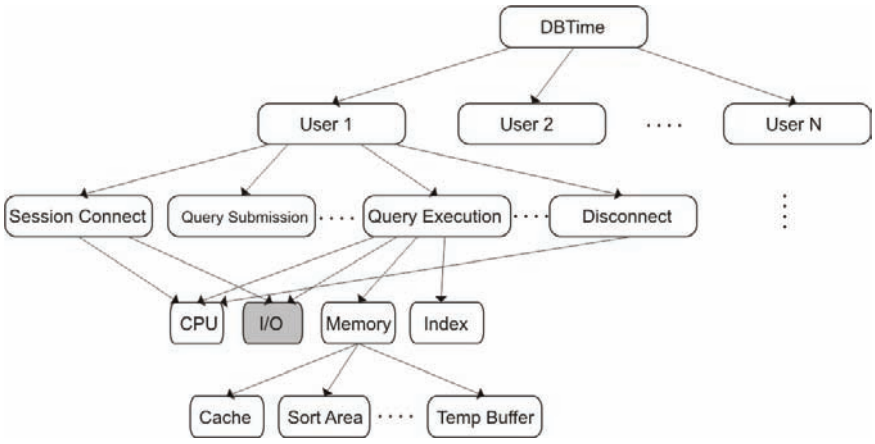


Figure 2. A DBTime oriented graph



graph from an identified node, a diagnosis of the problem can be presented to the DBA. Inversely, if the DBA wants to know where is the bottleneck for a resource or component (intermediary nodes), the DBMS can find the answer by drilling down the graph from the resource or component until to reach the cause node.

AUTOMATED INDEX TUNING

Indexing tuning is not a trivial task. Frequently the DBA does not have enough time to analyze index metrics, whether it is interesting to create a new indexing or remove an existent one. Hence, automatizing this process seems to be a very good approach.

The basic idea consists of submitting to the DBMS a set of statements, named workload. The self-tuning index manager will analyze the workload, according to criteria such as query filters, tables and attributes, table joins, index selectivity, and so on. Then, two decisions may arise: 1) the hint to the DBA to create or delete some indexes; or 2) to automatically create or remove indexes, that is, without DBA intervention.

To find out whether a given index may improve performance in an execution of a SQL statement, the index manager may use the index statistics or virtual indexes. The procedure is done as follows: for each SQL statement from the workload, the index manager analyzes its structure and selects a set of indexes, known as candidate index set, which may give a gain in performance. Then the information about the candidate indexes are inserted in the DBMS catalogue, but their respective data structures are not created (that is why they are called virtual indexes). This is done in order to enable the virtual indexes to be taken into account by the query optimizer. If during the generation of the query execution plan, a virtual index is invoked, then this index will be either suggested to be created by the DBA, or automatically created.

There are some variations of the previous procedure due to side effects which may be generated. One of them is related to local optimization, once those indexes are analyzed individually for each statement in the workload submitted to the DBMS. This may result in an index providing an excellent performance for a specific SQL statement, but unacceptable slow for other SQL statement.

That is, the index local analysis does not assure a global optimization in the workload. In the following we discuss some research works which proposes self-tuning indexing techniques.

SATTLER (2005) propose an access-balanced index structure, Simple Adaptable Binary Tree (SABT), instead of traditional data-balanced ones. Their experimental evaluation has evidenced that the benefit of SABT is greater for data accessed more frequently than for data accessed less often. Essentially, a SABT has an adjustable size, i.e. the number of nodes can be increased or decreased during runtime, and it is deeper for data which are accessed more often to minimize page access. Moreover: (1) the index is sparse; (2) the leaf nodes are page containers, which hold pages of tuples with keys of the value range specified by the tree; (3) page containers keep local page access statistics and (4) the lookup in the tree works as usual.

The crucial issues of reorganizing and balancing is resolved by observing if a page container is more or less accessed; in other words, if its access counter is above or below a certain threshold. In the first case, and if there are free nodes, a new node is inserted with the median key from the container, and the container is split into two according new containers. By judiciously configuring the free node pool variable, the tree can always grow, when necessary. On the other hand, shrinking can be done the same way. Finally, these reorganization as well as statistics updates are performed during lookup operations.

The main drawback of this approach seems to be the fact that binary trees are not the optimal choice for a DBMS index structure. Other open issues include concurrency (hot spots), implications for secondary indexes and multidimensional indexing schemes, and operations based on access-balanced index structures.

Due to space limitations, we now shortly comment some other related works. LIFSCHITZ (2005) and COSTA (2005) present an architecture based on agents to implement the automatic management of indexes. DAGEVILLE (2004) highlights the Oracle 10g component which is responsible for SQL tuning and it recommends the creation of indexes to improve the performance of the top queries. SCHNAITTER (2006) implements an index manager for the PostgreSQL DBMS and it follows the same ideas discussed in this section. AGRAWAL (2005) proposes an index manager for the SQL Server 2005 which aims optimal optimization of a workload submitted to the DBMS. SATTLER (2003) presents a solution for indexing automatic management which does not need DBA intervention.

Automatic Memory Management

Memory is considered one of most used resources used by DBMSs. Obviously, this is due to its fast access when compared to disks. For this reason, DBMSs try to automatically place in the memory the most frequently used objects to minimize disk read/write operations (MULLINS, 2005).

Nonetheless, several kinds of data compete for memory including application data, execution plans of frequent queries, stored procedures, concurrency locks and log buffers. Furthermore, some memory areas should be allocated and reserved for special types of processing, such as data sorting, join operations, temporary objects and so on.

Once the memory used by DBMSs is limited, adjustments in any **memory buffer** unavoidably will affect other buffers. Hence, memory tuning is a tricky task.

The memory allocation problem can be defined as follows: given a workload and a fixed and limited amount of memory, it is necessary to determine the size of each specific buffer area in order to optimize overall performance. This would demand from the DBA a strong expertise. Moreover, sometimes DBA's experience is not enough to succeed in **memory optimization**. The

reason for that is that the time spent by DBA to make an analysis of the workload parameters and, soon after, to decide which adjustments should be made can be quite long. Thus, it may compromise the quality of the DBMS tuning.

Due to these difficulties, it is necessary to implement a DBMS autonomous component that is able to manage memory buffers automatically, efficiently and effectively.

There are several research works on **automatic memory management**. XI (2001) presents an analytic model, using chains of Markov, for prediction of the hit ratio of the data cache of a DBMS. POWLEY (2005) describes an autonomous component for administration of the PostgreSQL memory buffers. The rationale for that component consists of monitoring the average time of I/O requests; in case the I/O resource suffers an increment of 5% in its average value, an algorithm of memory buffer configuration is immediately executed.

BROWN (1996) has proposed a technique which is used to assist response time requirements for different classes of transactions submitted to a DBMS. The solution consists of allocating data used by different classes of transactions in different data caches (fragments). This approach assumes the proportional hypothesis between buffer miss ratio and response time. Later on, this technique was improved with the incorporation of a model for estimating buffer hit ratio. The new model is based on the Theorem of the Concavity which states: "… the hit ratio, as a function of the allocation of memory buffer, is a concave function". Hence, a policy for buffers allocation may be mathematically optimum.

MARTIN (2006) presents an approach to map database objects into different data caches with the goal of maximizing the database transaction throughput. The algorithm takes into account the access patterns and inherent characteristics of database objects for a given workload. Similar objects are then grouped in the same buffer pools. The amount of memory allocated to each buffer pool is determined according to the size of the database objects that will reside in the respective buffer pool. The approach uses data mining techniques to analyze and discover access patterns for database objects.

To summarize, state of the art on automatic memory management presents some open issues:

1. Current techniques have focused only on buffer pool but they have not addressed other important memory areas such as those used for statements, sorting, and temporary data;
2. Lack of a holistic approach which would address not only memory but also other DBMS components such as indexes, disks, and optimizer;
3. There are few works on open-source database systems, once many contributions focus on DBMS proprietary architecture;
4. Lack of standards in the monitoring process; and
5. Lack of a framework which may monitor heterogeneous DBMS.

FUTURE TRENDS

There are still open issues related to the ability of a DBMS to execute DBA tasks automatically. Scientists are seeking alternatives to offer larger autonomy to DBMSs. According to LIGHTSTONE (2003), besides the characteristics discussed up to now, future DBMSs will have to incorporate some essential characteristics of autonomic computing (KEPHART, 2005) to reduce their maintenance and administration costs. This tendency aims, in a holistic way, to turn DBMSs capable to automatically accomplish any task, besides those related to performance administration. That is, DBMSs, instead of self-tuning, would become self-manageable.

The first characteristic to be incorporated is the continuous adjustment and adaptation of a DBMS face to changes of any type. More specifically, the DBMS should monitor itself to detect changes occurred in the workload and to adjust parameters so that some of its components may adapt to the changes. Query optimizers should automatically search for new execution plans, memory buffers should be readjusted to support an abrupt change of workload, and the physical database design should be altered in an automatic way to create new indexes, to remove undesirable indexes, to partition tables, to materialize views, etc.

The second characteristic that a DBMS should have is the capacity of discovering the workload type, for example, if it is either OLTP or OLAP. As it is known, these two workload types have quite different characteristics, demanding different database design and implementation.

The automatic detection and integration of new sources of data is also an important issue. Many human and financial resources have been spent to integrate different sources of corporation data. In this sense, it is highly recommended that DBMSs be able to discover where new sources of data are and how to integrate them into the corporative database.

Concerning the physical storage, a DBMS should be able to optimize disk usage in order to reduce data storage costs. The aim is to save physical space when this is restricted (e.g. by removing indexes or avoiding materialized views), or then to use the available disk free space (e.g. by creating indexes and materialized views).

DBMSs in the close future should be able to automatically react to eventual exception conditions, such as high disk occupation, or lack of enough memory. Also, functions such as subminute automatic failover (LIGHTSTONE, 2003) allow for preventing to detect situations in which there will occur exceptions: this way, DBMSs will become pro-active *stricto sensu*.

Last but not the least, DBMSs, besides explaining the reason of their decisions, should interact with DBA to receive his feedback, so that DBMSs can improve their decisions.

CONCLUSION

This chapter has addressed the importance of self-tuning DBMS. We have highlighted the main difficulties for a DBA to proper manage a DBMS currently, due to the large complexity of the latter. Then, the main issues on self-tuning DBMS — performance, indexing and memory — have been discussed. Lastly, future trends on self-tuning DBMSs have been examined: the main verification is that self-tuning DBMSs should evolve for self-manageable DBMSs.

Self-tuning databases are a very hot research topic which has been investigated by large database companies and academy. It is part of the next generation of computer systems based on autonomic computing principles.

REFERENCES

Agrawal, S., Chaudhuri, S., Kollar, L., Marathe, A., Narasayya, V., & Syamala, M. (2004). Database tuning advisor for Microsoft SQL server 2005. *In Proceedings of the 30th Very Large Database Conference* (VLDB), 1110-1121. Toronto, Canada.

Bernoit, D. G. (2005). Automatic diagnosis of performance problem in database management systems. *In Proceedings of the Second International Conference on Autonomic Computing (ICAC'05)*, 326-327.

Brown, K. P., Carey, M. J., & Livny, M. (1993). Managing memory to meet multiclass workload response time goals. In Proceedings of 19th VLDB, 328-341. Dublin, Ireland.

Brown, K. P., Carey, M. J., & Livny, M. (1996) Goal-oriented buffer management revisited. *In*

Proceedings. Of the ACM SIGMOD International Conference on Management of Data, 353- 364.

Costa, R. L. de C., Lifschitz, S., Noronha, M. F de., & Salles, M. A. V. (2005). Implementation of an agent architecture for automated index tuning. In Proceedings. *Of the 21st International Conference on Data Engineering Workshops (ICDEW),* 1215-1222.

Dageville, B., & Dias, K. (2006) Oracle's self-tuning architecture and solutions. *Bulletin of the IEEE Computer Society Technical Committee on Data Enginering, 29*(3).

Dageville, B., Das, D., & Dias, K. (2004). Automatic SQL tuning in oracle 10g. *In Proceedings of the 30th Very Large Database Conference (VLDB),* 1098-1109. Toronto, Canada.

Dias, K., Ramacher, M., Shaft, U., Vftenkataramani, V., & Wood, G. (2005). Automatic performance diagnosis and tuning in oracle. *In Proceedings of CIDR.*

Kephart, J. O. (2005). Research challeneges of autonomic computing. *In Proceedings of the 27th International Conference on Software engineering,* 15-22.

Lifschitz, S., & Sales, M. A. V. (2005). Autonomic index management. *In Proceedings of the Second International Conference on Autonomic Computing (ICAC'05),*304-305.

Lightstone, S., Schiefer, B., Zilio, D., & Kleewein, J. (2003). Autonomic computing for relational database: The ten-year vision. In *Proceedings of the IEEE Workshop Autonomic Computing Principles and Architectures (AUCOPA'03),* 419-424.

Martin, P., Powley, X. U., & Tian, W. (2006). Automated configuration of multiple buffer pools. *The Computer Journal. IEEE, 49*(4).

Mullins, C. S. (2005). Database administration. *The complete guide to practices and procedures.* Addison-Wesley.

Powley, W., Martin, P., Ogeer, N., & Tian, W. (2005). Autonomic buffer pool configuration in PostgreSQL. IEEE *International Conference on Systems, Man and Cybernetics,* 53-58.

Sattler, K., Geist, I., & Schallehn, E. (2003). QUIET: Continuos query-driven index tuning. *In Proceedings Of the 20th VLDB Conference,* 1129-1132. Berlim, Germany.

Sattler, K., Geist, I., & Schallehn, E. (2005). Towards index schemes for self-tuning DBMS. *In Proceedings of the 21st International Conference on Data Engineering Workshops,* 1216-1223.

Sschnaitter, K., Abiteboul, S., Milo, T., & Polyzotis, N. (2006). COLT: Continuous on-line tuning. *In Proceedings of the 2006 ACM SIGMOD International Conference on Management of data*, 793-795).

Xi, Y., Martin, P., & Powley, W. (2001). An analytical model for buffer hit rate prediction. *In Proceedings Of the 2001 Conference of the Centre for Advanced Studies on Collaborative Research*, 18-29.

KEY TERMS

Cache Hit Ratio: Is a ratio of buffer cache hits to total of requests, that is, the probability that a data block will be in memory on a subsequent block re-read.

Cache Miss Ratio: A ratio of the number of times a DBMS cannot find a given data in the memory.

Database Administrator (DBA): Profissional responsible for DBMS administration.

Database Management Systems (DBMS): Software responsible for management of data in a corporation.

Index: Data structure which enables fast access to data stored in disks.

Memory Buffer: Area of memory which is alocated for a specific destination, such as data caching, session caching, program caching, sorting, and temporary area.

Self-Manageable Database: DBMS which is able to self-management, that is, it is able to execute the DBA´s tasks automatically.

Self-Tuning Database: DBMS which is able to automatically administrate and tune its performance.

Chapter LXXXI
A Survey of Approaches to Database Replication

F. D. Muñoz-Escoí
Universidad Politécnica de Valencia, Spain

H. Decker
Universidad Politécnica de Valencia, Spain

J. E. Armendáriz
Universidad Politécnica de Valencia, Spain

J. R. González de Mendívil
Universidad Politécnica de Valencia, Spain

INTRODUCTION

Databases are replicated in order to get two complementary features: performance improvement and *high availability*. Performance can be improved when a database is replicated since each replica can serve read-only accesses without requiring any coordination with the rest of replicas. Thus, when most of the application accesses to the data are read-only, they can be served locally and without preventing accesses in the same or other replicas. Moreover, with a careful management, the failure of one or more replicas does not compromise the availability of the database.

Initially, database replication management was decomposed into two tasks: concurrency control and replica control, usually solved by different protocols. The solutions of the non-replicated domain were evolved into distributed concurrency control protocols (Bernstein & Goodman, 1981), taking as their base either the two-phase-locking (2PL) or some timestamp-ordering protocol. Replica con-

trol management was based on *voting techniques* (Gifford, 1979). These voting techniques assign a given number of votes to each replica, usually one, and require that each read access collects a read *quorum* (i.e., "r" votes) and each write access a write *quorum* (i.e., "w" votes). The database must assign version numbers to the items being replicated, and the values of "r" and "w" must ensure that r+w is greater than the total number of votes, and that "w" is greater than a half of the amount of votes. Thus, it can be guaranteed that each access to the data reaches at least one copy with the latest version number for each item. This ensured consistency, but the communication costs introduced by these techniques were high.

Voting replica-control protocols were still used in the next decade, boosting their features and including also management for system partition handling when dynamic voting approaches were included (Jajodia & Mutchler, 1990).

However, replication management is not so easy to achieve when both concurrency and replica control are merged, since what the replica control protocols do for ensuring consistency has to be accepted by the concurrency control being used. Deadlocks and transaction abortions are common when both protocols are joined. So, it seems adequate to find better solutions for this global management, i.e., replication protocols that consider both concurrency and replica controls. A first example of this combined technique is (Thomas, 1979), where a special kind of *voting technique* is combined with timestamp-based concurrency control. However, his solution still relies on simple communication primitives, and is not efficient enough, both in terms of response time and abortion rate. Note that efficient total order broadcast protocols were not produced until the middle eighties (Birman & Joseph, 1987), and they could not be used in these first stages.

So, new replication techniques were introduced for databases, as an evolution of the process replication approaches found in distributed systems. Thus, depending on the criteria being used, several classifications are possible (Gray et al., 1996; Wiesmann et al., 2000). The proposal of Wiesmann & Schiper (2005), distinguishing five different techniques (active, weak-voting, certification-based, primary copy and lazy replication) will be followed here. In all these techniques, the protocols need a reliable total order broadcast in order to propagate the transaction updates. This is the regular way for achieving replica consistency in any distributed application.

ACTIVE REPLICATION

In the *active replication* technique, the client initially submits a transaction request to one of the replicas. Such *delegate replica* broadcasts the request to all replicas and all of them process the transaction from its start. Note that different transactions can use different *delegate replicas*. This technique requires complete determinism in the execution of a transaction, since otherwise the resulting state could be different among replicas. In order to easily develop this model, transactions can be implemented as stored procedures. Note that all transaction operations or parameters should be known before such transaction is started, being propagated in its starting broadcast.

The main advantage of this technique is that the support for executing transactions can be the same in both *delegate* and non-delegate *replicas*. On the other hand, its disadvantages consist of (1) requiring determinism in the transaction code, and (2) compelling read operations to be executed in all replicas, losing the possibility of balancing reads among replicas, and thus compromising one of the best performance improvements of database replication.

This technique is a direct translation of the *active replication* model for distributed systems. It has not been directly used for database replication, but there are some adaptations that have eliminated several of its intrinsic problems, improving its performance and behavior. One of

such approaches is the Database State Machine (Pedone et al., 2003) that uses deferred updates, i.e., it delays the synchronization/broadcast point until commit is requested in the *delegate replica*.

WEAK-VOTING REPLICATION

In the *weak-voting replication* technique, the *delegate replica* initially executes the complete transaction, and when the application requests its commitment, its writeset is collected and broadcast to all replicas (including the delegate one). Once such writeset is delivered, the replication protocol evaluates if there is any conflict with any previously committed transaction. If so, the transaction is aborted. Otherwise it is accepted and committed. Once the termination decision has been taken, a second message is broadcast communicating to the other replicas its result (commit or abort). The non-delegate replicas can not decide by themselves, and they should wait such a second broadcast in order to complete such transaction.

The advantages of this technique are: (1) no readset should be broadcast in order to decide the outcome of a transaction, since read-write conflicts can be evaluated in the *delegate replicas*; i.e., the readset of the local transaction against the writesets of the remote ones, (2) read-only transactions can be locally executed and completed in their *delegate replicas*, without needing any broadcast. On the other hand, its main inconvenience consists in needing two broadcasts in order to complete an updating transaction, although the second one does not need to be totally ordered. Note that the other techniques only need a single broadcast per transaction.

This technique is particularly suitable for replication protocols ensuring the serializable isolation level, since such level needs to evaluate read-write conflicts, and they do not demand readset propagation in this technique. The SER protocol of Kemme & Alonso (2000) is a good sample of this kind of replication approach.

CERTIFICATION-BASED REPLICATION

The *certification-based replication* technique shares some characteristics with the *weak-voting* one, but using a symmetrical transaction evaluation approach; i.e., letting each replica to individually decide the outcome of each transaction. To this end, this technique needs to total-order broadcast the transaction readset and writeset when its commit is being requested by the user application. As a result, when such message is delivered, all replicas share the same historic list of delivered messages and can look for conflicts with concurrent transactions in the same way. This eliminates the need of a second reliable broadcast for announcing the outcome of each transaction. Note, however, that this validation/certification process needs both readsets and writesets in order to work; at least, when read-write conflicts should be evaluated.

Hopefully, not all isolation levels demand read-write conflict evaluation. Indeed, the *snapshot isolation* (Berenson et al., 1995) level (or SI, for short) only needs to check for write-write conflicts. So, the *certification-based replication* technique has been mainly used in order to provide such isolation level. Examples of this kind of protocols are (Elnikety et al., 2005; Lin et al., 2005; Muñoz-Escoí et al., 2006).

PRIMARY COPY REPLICATION

In the *primary copy replication* technique all transactions are forwarded to and executed by a single replica, the primary one. The other replicas are only its backups or secondaries, and apply the updates of writing transactions before they are committed in the primary. Since this approach easily overloads the primary replica, read-only transactions can also be applied in secondary replicas, thus balancing the load.

At a glance, the main problem of this technique is its lack of scalability, since updating transac-

tions should be executed by a single replica and this compromises its performance. However, this also introduces some performance gains, since this kind of effort removes the need of coordination among multiple replicas in order to decide the outcome of each transaction. Moreover, since local concurrency control can be used, conflicting transactions improve their probability of success since a pessimistic concurrency control (e.g., 2PL) can be employed.

Indeed, Patiño-Martínez et al. (2005) have used the primary copy technique in order to increase the performance of a middleware-based data replication system. To this end, they divide the database into a set of conflict classes, and assign a master replica to each of such classes; i.e., playing the primary role for such conflict class. Each incoming transaction is forwarded by its *delegate replica* to its associated master replica, once the items to be accessed are known. Thus, the load can be easily balanced, and the concurrency control can be locally managed by each master replica. As a result, the performance is highly increased.

LAZY REPLICATION

Lazy replication propagates the updates of a transaction once it has already committed. This allows a fast transaction completion, but does not always ensure replica consistency and may lead to a high abort rate. Despite its disadvantages, this technique has been used in several commercial DBMSs and it is the main option when mobile or disconnected databases are considered.

If any replica can update directly its local data, transmitting later the updates to other replicas, concurrent load may lead to a high abort rate. For concurrency control purposes, a timestamp-based solution can be used. There have not been many replication protocols of this kind, being Irún et al. (2003) one of the exceptions. However, its solution is not completely lazy, but hybrid, since the number of replicas that receive the updates before commit time is configurable. In its purely lazy configuration, the abort rate is reduced using an empirical expression that forecasts the probability that a data item was outdated, before it is locally accessed. All computations needed in these expressions use locally collected data. If such a probability exceeds a dynamically-set threshold, the data item is updated from its owner replica. Such owner replica always maintains the latest version of the item. Different items may have different owner replicas. Note that when the protocol arrives to its voting termination stage, the probability of success is improved.

In lazy primary copy protocols, since the accesses are always managed by the same replica, such a replica may use any local concurrency control approach for avoiding conflicts between transactions. Primary copy solutions have been used in some commercial database systems, for instance Sybase Replication Server 11 (Breton, 1998). In these systems two trends can be found. The first one uses replication to ensure only availability, not to improve performance. In such cases, the replicas behave as standby copies of the primary replica. In the second one, replication is mainly used to enhance performance, and serializable consistency is not maintained. Note that most applications can be perfectly run with relaxed consistency modes, or isolation levels. Indeed, the default isolation level of most relational DBMSs is not "serializable", but "read committed" (for instance, in PostgreSQL). Another sample of lazy *primary copy replication* protocol is the one being described by Daudjee & Salem (2006) for providing the snapshot isolation level.

FUTURE TRENDS

Several emerging lines of research in the database replication field can be distinguished:

a. **Load balancing:** In order to improve performance, one of the key issues consists in balancing the load being received by each server replica. Several approaches have been tried in this research line, considering different parameters that rule the replica assignment: memory usage (Elnikety et al., 2007), transaction lengths (Milán-Franco et al., 2004), transaction scheduling (Amza et al., 2005), etc. One of the promising approaches in load balancing is the distribution of transactions based on their conflicts. Thus, inter-conflicting transactions are placed in the same node, reducing their probability of abortion, since they will be managed by a local (and pessimistic) concurrency control mechanism. Additionally, non-conflicting transactions can be placed on lightly loaded replicas. The combination of both approaches has generated good results in (Zuikeviciute & Pedone, 2008).
b. **Support for multiple isolation levels:** Current database replication protocols have focused on supporting a single isolation level, but DBMSs have always supported multiple levels. This allows the application designer to select the most appropriate level for each transaction, improving their response time, since transactions that use relaxed isolation levels seldom get blocked. Our groups have published some initial works in this area (Bernabé-Gisbert et al., 2006; Armendáriz-Íñigo et al., 2007).
c. **scalability:** Although there have been some interesting results (Patiño-Martínez et al., 2005) improving the scalability of data replication systems, further work is still needed in this area.

CONCLUSION

Database replication had commonly used lazy protocols in commercial DBMSs, ensuring thus good performance, but without guaranteeing full replica consistency. This may lead to some transaction losses in case of failure. To overcome these problems, eager replication protocols with group communication support have been proposed in the last decade. The use of total order broadcast protocols has allowed the development of new kinds of eager replication: weak-voting and certification-based techniques. Such solutions are able to ensure one-copy serializability with a performance similar to lazy protocols, with better (although still limited) scalability and lower abort rates. However, these solutions are not applied to commercial database management systems, yet.

In this survey, we have only dealt with fully replicated databases. In future, the partial replication of databases can be expected to become an important research area too. Current research trends in database replication focus on scalability. In addition to that, we suggest that also the simultaneous support of multiple isolation levels should be a research topic worth targeting.

REFERENCES

Amza, C., Cox, A. L., & Zwaenepoel, W. (2005). A Comparative Evaluation of Transparent Scaling Techniques for Dynamic Content Servers. *International Conference on Data Engineering,* (pp. 230-241).

Armendáriz-Íñigo, J. E., Juárez, J. R., González de Mendívil, J. R., Decker, H., & Muñoz-Escoí, F. D. (2007). k-Bound GSI: A Flexible Database Replication Protocol. *ACM Annual Symposium on Applied Computing,* (pp. 556-560).

Berenson, H., Bernstein, P. A., Gray, J., Melton, J., O'Neil, E. J., & O'Neil, P. E. (1995). A Critique of ANSI SQL Isolation Levels. *ACM SIGMOD Conference,* (pp. 1-10).

Bernabé-Gisbert, J. M., Salinas-Monteagudo, R., Irún-Briz, L., & Muñoz-Escoí, F. D. (2006). Managing Multiple Isolation Levels in Middleware Database Replication Protocols. *Lecture Notes in Computer Science, 4330,* 710-723.

Bernstein, P. A., & Goodman, N. (1981). Concurrency Control for Distributed Database Systems. *ACM Computing Surveys, 13*(2), 185-221.

Birman, K. P., & Joseph, T. A. (1987). Reliable Communication in the Presence of Failures. *ACM Transactions on Computer Systems, 5*(1), 47-76.

Breton, R. (1998). Replication Strategies for High Availability and Disaster Recovery. *IEEE Bulletin of the Technical Committee on Data Engineering, 21*(4), 38-43.

Daudjee, K., & Salem, K. (2006). Lazy Database Replication with Snapshot Isolation. *International Conference on Very Large Data Bases,* (pp. 715-726).

Elnikety, S., Zwaenepoel, W., & Pedone, F. (2005). Database Replication Using Generalized Snapshot Isolation. *Symposium on Reliable Distributed Systems,* (pp. 73-84).

Elnikety, S., Dropsho, S., & Zwaenepoel, W. (2007). Tashkent+: Memory-aware Load Balancing and Update Filtering in Replicated Databases. *EuroSys Conference,* (pp. 399-412).

Gifford, D. K. (1979). Weighted Voting for Replicated Data. *7th ACM Symposium on Operating System Principles,* (pp. 150-162).

Gray, J., Helland, P., O'Neil, P., & Shasha, D. (1996). The Dangers of Replication and a Solution. *ACM SIGMOD Conference,* (pp. 173-182).

Irún, L., Muñoz, F. D., & Bernabéu, J. M. (2003). An Improved Optimistic and Fault-Tolerant Replication Protocol. *Lecture Notes in Computer Science, 2822,* 188-200.

Jajodia, S., & Mutchler, D. (1990). Dynamic Voting Algorithms for Maintaining the Consistency of a Replicated Database. *ACM Transactions on Database Systems, 15*(2), 230-280.

Kemme, B., & Alonso, G. (2000). A New Approach to Developing and Implementing Eager Database Replication Protocols. *ACM Transactions on Database Systems, 25*(3), 333-379.

Lin, Y., Kemme, B., Patiño-Martínez, M., & Jiménez-Peris, R. (2005). Middleware-Based Data Replication providing Snapshot Isolation. *ACM SIGMOD Conference,* (pp. 419-430).

Milán-Franco, J. M., Jiménez-Peris, R., Patiño-Martínez, M., & Kemme, B. (2004). Adaptive Middleware for Data Replication. *International Conference on Middleware,* (pp. 175-194).

Muñoz-Escoí, F. D., Pla-Civera, J., Ruiz-Fuertes, M. I., Irún-Briz, L., Decker, H., Armendáriz-Íñigo, J. E., & González de Mendívil, J. R. (2006). Managing Transaction Conflicts in Middleware-based Database Replication Architectures. *Symposium on Reliable Distributed Systems,* (pp. 401-410).

Patiño-Martínez, M., Jiménez-Peris, R., Kemme, B., & Alonso, G. (2005). MIDDLE-R: Consistent Database Replication at the Middleware Level. *ACM Transactions on Computer Systems, 23*(4), 375-423.

Pedone, F., Guerraoui, R., & Schiper, A. (2003). The Database State Machine Approach. *Distributed and Parallel Databases, 14,* 71-98.

Thomas, R. H. (1979). A Majority Consensus Approach to Concurrency Control for Multiple Copy Databases. *ACM Transactions on Database Systems, 4*(2), 180-209.

Wiesmann, M., Pedone, F., Schiper, A., Kemme, B., & Alonso, G. (2000). Database Replication Techniques: A Three-Parameter Classification. *Symposium on Reliable Distributed Systems,* (pp. 206-215).

Wiesmann, M., & Schiper, A. (2005). Comparison of Database Replication Techniques Based on Total Order Broadcast. *IEEE Transactions*

on Knowledge and Data Engineering, 17(4), 206-215.

Zuikeviciute, V., & Pedone, F. (2008). Conflict-Aware Load-Balancing Techniques for Database Replication. *ACM Symposium on Applied Computing,* (pp. 2169-2173).

KEY TERMS

Delegate Replica: In replication models that do not follow the primary copy technique, the delegate replica directly processes a transaction, generating the updates that later will be transmitted to the other replicas. Different transactions can use different delegate replicas.

Primary Replica: In the passive replication model, or primary copy technique, the single replica that is able to directly manage all database transactions.

Readset: The set of data items read by a given transaction.

Reliable Broadcast: One that requires that all correct processes deliver the same set of messages, and such a set includes all messages broadcast by correct processes, and no spurious messages.

Secondary Replica: In the passive replication model, or primary copy technique, all non-primary replicas are also known as secondary replicas. Such replicas can not accept user transactions, and only behave as backup copies of the primary one.

Total Order Broadcast: One that requires that all correct processes deliver all messages in the same order; i.e., a reliable broadcast with total order.

Writeset: The set of data items written by a given transaction.

Chapter LXXXII
A Novel Crash Recovery Scheme for Distributed Real-Time Databases

Yingyuan Xiao
Tianjin University of Technology, China

INTRODUCTION

Recently, the demand for real-time data services has been increasing (Aslinger & Son, 2005). Many applications such as online stock trading, agile manufacturing, traffic control, target tracking, network management, and so forth, require the support of a distributed real-time database system (DRTDBS). Typically, these applications need predictable response time, and they often have to process various kinds of queries in a timely fashion. A DRTDBS is defined as a distributed database system within which transactions and data have timing characteristics or explicit timing constraints and system correctness that depend not only on the logic results but also on the time at which the logic results are produced. Similar to conventional real-time systems, transactions in DRTDBSs are usually associated with timing constraints. On the other hand, a DRTDBS must maintain databases for useful information, support the manipulation of the databases, and process transactions. Timing constraints of transactions in a DRTDBS are typically specified in the form of deadlines that require a transaction to be completed by a specified time. For soft real-time transactions, failure to meet a deadline can cause the results to lose their value, and for firm or hard real-time transactions, a result produced too late may be useless or harmful. DRTDBSs often process both temporal data that lose validity after their period of validity and persistent data that remain valid regardless of time. In order to meet the timing constraints of transactions and

data, DRTDBSs usually adopt main memory database (MMDB) as their ground support. In an MMDB, "working copy" of a database is placed in the main memory, and a "secondary copy" of the database on disks serves as backup. Data I/O can be eliminated during a transaction execution by adopting an MMDB so that a substantial performance improvement can be achieved. We define a DRTDBS integrating MMDB as a distributed real-time main memory database system (DRTMMDBS).

The existing researches on DRTDBS focus mainly on concurrency control (Gustafsson, Hallqvist & Hansson, 2005; Lam, Kuo, Tsang & Law, 2000; Ulusoy, 1993), replication (Aslinger & Son, 2005; Son & Kouloumbis, 1993; Ulusoy, 1994), and commitment (Haritsa, Ramamritham & Gupta, 2000; Qin & Liu, 2003; Xiao, Liu, Deng & Liao, 2006). The studies of failure recovery for a DRTMMDBS are relatively scarce. However, due to the complexity of distributed environments together with volatility and vulnerability of the main memory, the possibility of failure in a DRTMMDBS becomes much larger than in centralized disk resident database systems. When a site crash occurs, as the databases are not available for transactions, many transactions may miss deadlines, and a large amount of temporal data may lose their validity before they can be used, so a DRTMMDBS should have the ability of high fault-tolerance and can resume services again as quickly as possible after crashes.

BACKGROUND

The traditional sequential logging and disk-based recovery techniques cannot meet the high performance requirement of a DRTMMDBS. To aim at real-time databases, some failure recovery methods have been put forward. Choi, et al. (2000) presented a parallel processing architecture adopting double-CPU. In this architecture, one CPU, which is called a DP, is responsible for usual transaction processing, while another CPU, called an RP, is only responsible for recovery processing such as logging, checkpoint, and failure recovery. This architecture has better performances than the traditional single CPU, but the utilization of RP is not high. Sivasankaran, Ramamritham, Stankovic, and Towsley (1995) analyzed the characteristics of data in real-time databases and discussed the strategies of logging and recovery in real-time active databases. To solve the problem of low efficiency of the sequential permanent logging, partitioned logging and ephemeral logging have been proposed, respectively (Agrawal & Jagadish, 1989; Lam & Kuo, 2001). Partitioned logging stores log records according to transaction class (data class); that is, the log records belonging to a different transaction class (data class) are stored in different partitions. Partitioned logging can avoid the performance bottleneck caused by severe contention for single logging store partition. However, the problems such as how to classify transactions (data), how to standardize logging partitions, and how to recover database systems to the correct and consistent state after failures must be solved for partitioned logging. Ephemeral logging does not have to keep log records permanently. For ephemeral logging, when a transaction commits, the data buffers modified by the transaction need to be flushed into stable storage devices. So the log records may be deleted as soon as the corresponding transactions commit. The advantage of ephemeral logging is that the log processing time after failures is reduced prominently, while the disadvantages lie with the lack of permanent logs and inconvenience in auditing and tracing. Liu, Amman, and Jajodia (2000) gave the technique of accelerating recovery speed, but the technique still requires stopping the system services in entire recovery process. For MMDBs, shadow paging recovery schemes have been proposed (Kim & Park, 1996). The shadow paging recovery scheme does not require writing up a log, but it requires a large amount of main memory space. Woo, Kim, and Lee (1997)

proposed a kind of method to combine logging with a shadow paging scheme, but the method still requires a great deal of time to deal with log records. For distributed database systems, the researches on failure recovery mainly focus on the recovery processing of distributed transaction failure and the backup strategies. When a transaction failure occurs in distributed database systems, the recovery method based on checkpoint (Kim & Park, 1993; Wang & Fuchs, 1993) is usually adopted to decide how to rollback and restart the execution of the failure transaction. For media failures, the consistent backup (consistent global checkpointing) strategies were presented (Kim & Park, 1994; Son & Agrawala, 1989).

For purposes of recovery processing, a DRTMMDBS must deal with different types of failures, such as communication failure, transaction failure, system failure, and media failure. In this chapter, we concentrate on system failure, also called crash, that occurs when the contents of volatile main memory storage are lost or corrupted.

This chapter presents a real-time dynamic crash recovery scheme (RTDCRS) suitable for a DRTMMDBS on the basis of considering carefully the timing constraint characteristics of data and transaction. The rest of the chapter is organized as follows: Section 3 first analyzes the recovery requirements of a DRTMMDBS and then gives the recovery correctness criteria suitable for a DRTMMDBS. In Section 4, we propose an RTDCRS and prove its correctness. Section 5 gives performance evaluation of a RTDCRS. In Section 6, we conclude the proposed RTDCRS. Finally, in Section 7, we discuss future and emerging trends.

RECOVERY REQUIREMENTS AND CORRECTNESS CRITERIA

Before analyzing the failure recovery requirements of a DRTMMDBS, we first give the following definitions.

Definition 1: Temporal data object X is defined as the following 3-tuples:

$<V(X), ST(X), VI(X)>$

where $V(X)$ denotes the current state or value of X, $ST(X)$ is sampling time scale (i.e. the time of sampling the value of data object X), and $VI(X)$ is the period of validity of X.

Definition 2: A data object X is said to satisfy the temporal consistency, iff $ST(X)+VI(X) \geq T_c$. Here, T_c denotes the current instant.

For example, Assume the value of temporal data object X is x_0 (Suppose $x_0 = 8$) at the latest sampling time scale $ST(X)$ (Suppose $ST(X) = 1000$ seconds; here, we assume time scale is uniformly transformed into relative time). Assume the period of validity of X is 10 seconds (i.e., $VI(X) = 10$ seconds). Then we signify X with <8, 1000, 10>. When the current instant T_c does not exceed 1010, the value of X (i.e., $x_0 = 8$) is valid. Once $T_c > 1010$ (i.e., $ST(X)+VI(X) < T_c$), the value of X (i.e., $x_0 = 8$) loses validity (i.e., the value of X is overdue). The overdue data will be refreshed by triggering the corresponding sample transaction.

Comparing with traditional database systems, a DRTMMDBS has the following new recovery requirements after failures:

(1) Recovery schemes must consider the timing constraints and updating frequency of data (i.e., the data with an urgent period of validity), and hot data with high access frequency should be recovered beforehand.
(2) Recovery schemes must take into account the timing constraints of transactions as well, such as data accessed by transactions with high priority should be recovered beforehand.
(3) Besides logical consistency, the temporal consistency of data must be guaranteed as

well. The overdue data need to be restored by sampling the corresponding objects in the physical world over again.

(4) Besides recovering the state of the database, the state of real-world changed by uncommitted transactions should be restored through "compensating" or "alternate" transactions.

(5) Recovery processing must be combined with a corresponding commit protocol to guarantee failure atomicity of distributed real-time transactions.

(6) Recovery processing requires considering the characteristics of MMDB.

(7) Speedy logging and recovery.

Considering the preceding recovery requirements, we propose the following recovery correctness criteria of a DRTMMDBS.

Criterion 1: (Temporal data recovery criterion) $\forall X \in$ TDS, if $ST(X) + VI(X) \leq T_c$, UNDO and REDO recovery operations of aiming at X are not necessary after failures. These overdue data need to be restored by sampling the corresponding object in the physical world. Here, TDS denotes the set of temporal data objects.

A DRTMMDBS interacts with the physical world directly by two pathways. One pathway is that information about the physical world state or events is written into databases, and another is that transactions running in database systems may trigger the activities that can change the physical world state. We term the transactions that trigger the activities of changing the physical world state as control transactions.

We use A_T to denote the activity triggered by a control transaction T, and CA_T to denote the compensative or alternative transaction of T.

Criterion 2: (Physical world state recovery criterion). If a control transaction T does not commit successfully and at the same time A_T has happened, the physical world states changed by A_T must be restored through executing CA_T.

Suppose a distributed real-time transaction $T=\{st_1, st_2, \ldots st_m\}$, where st_i $(1 \leq i \leq m)$ stands for a subtransaction of T executed at site *i*.

Criterion 3: (Distributed real-time transaction REDO recovery criterion). At the failure instant, if T has committed, at failure site *i*, REDO operations are required to confirm the effect of st_i exception to the following two conditions:

(1) The values of data objects updated by st_i have been written into disk databases physically.

(2) The data objects updated by st_i meet the condition: $ST(X) + VI(X) \leq T_c$.

Criterion 4: (Distributed real-time transaction UNDO recovery criterion). At the failure instant, if T has not committed, for each st_i $(1 \leq i \leq m)$ at the corresponding site, UNDO operations are required to erase the effect of st_i exception to the following two conditions:

(1) The values of data objects updated by st_i are not written into disk databases physically.

(2) The data objects updated by st_i meet the condition: $ST(X) + VI(X) \leq T_c$.

REAL-TIME DYNAMIC CRASH RECOVERY SCHEME (RTDCRS)

Real-Time Logging of RTDCRS

The traditional recovery schemes adopt sequential permanent logging based on disks. Obviously, the logging technique has not met the high per-

formance requirements of DRTMMDBS. In this chapter, we propose a novel real-time logging scheme based on nonvolatile RAM (NRPERTL, for short), which integrates the properties of partitioned logging and ephemeral logging. The two aspects show the difference between real-time logging and traditional logging. The first aspect finds expression in the types and structure of log record. Real-time logging adds "Compensate log record," which is used to restore the state of real world changed by control transactions. In addition, real-time logging extends the structure of the log record by adding time fields of "*TS*" and "*VTI*." The second difference between real-time logging and traditional logging is logging processing strategy after crashes. For the overdue data, real-time logging does not execute REDO recovery but triggers the corresponding sample transaction to update overdue data.

Transaction Classification Based on NRPERTL

The sequential permanent logging based on disks stores log records into a single disk-logging file so the tail of the logging file becomes a "hot spot" because of severe contention for it and becomes the bottleneck of high performance. Therefore, NRPERTL adopts the idea of logging partitions, (i.e., logging storage area is divided into multiple partitions and log records belonging to different transaction classes are stored in different partitions, respectively). Thus, the division of transaction classes decides the logging partitions in NRPERTL. In a DRTMMDBS, transactions are abounding in semantic characteristics and may be classified according to their different characteristics. In this chapter, we classify transactions according to the requirement of logging partitions.

In the following definitions, ST denotes the set of possible transactions in a DRTMMDBS; DS(ST) denotes the set of data objects accessed by the transactions belonging to ST; LS denotes the entire logging storage area; LS_i $(1 \le i \le n)$ denotes a partition of LS; and SA(T) denotes the storage area of the log records belonging to transaction T.

Definition 3: Let $ST_i, ST_j \subseteq ST$. If $ST_i \bigcap ST_j = \phi$, then ST_i and ST_j are said to be disjoint, notated by $ST_i \perp ST_j$.

Here, the formula, $ST_i \bigcap ST_j = \phi$, denotes that ST_i and ST_j are two disjoint transaction sets. That is, $\neg \exists T ((T \in ST_i) \wedge (T \in ST_j))$.

Definition 4: Let $B = \{ST_1, ST_2, \ldots, ST_n\}$, $ST_i \subseteq ST$ $(1 \le i \le n)$. If $\forall ST_i, ST_j \in B$ $(ST_i \perp ST_j)$, then B is said to be a disjoint set.

Definition 5: Let $B = \{ST_1, ST_2, \ldots, ST_n\}$ be a disjoint set. If $ST_1 \bigcup ST_2 \bigcup \ldots \bigcup ST_n = ST$, then B is said to be a division of ST, and ST_i $(1 \le i \le n)$ is said to be a transaction class.

Definition 6: Let $L = \{LS_1, LS_2, \ldots, LS_n\}$. If the condition $(\forall LS_i, LS_j \in L\ (LS_i \bigcap LS_j = \phi)) \wedge (LS_1 \bigcup LS_2 \bigcup \ldots \bigcup LS_n) = LS)$ is held, then L is said to be a division of LS.

Here, the formula, $LS_i \bigcap LS_j = \phi$, means that LS_i and LS_j are two disjoint logging storage partition.

Definition 7: Let $\pi = \{ST_1, ST_2, \ldots, ST_n\}$ be a division of ST, $\mu = \{LS_1, LS_2, \ldots, LS_n\}$ be a division of LS. If there exists a one-to-one mapping $f : \pi \rightarrow \mu$, then μ is said to be a division of LS relating to π, notated by $\mu \prec \pi$, and LS_i $(1 \le i \le n)$ is said to be a logging partition. Here, *f* denotes for any $ST_i \in \pi$, existing one and only $LS_i \in \mu$ makes $\forall T \in ST_i$ $(SA(T) \subseteq LS_i)$ hold, and vice versa.

Definition 8: Let $B = \{ST_1, ST_2, \ldots, ST_n\}$ be a division of ST, if $\forall ST_i, ST_j \in B$ $(DS(ST_i) \bigcap DS(ST_j) = \phi)$, then B is said to be an orthogonal division, notated by $B \equiv \perp_{division}$.

In a real-time system, data and transaction are abundant in semantic characteristics and can be divided according to the different attributes. For example, data can be broadly classified into *hot* and *cold*, depending on the type of data (e.g., critical, temporal, etc.), and the type of the transactions (e.g., high priority, low priority, etc.), which accesses the data. Usually, critical transactions only access critical data in real-time systems.

Due to adopting the idea of classification recovery (introduced next), the difference in the single partitioned logging technique that has no special need for the division of transaction class is that RTDCRS requires the division of the transaction class to satisfy the following rule in order to guarantee recover databases to correct and consistent states after failures.

Rule 1: (transaction classification rule). To guarantee recovery correctness after failures, RTDCRS requires the division $\pi = \{ST_1, ST_2, \ldots, ST_n\}$ related to the division of LS to meet the following condition: $\exists\, \pi_1 \subset \pi\, (DS(\pi_1) \bigcap DS(\pi - \pi_1) = \phi)$; i.e. $\{\pi_1, (\pi - \pi_1)\} \equiv \perp_{division}$.

If a division of ST meets rule 1, then it is said to be a qualified division. There exists the following theorem:

Theorem 1: If $\{\pi_1, \pi_2\} \equiv \perp_{division}$, $\{ST_{k1}, ST_{k2}, \ldots, ST_{kj}\}$ is a division of π_1 and $\{ST_{m1}, ST_{m2}, \ldots, ST_{mi}\}$ is a division of π_2, then $\{ST_{k1}, ST_{k2}, \ldots, ST_{kj}, ST_{m1}, ST_{m2}, \ldots, ST_{mi}\}$ is a qualified division.

Proof: Because of $\{\pi_1, \pi_2\} \equiv \perp_{division}$, according to rule 1, to prove the correctness of theorem 1, we only require proving the correctness of $\{ST_{k1}, ST_{k2}, \ldots, ST_{kj}, ST_{m1}, ST_{m2}, \ldots, ST_{mi}\}$ being a division of ST. Since $\{ST_{k1}, ST_{k2}, \ldots, ST_{kj}\}$ is a division of π_1, $\forall\, ST_{kp}, ST_{kq} \in \{ST_{k1}, ST_{k2}, \ldots, ST_{kj}\}(ST_{kp} \bigcap ST_{kq} = \phi)$ and $ST_{k1} \bigcup ST_{k2} \bigcup \ldots \bigcup ST_{kj} = \pi_1$. Similarly, because $\{ST_{m1}, ST_{m2}, \ldots, ST_{mi}\}$ is a division of π_2, $\forall\, ST_{mp}, ST_{mq} \in \{ST_{m1}, ST_{m2}, \ldots, ST_{mi}\}(ST_{mp} \bigcap ST_{mq} = \phi)$ and $ST_{m1} \bigcup ST_{m2} \bigcup \ldots \bigcup ST_{mi} = \pi_2$. Further, since $\{\pi_1, \pi_2\} \equiv \perp_{division}$, $\pi_1 \bigcup \pi_2 = ST$ and $\pi_1 \bigcap \pi_2 = \phi$; thus, we have $ST_{k1} \bigcup ST_{k2} \bigcup \ldots \bigcup ST_{kj} \bigcup ST_{m1} \bigcup ST_{m2} \bigcup \ldots \bigcup ST_{mi} = ST$ and $\forall\, ST_{kp}, ST_{kq} \in \{ST_{k1}, ST_{k2}, \ldots, ST_{kj}, ST_{m1}, ST_{m2}, \ldots, ST_{mi}\}(ST_{kp} \bigcap ST_{kq} = \pi)$, i.e., $\{ST_{k1}, ST_{k2}, \ldots, ST_{kj}, ST_{m1}, ST_{m2}, \ldots, ST_{mi}\}$ is a division of ST. Therefore, $\{ST_{k1}, ST_{k2}, \ldots, ST_{kj}, ST_{m1}, ST_{m2}, \ldots, ST_{mi}\}$ is a qualified division.

According to theorem 1, we may easily construct a qualified division by means of the following method. First, we divide ST into two disjoint subsets: $ST_{>p} = \{T_i \mid Pr(T_i) > p, T_i \in ST\}$ and $ST_{<=p} = \{T_i \mid Pr(T_i) \le p, T_i \in ST\}$, where $Pr(T_i)$ denotes the priority of transaction T_i. Obviously, $\{ST_{<=p}, ST_{<=p}\}$ is a division of ST. Then, by means of adjusting the value of *p*, we guarantee the conditions, $DS(ST_{>p}) \bigcap DS(ST_{<=p}) = \phi$, is held; i.e., $\{ST_{>p}, ST_{<=p}\} \equiv \perp_{division}$. $DS(ST_{>p})$ accessed by the high priority transaction belonging to $ST_{>p}$ is said to be a critical data class, and $DS(ST_{<=p})$ accessed by the low priority transaction belonging to $ST_{<=p}$ is said to be an ordinary data class. Further, we divide $ST_{>p}$ into two (or multiple) subsets: $ST_{>p, >f} = \{T_i \mid Fr(T_i) > f, T_i \in ST_{>p}\}$ and $ST_{>p, <=f} = \{T_i \mid Fr(T_i) \le f, T_i \in ST_{>p}\}$, where $Fr(T_i)$ denotes the estimated execution frequency of T_i. Obviously, $\{ST_{>p, >f}, ST_{>p, <=f}\}$ is a division of $ST_{>p}$. Similarly, $ST_{<=p}$ is also divided into two (or multiple) subsets: $ST_{<=p, >f} = \{T_i \mid Fr(T_i) > f, T_i \in ST_{<=p}\}$ and $ST_{<=p, <=f} = \{T_i \mid Fr(T_i) \le f, T_i \in ST_{<=p}\}$. Obviously, $\{ST_{<=p, >f}, ST_{<=p, <=f}\}$ is a division of $ST_{<=p}$. Let $\sigma = \{ST_{>p, >f}, ST_{>p, <=f}, ST_{<=p, >f}, ST_{<=p, <=f}\}$, by means of theorem 1, σ is a qualified division.

Storage Media Organization Based on NRPERTL

Storage media at each site consist of three tiers: disk storage, nonvolatile RAM, and volatile main memory, as shown in Figure 1. Local disk resident database (LDDB), which acts as "sec-

ondary copy" of a database, is stored on disks. Nonvolatile RAM serves as logging storage area, and entire logging storage area is divided into four independent partitions: critical active logging partition, critical dull logging partition, ordinary active logging partition, and ordinary dull logging partition (CALP, CDLP, OALP, and ODLP for short, respectively), according to the qualified division σ. CALP, CDLP, OALP, and ODLP are used for storing the corresponding log records of $ST_{>p, >f}$, $ST_{>p, <=f}$, $ST_{<=p, >f}$, and $ST_{<=p, <=f}$, respectively; i.e., $\{ST_{>p, >f}, ST_{>p, <=f}, ST_{<=p, >f}, ST_{<=p, <=f}\} \prec \{CALP, CDLP, OALP, ODLP\}$. We call CALP and CDLP uniformly as critical logging partition, and OALP and ODLP uniformly as ordinary logging partition. Local main memory database (LMDB) is placed in the volatile main memory, which is divided into two partitions: critical data partition and ordinary data partition, which store critical data class and ordinary data class, respectively. CHLP denotes checkpointing logging partition. During the local checkpointing process, checkpointing log record is written into CHLP. In the following text, we will elaborate the local checkpoint scheme.

The Types and Structure of Real-Time Log Record

Updating strategies of database (including delayed updating and immediate updating) influence on log processing. Immediate updating is likely to produce dirty data, so it may result in cascading abort, which has a very large effect on the system performance. Therefore, in a DRTMMDBS, delayed updating strategy is usually adopted. For delayed updating, the values of data objects updated by a transaction are not written into LMDB until the transaction commits, so UNDO operations are no longer required after failures. In this chapter, we assume that the delayed updating strategy is adopted.

In combination with the real-time commit protocol 1PRCP (Xiao et al., 2006), we design five kinds of log record for NRPERTL; namely, Begin, Redo, Compensate, Ready, and Commit. Figure 2 describes their structures.

In Figure 2, *P-TID* denotes the identifier of a distributed transaction (global transaction) and is defined as follows: *P-TID* = (*Cor-Adrr*, *SN*), where *Cor-Adrr* stands for the network address of coordinator of the distributed transaction, and *SN* denotes the serial number of the distributed transaction that is exclusive at the coordinator.

Figure 1. Storage media organization at a site

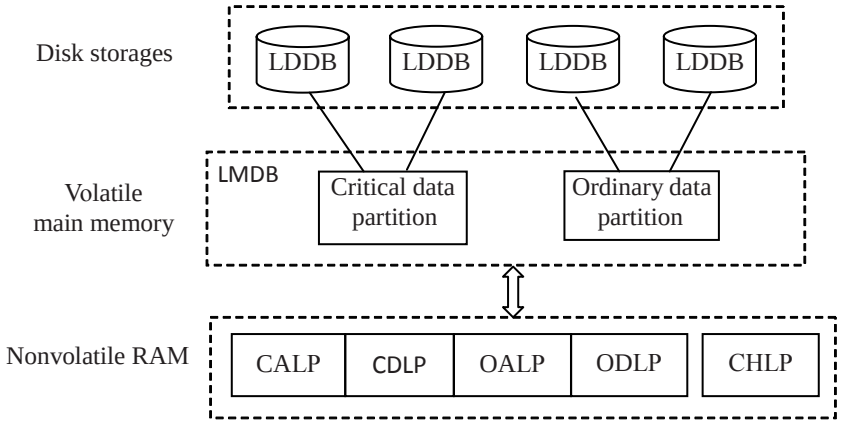
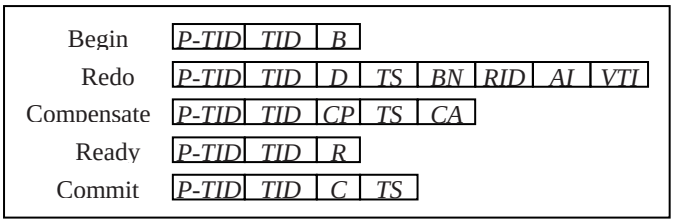


Figure 2. Types and structures of real-time log record



TID denotes the identifier of a subtransaction of *P-TID* and is defined as (*Adrr*, *SN*), where Adrr stands for the network address of the site at which the subtransaction is executed and SN denotes the exclusive serial number of the subtransation. For a local transaction, *P-TID* is set as null.

"*B*," "*D*," "*CP*," "*R*," and "*C*" are used for labeling the log record types of Begin, Redo, Compensate, Ready, and Commit, respectively. *TS* denotes the logic timestamp when the Redo or Compensate or Commit log record are created, and here, we denote the logical timestamp at each site as a sequence number that starts at 0 and is incremented by 1 each time a new Redo or Compensate or Commit log record is written. *RID* stands for the identifier of the updated data object. *BN* denotes the logic number of the disk block that is the backup of the data page, which is in LMDB and contains the updated data object. *AI* denotes the after-image of the updated data object. *VTI* denotes the validity instant of the temporal data object; namely, $VTI = ST(X)+VI(X)$. For persistent data objects, *VTI* is set as infinite. *CA* denotes the compensating activity of the control transaction.

When a transaction begins to execute, at its executing site, a Begin log record is created in the corresponding logging partitions; for each updating operation, the corresponding Redo log record is created; when a control transaction sends the message to a controlled subsystem, a corresponding Compensate log record is created; at a participant site, when a subtransaction comes into the Prepare Commit State and does not miss its deadline, a Ready log record is created at the participant site; when the coordinator of a distributed real-time transaction receives "Vote-Commit" messages from all participants in the limited time, the corresponding Commit log record is created at the coordinator site; when a subtransaction receives "Global Commit" decision from the coordinator at its executing site, a commit log record is created.

Local Checkpoint Scheme of RTDCRS

The purpose of checkpoint is to maintain the recent version of databases at a secondary storage device, to determine the starting point of recovery processing, and to limit recovery time. In RTDCRS, each site executes the local checkpointing process independently. During the local checkpointing process, the updating of LMDB is written out to LDDB, some useless log records are deleted, and the corresponding logging store area is freed. Checkpointing scheme (i.e., static and fuzzy) and executing frequency have large effects on the system performance. In RTDCRS, local checkpoint adopts a fuzzy checkpointing scheme, which allows the existence of active transaction during the checkpointing process so it can improve the performance of the system. With respect to the triggering mode of checkpoint, we

do not adopt periodic triggering based on fixed time interval but rather decide whether to trigger the local checkpoint or not according to the utilization rate of the logging storage area R_{LU}. Only if $R_{LU} > \alpha$, is local checkpoint triggered, where α is the threshold of R_{LU}, which can be adjusted dynamically according to the application requirements.

During the local checkpointing process, checkpointing log record is written into checkpointing logging partition, which resides in nonvolatile RAM. Checkpointing log record includes five fields: checkpointing bit field (*CKB*), critical transaction class recovery start timestamp (*CTRST*), ordinary transaction class recovery start timestamp (*OTRST*), checkpointing timestamp (*CKT*), and updating page field. *CKB* denotes whether this local checkpoint is completed successfully or not, and 1 stands for successful completion, while 0 represents crash happening during this local checkpointing process. *CTRST* denotes the minimal timestamp of critical transaction class that requires executing REDO recovery after a crash. *OTRST* denotes the minimal timestamp of ordinary transaction class that requires executing REDO recovery after a crash. *CKT* stands for the logic timestamp when this local checkpoint is triggered. Updating page field sets a bit for each page of LMDB to denote updating state of the data page, and 1 denotes the page is updated since the last local checkpoint, while 0 denotes the page is not updated yet. The local checkpointing procedure is described in Exhibit A.

In the above description and following recovery algorithm, *TCounter* denotes the counter of logic timestamp, which resides in nonvolatile RAM and records the current value of logic timestamp. *TCounter* starts at 0 and is incremented by 1 each time a new Redo or Compensate or Commit or checkpointing log record is created.

Dynamic Recovery Processing Strategy

During recovery processing after a crash, first LDDB is loaded into the main memory to reconstruct LMDB, and then the recovery subsystem is responsible for restoring LMDB to recent consistent state. In order to improve the system performance, RTDCRS adopts the dynamic recovery-processing strategy based on the idea of classification recovery, which supports concurrent of system services and recovery processing. The strategy is based on NRPERTL, and its key characteristic is critical data class first recovered, and then system services are brought back before ordinary data class is recovered; that is, the dynamic recovery supports the concurrent of system services and the recovery processing of ordinary data class. In combination with the real-time commit protocol 1PRCP, the dynamic recovery-processing strategy can guarantee failure atomicity of distributed real-time transactions. In detail, the dynamic recovery-processing strategy consists of the steps as follows:

(1) Reload critical data class into main memory to reconstruct LMDB.
(2) Recover critical data class to consistent state and eliminate the effects caused by uncommitted critical control transactions.
(3) Restore the system services of failure sites.
(4) Reload ordinary data class into LMDB.
(5) Recover ordinary data class to consistent state and eliminate the effects caused by uncommitted ordinary control transactions.

In these steps, the realization algorithm of step (2) and step (5) is described as in Exhibit B.

In Exhibit B, $RedoLog_{Ti,\ k}$ denotes the *k*th Redo log record of transaction T_i; the procedure, $REDO(RedoLog_{Ti,k})$, realizes the function, which restores the value of the data object whose *RID* is

recorded in $RedoLog_{Ti,k}$ by using its *AI* (after-image); $RedoLog_{Ti,k}.VTI$ denotes the *VTI* of $RedoLog_{Ti,k}$; T_c denotes the current instant.

Correctness of RTDCRS

In the traditional sequential logging scheme, log records are sequentially stored in the single logging file according to the executing order of the operations of the corresponding transactions. For database systems based on the sequential logging scheme, recovery algorithm after a crash scans reversely the logging file to execute UNDO operations first, and then scans forward the logging file to execute REDO operations. The previous logging processing strategy can ensure to restore databases to recent consistent state if only database systems adopt the sequential logging scheme and follow the principle of WAL (Write Ahead Logging). However, due to adopting partitioned logging and the dynamic recovery strategy, RTDCRS is obviously different from the traditional sequential logging scheme. So its correctness must be proved.

Lemma 1: Let $\{ST_1, ST_2\} \equiv \perp_{division}$. If the log records belonging to transaction class ST_1 are sequentially stored in the logging partition LS_1 according to the executing order of the operations of the corresponding transactions and the log records belonging to transaction class ST_2 are sequentially stored in the logging partition LS_2 according to the executing order of the operations of the corresponding transactions, then after a crash, the effect of executing logging processing to LS_1 first and then executing logging processing to LS_2 is equal to the effect of executing logging processing to a single log file.

Proof: Because of $\{ST_1, ST_2\} \equiv \perp_{division}$, $DS(ST_1) \bigcap DS(ST_2) = \phi$ can hold water. Suppose *X* is any data object that requires being recovered by executing UNDO or REDO operations after a crash, then either $X \in DS(ST_1)$ or $X \in DS(ST_2)$. Without loss of generality, assume $X \in DS(ST_1)$ and $\{U_1, U_2, \ldots, U_m\}$ is an update operation sequence for *X* that requires executing UNDO or REDO operations after a crash. Assume the corresponding log record sequence of $\{U_1, U_2, \ldots, U_m\}$ is $\{L_1, L_2, \ldots, L_m\}$. Obviously, $\{L_1, L_2, \ldots, L_m\}$ is stored in LS_1 according to the condition of lemma; therefore, the value of *X* of executing logging processing to LS_1 first and then executing logging processing to LS_2 is equal to the value of *X* of executing logging processing to a single log file in which $\{L_1, L_2, \ldots, L_m\}$ is stored. Thus, lemma is proved.

Exhibit A.

Procedure *LocalCheckpoint()*
1: *CKB = 0, CKT = ++TCounter;*
2: *Write data pages updated by committed transaction out to LDDB*;
3: *Set updating page field according to the updating state of data page;*
4: *Write the data pages whose updating bits are 1 out to LDDB;*
5: *Delete useless log records, i.e. the log records of aborted transactions or whose timestamp in Commit log records is not larger than CKT, and free the corresponding logging store spaces;*
6: *Get the minimal logging timestamp of critical transaction class, set the value of CTRST;*
7: *Get the minimal logging timestamp of ordinary transaction class, set the value of OTRST;*
8: *CKB = 1;*

Exhibit B.

Procedure *DyCrashRecovery(char dc)*
Input: *dc denotes the data class asking to be recovered, and dc="C" stands for critical data class, while dc="O" stands for ordinary data class.*
1: *FT = TCounter*;
2: ***if*** *(dc="C")* ***then***
3: *RST = CTRST;*
4: *Scan the critical logging partition (including CALP and CDLP) to look for all Compensate log records* $CPLog_{Ti}$*, which satisfy the following conditions: the corresponding Commit log records* $CommitLog_{Ti}$ *do not exist, i.e. the corresponding transactions do not commit successfully, and timestamp of* $CPLog_{Ti}$ *is smaller than FT. At the same time insert these* $CPLog_{Ti}$ *into compensating activity recovery list (CARL) in order of their timestamp;*
5: ***else***
6: *RST = OTRST;*
7: *Scan the ordinary logging partition (including OALP and ODLP) to look for all Compensate log records* $CPLog_{Ti}$*, which satisfy the following conditions: the corresponding Commit log records* $CommitLog_{Ti}$ *do not exist, i.e. the corresponding transactions do not commit successfully, and timestamp of* $CPLog_{Ti}$ *is smaller than FT. At the same time insert these* $CPLog_{Ti}$ *into compensating activity recovery list (CARL) in order of their timestamp;*
8: *Scan reversely CARL until the head of CARL and for each* $CPLog_{Ti}$*, execute the corresponding compensating activity* $CPLog_{Ti}.CA$*;*
9: ***while*** *(RST* $\leq$ *FT)*
10: ***if*** *(dc="C")* ***then***
11: *Scan the critical logging partition to look for the Redo log record* $RedoLog_{Ti,\,k}$ *whose timestamp is RST;*
12. ***else***
13: *Scan the ordinary logging partition to look for the Redo log record* $RedoLog_{Ti,\,k}$ *whose timestamp is RST;*
14: ***if*** *(find the Redo log record* $RedoLog_{Ti,\,k}$*)* ***then***
15: ***if*** *(find the corresponding Commit log record* $CommitLog_{Ti}$*)* ***then***
16: ***if*** *(*$RedoLog_{Ti,\,k}.VTI > T_c$*)* ***then***
17: *REDO(*$RedoLog_{Ti,\,k}$*)*
18: *RST ++; continue;*
19: ***else***
20: *RST++;*
21: *Trigger the corresponding sample transaction to update overdue data;*
22: ***if*** *(find the corresponding Ready log record* $ReadyLog_{Ti}$*)* ***then***
23: ***if*** *(receive Commit message)* ***then***
24: *REDO(*$RedoLog_{Ti,\,k}$*);*
25: *Add* $CommitLog_{Ti}$ *to the corresponding logging partition;*
26: *RST++;*
27: ***if*** *(receive Abort message)* ***then***
28: *RST++;*
29: ***else***
30: *RST++;*
31: ***else***
32: *RST++;*
33: ***endwhile***

Theorem 2: RTDCRS can guarantee recovering databases to the recent consistency state (including logic consistency and temporal consistency) after a crash.

Proof:

(1) First, we prove the logic consistency of a database can be guaranteed. In RTDCRS, log records belonging to different transaction classes are stored in different logging partitions. Assume $\pi = \{ST_1, ST_2, \ldots, ST_n\}$ is a qualified division adopted in RTDCRS, then in terms of rule 1, there exist $\pi_1 = ST_{i1} \bigcup ST_{i2} \bigcup \ldots \bigcup ST_{ik}$ and $\pi_2 = \pi - \pi_1$ to satisfy the condition: $\{\pi_1, \pi_2\} \equiv \perp_{division}$, where {i1, i2, …, ik} $\subset$ {1, 2, …, n}. Without loss of generality, assume $\pi_2 = ST_{j1} \bigcup ST_{j2} \bigcup \ldots \bigcup ST_{jm}$, where {j1, j2, …, jm} $\subset$ {1, 2, …, n} and m+k = n. Suppose $L = \{LS_1, LS_2, \ldots, LS_n\}$ is a division of LS relating to π (i.e., $L \prec \pi$.) Without loss of generality, assume $L_1 = (L_{i1} \bigcup L_{i2} \bigcup \ldots \bigcup L_{ik}) \prec \pi_1$, $L_2 = (L_{j1} \bigcup L_{j2} \bigcup \ldots \bigcup L_{jm}) \prec \pi_2$, where $L_1 \bigcup L_2 = L$. The dynamic recovery strategy of RTDCRS first recovers $DS(\pi_1)$ on the basis of the logging partition L_1 and then recovers $DS(\pi_2)$ according to the logging partition L_2. In detail, the recovery processing of $DS(\pi_1)$ is concerned with the logging processing for multiple logging partitions (i.e., L_{i1}, L_{i2}, …, L_{ik}) belonging to L_1. During the logging processing for multiple logging partitions (i.e., L_{i1}, L_{i2}, …, L_{ik}) belonging to L_1, the recovery-processing algorithm ensures the correct executing order of UNDO and REDO on the basis of the timestamps of updating logging records (i.e., for the Redo logging records in L_1), which require being recovered, the REDO operations are sequentially executed according to the increment order of their timestamps and for the Compensate log records of uncommitted control transactions, the corresponding compensating activities are executed according to the degression order of their timestamps to restore the physical world state changed by uncommitted control transactions (due to adopting delayed updating strategy, UNDO operations of update operations to database are not required after a crash). Similarly, during the recovery processing of $DS(\pi_2)$, the logging processing for multiple logging partitions (i.e., L_{j1}, L_{j2}, …, L_{jm}) belonging to L_2 adopts the previous similar method. Therefore, the effect of the logging processing based on the qualified division, $\pi = \{ST_1, ST_2, \ldots, ST_n\}$, is equal to the effect of the logging processing based on the orthogonal division $\{\pi_1, \pi_2\}$. According to lemma 1, the effect of the logging processing based on the qualified division of RTDCRS is equal to the effect of the logging processing based on a single logging file, so RTDCRS can ensure the logic consistency of database.

(2) Second, we prove the temporal consistence of database can be guaranteed. According to our recovery-processing algorithm, for each temporal data object *X* that requires being recovered, if its *VTI* is larger than the current instant T_c (i.e., *X* is not yet overdue in T_c), then *X* is recovered by executing the corresponding REDO operation; if its *VTI* is not larger than T_c (i.e., *X* has already been overdue in T_c), then *X* is updated by triggering the corresponding sample transaction. Therefore, our recovery-processing algorithm can always guarantee the recovered *X* to meet the following condition: $VTI \geq T_c$ (i.e., $ST(X) + VI(X) \geq T_c$). Thus, RTDCRS can guarantee the temporal consistence of the database. To summarize, theorem 2 is proved.

Besides the consistency of databases, RTDCRS also meets the recovery correctness criteria of DRTMMDBS proposed in Section 3.

Theorem 3: RTDCRS meets temporal data recovery criterion.

Proof: Since adopting delayed updating strategy, RTDCRS needs not to execute UNDO recovery for updating operations of uncommitted transactions. For any one data object *X* updated by committed transactions, if *X* has been overdue at the recovery instant (i.e., $ST(X) + VI(X)) \leq T_c$), then according to the dynamic recovery processing algorithm of RTDCRS, *X* is updated by triggering the corresponding sample transaction. Thus, RTDCRS meets the temporal data recovery criterion.

Theorem 4: RTDCRS can guarantee restoring the physical world to consistency state (i.e., meets criterion 2).

Proof: First, RTDCRS adds the type of Compensate log record, which records the corresponding compensating activity. Second, in the dynamic recovery-processing algorithm, for the Compensate log records of uncommitted control transactions, the corresponding compensating activities are executed according to the degression order of their timestamps to restore the physical world state changed by uncommitted control transactions. Thus, RTDCRS can guarantee the consistency of the physical world.

Theorem 5: RTDCRS meets distributed real-time recovery criterion (i.e., meets criterion 3 and criterion 4).

Proof: Since adopting delayed updating strategy, RTDCRS needs not execute UNDO recovery for updating operations of uncommitted transactions. For the data objects updated by the subtransactions of committed transactions, RTDCRS decides on executing REDO or sampling recovery according to these data objects are overdue or not at the recovery instant. For overdue data objects at the recovery instant, RTDCRS does not execute the corresponding REDO recovery but executes sampling recovery. Therefore, RTDCRS meets distributed real-time recovery criterion.

PERFORMANCE TEST AND EVALUATION

The performance of a crash recovery scheme is mainly decided by (1) logging cost in up time and (2) the time of system denying services after crashes (i.e., the time interval from crash instant to the instant of reproviding system service). In this section, we evaluate the performance of RTDCRS through experiments in ARTs-II, a distributed real-time database management prototype system developed by the authors. First, we study how logging overheads affect the system run-time performance and then test how the number of partitions affects the performance of NRPERTL. Last, we evaluate the time of system denying services of RTDCRS after a crash.

In our experiments, global main memory database (GMDB) consists of 50,000 data pages, which are equally allocated to five sites to form the corresponding local main memory databases (LMDBs). The granularity of both data access and concurrency control is data page. Priority-assigning policy adopts Earliest Deadline First (EDF); that is, the priority of a transaction is directly proportional to its deadline. The deadline of a transaction *T* is set as follows: $D(T)=AT(T)+Slack \times ET(T)$, where $D(T)$ denotes the deadline of *T*; $AT(T)$ denotes the arrival time of T, *Slack* denotes a random variable satisfying uniform distribution, and $ET(T)$ denotes estimative executing time of *T*, where $ET(T)=NTO \times AET$. Here, the meanings of *NTO* and *AET* are described in Table 1. The main performance metric used for evaluation is the ratio of transactions missing their deadlines, denoted as MDR. MDR is defined as follows: MDR = (Number of transactions missing their deadlines) /

(Total number of transaction in the system). Main experiment parameters are presented in Table 1, where U[i, j] denotes a uniformly distributed random variable in the range [i, j].

We compare NRPERTL of RTDCRS with the other three kinds of logging schemes: non-logging scheme (NLS), partitioned logging scheme based on disk (PLSD), and sequential logging scheme based on disk (SLSD). NLS denotes there is not the cost of transaction logging in the running period of database system. Obviously, NLS cannot meet the requirement of recovery after failure. Here, NLS is regarded as the baseline of performance. For PLSD, log records belonging to different transaction classes are stored in different disks, respectively.

As shown in Figure 3, the experimental results show when arrival ratio of transaction enhances and when MDR of all logging schemes increases; meanwhile, NRPERTL is closest to NLS and obviously has an advantage over another two kinds of schemes. The main reason is that NRPERTL uses nonvolatile RAM as logging store and adopts partitioned logging technique. Thus, compared with SLSD and PLSD, the logging cost of NRPERTL is sharply reduced.

The number of logging partitions is an important factor that influences the performance of NRPERTL. Figure 4 shows how the number of logging partitions affects MDR in the case of fixed arrival rate of transaction (40 trans/sec.). As we can see, the system performance improves (i.e., MDR degrades) correspondingly with the increase of the number of logging partitions. However, when the number of logging partitions exceeds 8, further increment of the number of logging partitions obviously has not been able to obviously improve the system performance. This is because the increment of the number of logging partitions also causes the increment of recovery processing cost.

The time of system denying services after a crash is another important metric measuring the performance of a recovery scheme. For a DRTMMDBS, the time of system denying services T_{down} mainly includes (1) the time of loading data into the main memory from LSDB, notated by T_1, and (2) the time of restoring LMDB to consistency state, notated by T_2. Due to the use of the dynamic recovery strategy, critical data class is first loaded and recovered, and then system services are restored before loading ordinary data class, so T_{down} of RTDCRS may be calculated approximately as follows: $T_{down} = T_1 + T_2 \approx S_{LMDB} \times R_{CD} \times T_{I/O} + (SL \times R_{CL} \times \alpha) \div S_{LR} \times T_{rp}$, where $T_{I/O}$ denotes the average time to perform one time disk access; R_{CL} denotes the ratio of critical logging partition in total used logging storage area; S_{LR} stands for the average size of a log record, and T_{rp} denotes the average time of processing a log record in main memory; the meanings of S_{LMDB}, R_{CD}, SL, and α are described in Table 1. Because the ratio of critical data class in total data is usually small, RTDCRS can obviously decrease the time of denying services in comparison to the traditional recovery strategy.

CONCLUSION

Distributed real-time main memory databases are usually applied in some time-critical applications. For these applications, rapid and efficient recovery in the event of site crash is very important. We systematically research into the crash recovery strategy suitable for DRTMMDBS, including real-time logging scheme, local fuzzy checkpoint, and dynamic recovery processing strategy.

In this chapter, recovery correctness criteria for DRTMMDBS are first given through analyzing recovery requirements after failures. Then, a real-time dynamic crash recovery scheme (RTDCRS) suitable for DRTMMDBS is presented, and the correctness of RTDCRS is proved. RTDCRS uses nonvolatile RAM as logging store and adopts real-time logging scheme integrating the properties of partitioned logging and ephemeral logging in order to reduce the logging cost as possible during

Table 1. Experiment parameters

Parameters	Value (unit)	Description
NS	5	Number of site of DRTMMDBS
S_{LMDB}	10000 (Page)	Size of LMDB at each site
R_{CD}	0.4	Ratio of critical data class in total data
R_{TD}	0.8	Ratio of temporal data in total data
NP	4	Number of logging storage partitions
SL	8 (MB)	Size of logging storage area (nonvolatile RAM) at each site
AET	0.4 (ms)	Average execution time per transaction operation
PU	0.4	Probability of a transaction operation to be update operation
Slack	U[2.0, 6.0]	Slack factor
NTO	U[4, 8]	Number of operations contained in a transaction
α	0.8	Threshold of the utilization rate of logging storage area

Figure 3. Comparison of four logging schemes

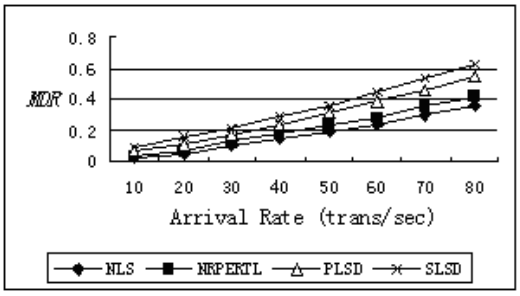
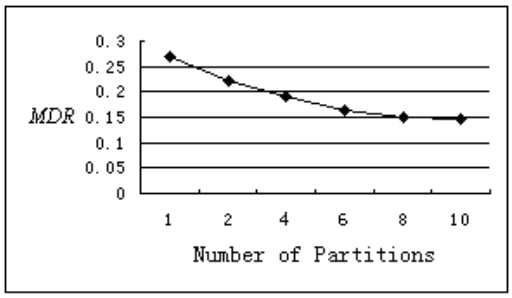


Figure 4. Influence of number of partitions



the normal running. During restart recovery after site crashes, a dynamic recovery method based on the classification recovery strategy, which supports concurrent of system services and recovery processing, is adopted to decrease the downtime to the most extent. Experiments and evaluations show that RTDCRS has better performance than traditional recovery schemes in two aspects: the missing deadlines ratio of transactions and the time of system denying services after crashes.

The main works of this chapter can be summarized as follows:

(1) Give the recovery correctness criteria of DRTMMDBS on the basis of analyzing the failure recovery requirements of DRTMMDBS.
(2) Present an efficient real-time logging scheme NRPERTL, which adopts nonvolatile RAM as logging storage area and integrates the properties of partitioned logging and ephemeral logging; propose the principle of the division of transaction classes to guarantee correct and consistent recovery.
(3) Present the local fuzzy checkpointing scheme adaptable to NRPERTL.
(4) In the processing strategy after a crash, propose the dynamic recovery processing strategy based on the idea of classification recovery.
(5) Prove the correctness of the previous recovery methods and verify their advantage in performance by the performance test and evaluation.

Although these recovery methods aim mainly at the requirement of DRTMMDBS, the ideas can apply to traditional distributed database and centralized databases as well.

FUTURE TRENDS

Although RTDCRS tries hard to minimize the time of system denying services after crashes by reducing the time of recovery processing and the logging cost, it still cannot prevent the occurrence of crashes. Some application fields, such as telecommunication, banks, insurance, and so forth, usually require database systems to provide ceaseless service; that is, the annual average service time of database systems must reach 99.999%, which means that the annual total denying services time is controlled in 5.3 minutes. Obviously, the existing failure recovery techniques (restart recovery after a crash) cannot already satisfy the previous requirements. Consequently, database management systems, which can detect and repair the possible errors in

advance to avoid the possible damages and prevent failures, are needed. In other words, we need to study the new recovery mechanisms and strategies supporting self-percepting, self-diagnosing, and self-adaptive repairing. The self-percepting, self-diagnosing and self-adaptive repairing mechanisms and strategies can prevent or delay failures effectively by perceiving the possible errors or hidden troubles in advance, which can result in system failures and self-adaptive repairing these errors and hidden troubles diagnosed. We give the recovery system architecture integrating the self-percepting, self-diagnosing, and self-adaptive repairing mechanisms.

Besides the components found in conventional database systems, the proposed recovery system architecture adds Error Perceptron, Error Diagnosis, Error Repairer, and Predictive Value Generator. The function of the Error Perceptron is to perceive the latent errors and hidden troubles by means of Error Characteristic Database. The Error Diagnosis is responsible for making a definite diagnosis to the errors captured by the Error Perceptron by the aid of the related knowledge database. The Error Repairer repairs the errors confirmed by the Error Diagnosis by means of Audit Information and Fuzzy Repair Schemes. The Predictive Value Generator is responsible for generating predictive values of damaged data items and providing them to these transactions that need immediate access to those data items.

In order to support the previous functional modules, besides audit information and log, some extra information such as error characteristics, repair schemes, statistical information of damaged data items, diagnosis knowledge database, and so forth, is also required. We intend to realize the Error Perceptron, Error Diagnosis, and Error Repairer based on intelligent agent.

The research on self-percepting, self-diagnosing, and self-adaptive repairing mechanisms is a challenging research subject. There are many works that are worth us doing.

ACKNOWLEDGMENT

This work is supported by the Natural Science Foundation of Tianjin under Grant No. 51415030203JW and the Science and Technology Development Foundatin of Tianjin Higer-learning under Grant 2006BA16.

REFERENCES

Agrawal, R., & Jagadish, H.V. (1989). *Recovery algorithms for database machines with non-volatile memory.* Proceedings of the 6th International Workshop on Database Machines, 269–285.

Aslinger, A., & Son, S.H. (2005). *Efficient replication control in distributed real-time databases.* Proceedings of the 3rd ACS/IEEE International Conference on Computer System and Applications.

Choi, M.S., et al. (2000). *Two-step backup mechanism for real-time main memory database recovery***.** Proceedings of the 7th International Conference on Real-time Computing Systems and Applications, 453–457.

Gustafsson, T., Hallqvist, H., & Hansson, J. (2005). *A similarity-aware multiversion concurrency control and updating algorithm for up-to-date snapshots of data*. Proceedings of the 17th Euromicro Conference on Real-Time Systems, 229–238.

Haritsa, J., Ramamritham, K., & Gupta, R. (2000). The PROMPT real-time commit protocol. *IEEE Transaction on Parallel and Distributed Systems, 11*(2), 160–181.

Kim, J.C., & Park, S. (1996). Recovery method using shadow paging in MMDBS. *Korea Information Science Society Journal, 14*(6), 428–431.

Kim, J.L., & Park, T. (1993). An efficient algorithm for checkpointing recovery in distributed systems.

IEEE Transactions on Parallel and Distributed Systems, 4(8), 955–960.

Kim, J.L., & Park, T. (1994). *An efficient recovery scheme for locking-based distributed database systems.* Proceedings of the 13th Symposium on Reliable Distributed Systems, 116–123.

Lam, K.Y., & Kuo, T.W. (Eds.). (2001). *Real-time database architecture and techniques.* Boston: Kluwer Academic Publishers.

Lam, K.Y., Kuo, T.W., Tsang, W.H., & Law, G.C.K. (2000). Concurrency control in mobile distributed real-time database systems. *Information Systems, 25*(4), 261–286.

Liu, P., Amman, P., & Jajodia, S. (2000). Rewriting histories: Recovering from malicious transactions. *Distributed and Parallel Databases, 8*(1), 7–40.

Qin, B., & Liu, Y.S. (2003). High performance distributed real-time commit protocol. *The Journal of Systems and Software, 68*(2), 145–152.

Sivasankaran, R.M., Ramamritham, K., Stankovic, J.A., & Towsley, D.F. (1995). *Data placement, logging and recovery in real-time active database.* Proceedings of the 1st International Workshop on Active and Real-time Database Systems, 226–241.

Son, S.H., & Agrawala, A.K. (1989). Distributed checkpointing for globally consistent states of databases. *IEEE Transactions on Software Engineering, 15*(10), 1157–1167.

Son, S.H., & Kouloumbis, S. (1993). A token-based synchronization scheme for distributed real-time databases. *Information Systems, 18*(6), 375–389.

Ulusoy. (1993). Lock-based concurrency control in distributed real-time database systems. *Journal of Database Management, 4*(2), 3–16.

Ulusoy. (1994). Processing real-time transactions in a replicated database system. *Distributed and Parallel Databases, 2*(4), 405–436.

Wang, Y.M., & Fuchs, W.K. (1993). *Lazy checkpoint coordination for bounding rollback propagation.* Proceedings of the 12th Symposium on Reliable Distributed Systems, 78–85.

Woo, S.K., Kim, M.H., & Lee, Y.J. (1997). A recovery method based on shadow updating and fuzzy checkpointing in main memory database systems. *Korea Information Science Society Journal, 24*(3), 266–278.

Xiao, Y.Y., Liu, Y.S., Deng, H.F., & Liao, G.Q. (2006). One-phase real-time commitment for distributed real-time transactions. *Journal of Huazhong University of Science and Technology, 34*(4), 1–4.

KEY TERMS

Deadline: In a real-time database system or a distributed real-time database system, deadline is used to describe the timing constraint of transaction. The deadline of a transaction defines the specified time by which the transaction must be completed.

Distributed Real-Time Database System (DRTDBS): A distributed database system within which transactions and data have timing characteristics or explicit timing constraints, and system correctness depends not only on the logic results but also on the time at which the logic results are produced.

Distributed Real-Time Main Memory Database System (DRTMMDBS): The distributed real-time database system that adopts a main memory database as its ground support.

Main Memory Database (MMDB): The database where the primary copy of a database is placed in main memory and a "secondary copy" of a database on disks serves as a backup.

Partitioned Logging: A novel logging technique relative to the sequential logging. Partitioned logging stores log records according to transaction class (data class); that is, log records belonging to different transaction class (data class) are stored in different partitions.

Sequential Logging: A traditional logging technique. The logging technique requires storing log records into a single logging file in sequence according to the execution order of concurrent transaction operations.

System Failure: Also called crash, occurs when the contents of volatile main memory storage are lost or corrupted.

Temporal Data Object: Used to record the status of external objects in the world. Each temporal data object is associated with a period of validity. A temporal data object will lose validity after its period of validity.

Chapter LXXXIII
Querical Data Networks

Cyrus Shahabi
University of Southern California, USA

Farnoush Banaei-Kashani
University of Southern California, USA

INTRODUCTION

Recently, a family of massive self-organizing data networks has emerged. These networks mainly serve as large-scale distributed query processing systems. We term these networks *Querical Data Networks (QDN).* A QDN is a federation of a dynamic set of peer, autonomous nodes communicating through a transient-form interconnection. Data is naturally distributed among the QDN nodes in extra-fine grain, where a few data items are dynamically created, collected, and/or stored at each node. Therefore, the network scales linearly to the size of the dataset. With a dynamic dataset, a dynamic and large set of nodes, and a transient-form communication infrastructure, QDNs should be considered as the new generation of distributed database systems with significantly less constraining assumptions as compared to their ancestors. Peer-to-peer networks (Daswani, 2003) and sensor networks (Estrin, 1999, Akyildiz, 2002) are well-known examples of QDN.

QDNs can be categorized as instances of "complex systems" (Bar-Yam, 1997) and studied using the complex system theory. Complex systems are (mostly natural) systems hard (or

complex) to describe information-theoretically, and hard to analyze computationally. QDNs share the same characteristics with complex systems, and particularly, bear a significant similarity to a dominating subset of complex systems most properly modeled as large-scale interconnection of functionally similar (or peer) entities. The links in the model represent some kind of system-specific entity-to-entity interaction. Social networks, a network of interacting people, and cellular networks, a network of interacting cells, are two instances of such complex systems. With these systems, complex global system behavior (e.g., a social revolution in a society, or food digestion in stomach!) is an emergent phenomenon, emerging from simple local interactions. Various fields of study, such as sociology, physics, biology, chemistry, etc., were founded to study different types of initially simple systems and have been gradually matured to analyze and describe instances of incrementally more complex systems. An interdisciplinary field of study, the complex system theory[a], is recently founded based on the observation that analytical and experimental concepts, tools, techniques, and models developed to study an instance of complex system at one field can be adopted, often almost unchanged, to study other complex systems in other fields of study. More importantly, the complex system theory can be considered as a unifying meta-theory that explains common characteristics of complex systems. One can extend application of the complex system theory to QDNs by:

1. Adopting models and techniques from a number of impressively similar complex systems to design and analyze QDNs, as an instance of engineered complex systems; and
2. Exporting the findings from the study of QDNs (which are engineered, hence, more controllable) to other complex system studies.

This article is organized in two parts. In the first part, we provide an overview, where we 1) define and characterize QDNs as a new family of data networks with common characteristics and applications, and 2) review possible database-like architectures for QDNs as query processing systems and enumerate the most important QDN design principles. In the second part of the article, as the first step toward realizing the vision of QDNs as complex distributed query-processing systems, we focus on a specific problem, namely the problem of effective data location (or search) for efficient query processing in QDNs. We briefly explain two parallel approaches, both based on techniques/models borrowed from the complex system theory, to address this problem.

BACKGROUND

Here, we enumerate the main componental characteristics and application features of a QDN.

Componental Characteristics

A network is an interconnection of nodes via links, usually modeled as a graph. Nodes of a QDN are often massive in number and bear the following characteristics:

- **Peer functionality:** All nodes are capable of performing a restricted but similar set of tasks in interaction with their peers and the environment, although they might be heterogeneous in terms of their physical resources. For example, joining the network and forwarding search queries are among the essential peer tasks of every node in a peer-to-peer network.
- **Autonomy:** Aside from the peer tasks mentioned above, QDN nodes are autonomous in their behavior. Nodes are either self-governing, or governed by out-of-control

uncertainties. Therefore, to be efficacious and applicable the QDN engineering should avoid imposing requirements to and making assumptions about the QDN nodes[b]. For example, strict regulation of connectivity (e.g., enforcing number of connections and/or target of connections) might be an undesirable feature for a QDN design.

- **Intermittent presence:** Nodes may frequently join and leave the network based on their autonomous decision, due to failures, etc.

On the other hand, links in various QDNs stand for different forms of interaction and communication. Links may be physical or logical, and they are fairly inexpensive to rewire. Therefore, a QDN is a large-scale federation of a dynamic set of autonomous peer nodes building a transient-form interconnection. Conventional approaches developed to model and analyze traditional distributed database systems (and classical networks, as their underlying communication infrastructure) are either too weak (oversimplifying) or too complicated (overcomplicated) to be effective with large-scale and topology-transient QDNs. The complex system theory (Bar-Yam, 1997), on the other hand, provides a set of conceptual, experimental, and analytical tools to contemplate, measure, and analyze systems such as QDNs.

Application Features

A QDN is applied as a distributed source of data (a *data network*) with nodes that are specialized for cooperative query processing and data retrieval. The node cooperation can be as trivial as forwarding the queries, or as complicated as in-network data analysis. In order to enable such an application, QDN should support the following features:

- **Data-centric naming, addressing, routing, and storage:** With a QDN, queries are declarative; i.e., query refers to the names of data items and is independent of the location of the data. The data may be replicated and located anywhere in the data network, the data holders are unknown to the querier and are only intermittently present, and the querier is interested in data itself rather than the location of the data. Therefore, naturally QDN nodes should be named and addressed by their data content rather than an identifier in a virtual name space such as the IP address space. Consequently, with data-centric naming and addressing of the QDN nodes (Heidemann, 2001), routing (Ratnasamy, 2002) and storage (Ratnasamy, 2003) in QDN are also based on the content. It is interesting to note that non-procedural query languages such as SQL also support declarative queries and are appropriate for querying data-centric QDNs.
- **Self-organization for efficient query processing:** QDNs should be organized for efficient query processing. A QDN can be considered as a database system with the data network as the database itself (see the next section). QDN nodes cooperate in processing the queries by retrieving, communicating, and preferably on-the-fly processing of the data distributed across the data network. To achieve efficiency in query processing with high resource utilization and good performance (e.g., response time, query throughput, etc.), QDN should be *organized* appropriately. Examples of organization are: intelligent partitioning of the query to a set of sub-queries to enable parallel processing, or collaborative maintenance of the data catalogue across the QDN nodes. However, the peer tasks of the QDN nodes should be defined such that they *self-organize* to the appropriate organization. In other words, organization must be a collective behavior that emerges from local interactions among nodes; otherwise the dynamic nature and

large scale of QDN render any centralized micro-management of QDN unscalable and impractical.

Application Areas

Below, we enlist some of the application areas where QDN is exploited as the enabling infrastructure:

- Resource sharing Applications
- Integrated Web Services Computing
- P2P Computing
- Grid Computing
- Pervasive Computing
- Autonomic Computing
- Mobile Computing
- Network Administration Applications
- Characterization
- Monitoring and Troubleshooting
- Optimization

VISION: A DATABASE QUERYING FRAMEWORK FOR QDNS

In previous section, we defined a Querical Data Network (QDN) as a distributed data source and a query processing system. On the other hand, a database system (DBS) is designed specifically as a general framework for *convenient* and *efficient* querying of static or dynamic collections of interrelated data. Thus, querying QDNs can be designed, developed, and executed by adopting DBS as the general-purpose querying framework, leveraging on its rich abstractions, theories, and processing methods (Govindan, 2002, Harren, 2002). In particular, adopting the DBS framework potentially results in 1) convenient and rapid application development for users at the conceptual level, by providing well-known and transparent abstractions independent of the implementation of the querying, and 2) efficient query processing at the physical level, by providing a general-purpose querying component that adopts and customizes query processing methods from the database literature as well as other related fields such as distributed computing.

Here, we depict and compare potential architectures for the DBS framework, define a taxonomy of approaches to generalize this querying framework for the entire family of QDNs, and enumerate some important design principles for query processing in QDNs.

Architecture

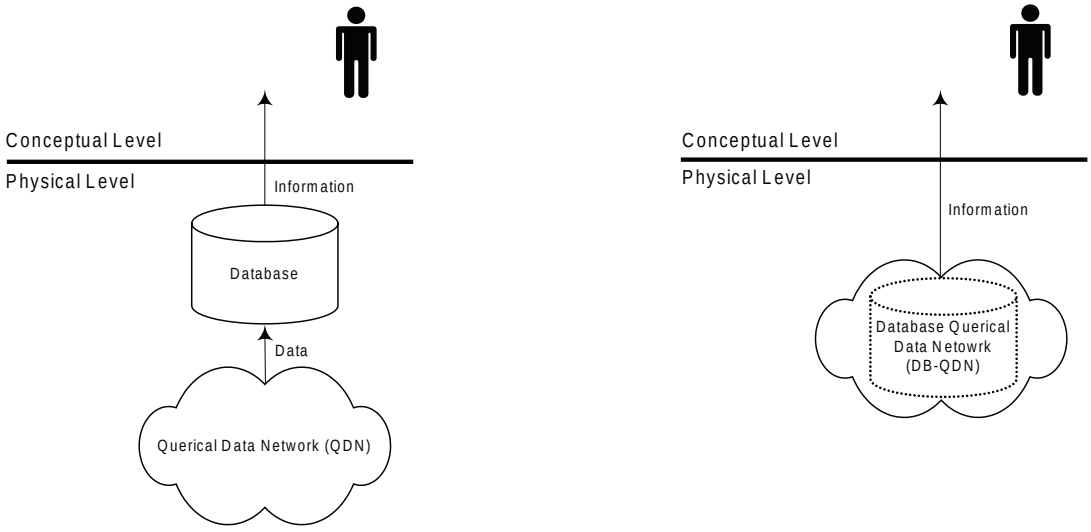
The DBS querying framework for QDNs suggests a 2-level architecture comprising of conceptual level and physical level. At the conceptual level, queries are defined based on the conceptual schema of the data network, independent of the physical implementation of the query processing. The physical data independence allows rapid development of QDN applications, similar to the database application development. For instance, a peer-to-peer application that monitors violation of speed limit at a highway can pose the following query to a (hypothetical) mobile peer-to-peer network of vehicles:

```
SELECT      vehicle-ID
FROM        Cars
WHERE       (speed > 70) AND (location IN
"Highway No. 88")
```

Similarly, a heat alert application may pose the following query to a heat detection sensor field: *"Report the outlier heat data in all offices during night."*

Queries are executed at the physical level. There are two extreme choices of design for query processing at the physical level: *centralized* and *decentralized*; a hybrid design may also be meaningful for particular applications. With a centralized design, a potential querier (i.e., one of the QDN nodes or an outsider) receives the query from the QDN application at the conceptual level.

Figure 1. Database system framework for querying querical data networks (QDNs)



a. Centralized Design

b. Decentralized Design

The query is disseminated to all QDN nodes, where the query is interpreted based on the local conceptual schema and all raw data required to process the query are transmitted back to the querier via the network (see Figure 1-a). The querier treats the collected data as a centralized database, and processes and analyzes the data to respond the query. With this scheme, data sourcing and query processing are completely decoupled; the data network maintains and communicates the data, and the querier processes the query individually. The cooperation among QDN nodes is limited to forwarding the query and the raw data, and the network is used only as a point-to-point communication infrastructure to communicate the data between the sources and the querier.

A centralized design is simple to implement. The centralized query processing approach is also resource-efficient for trivial queries such as typical peer-to-peer search queries. However, with complex queries, where, for example, two 10000-record tables must be joined to retrieve a few records, the overhead of transmitting the entire content of the tables to the querier for central processing is overwhelming and renders the centralized approach impractical. With most QDNs (e.g., sensor networks), this overhead is particularly intolerable due to the several orders of magnitude higher cost of communication as compared with that of computation in typical QDN nodes. Instead, in-network and on-the-fly processing of the query potentially eliminates the redundant communication of the data; hence, it is more efficient and scalable. The decentralized design adopts the latter approach.

With the decentralized design, the QDN is itself both the source of the data and the query-processing unit (see Figure 1-b). The data sourcing/communication and data analysis tasks are integrated, and QDN nodes cooperate to perform both tasks within the network. With this approach, queries are processed based on the following general scheme: query is disseminated to a selected set of QDN nodes; QDN nodes exploit their computing

power to process the query locally, cooperatively, in parallel, and in a distributed fashion, to extract the required information from the raw data; eventually, the extracted information merge to comprise the final query result while traveling toward the querier through the network.

The efficiency of the decentralized design is due to in-network processing. With this approach, communication of the raw data is restricted to short-range (hence, less costly) communications among local cooperative-analysis groups of QDN nodes, which process the voluminous raw data and extract the concise required information to respond to the query. Although the decentralized design potentially promises an efficient and scalable querying framework for QDNs, realizing this design is a challenging endeavor and requires designing efficient distributed and cooperative query processing mechanisms that comply with specific characteristics of QDNs.

Taxonomy

Based on the two fundamentally distinct design choices for the physical level of the DBS framework (i.e., centralized and decentralized), one can recognize two approaches to implement a DBS-based querying system for QDNs:

1. **Database *for* QDN:** This approach corresponds to the querying systems with centralized query processing. These systems are similar to other centralized database applications, where data are collected from some data sources (depending on the host application, the data sources can be text documents, media files, and in this case, QDN data) to be separately and centrally processed.
2. **Database-QDN (DB-QDN):** The systems that are designed based on this approach strive to implement query processing in a decentralized fashion within the QDN; hence, in these systems "*QDN is the database*".

Design Principles

By definition QDNs tend to be large-scale systems and their potential benefits increase as they grow in size. Therefore, between the two types of DBS-based querying systems for QDNs, the database-QDNs (DB-QDNs) are more promising because they are scalable and efficient. Among the most important design principles for distributed query processing at DB-QDNs one can distinguish the following:

1. **In-network query processing:** In-network query processing is the main distinction of DB-QDNs. In-network query processing techniques should be implemented in a distributed fashion, ensuring minimal communication overhead and optimal load-balance.
2. **Transaction processing with relaxed properties:** Due to the dynamic nature of QDNs, requiring ACID-like properties for transaction processing in DB-QDNs is too costly to be practical and severely limits the scalability of such processing technique. Hence, transaction-processing properties should be relaxed for DB-QDNs.
3. **Adaptive query optimization:** Since QDNs are inherently dynamic structures, optimizing query plans for distributed query execution in DB-QDNs should also be a dynamic/adaptive process. Adaptive query optimization techniques are previously studied in the context of central query processing systems (Avnur, 2000).
4. **Progressive query processing:** Distributed query processing tends to be time-consuming. With real-time queries, user may prefer receiving a rough estimation of the query result quickly rather than waiting long for the final result. The rough estimation progressively enhances to the accurate and final result. This approach, termed *progressive*

query processing (Schmidt, 2002-a), allows users to rapidly obtain a general understanding of the result, to observe the progress of the query, and to control its execution (e.g., by modifying the selection condition of the query) on the fly.

5. **Approximate query processing:** Approximation techniques such as wavelet-based query processing can effectively decrease the cost of the query, while producing highly accurate results (Schmidt, 2002-b). Inherent uncertainty of the QDN data together with the relaxation of the query semantics justify application of approximation techniques to achieve efficiency.

FUTURE TRENDS

One of the most fundamental functionalities required to realize a DB-QDN is the *search* primitive. Efficient location of the data within QDN, a large-scale and dynamic system with distributed and dynamic dataset, is a challenging task vital to QDN query processing. For the remainder of this section, we briefly explain two parallel approaches one can adopt from the complex system theory to address the QDN search problem. First, we discuss a self-organizing mechanism to structure the topology of the QDN to a search-efficient topology. This topology can be considered as a distributed index structure that organizes the nodes and therefore, the data content of the nodes, for efficient search. For the design of the search-efficient QDN topology as well as the search dynamics, we are inspired by the "small-world" models. Small-worlds are models proposed to explain efficient communication in a social network, which is a semi-structured complex system.

Second, we propose an efficient query flooding mechanism for QDNs. Flooding is not only required for broadcast queries at all QDNs, but also for uni-cast and multi-cast queries in unstructurable/unindexible QDNs. With these QDNs, the extreme dynamism of the QDN topology and the extreme autonomy of the QDN nodes renders any attempt to impose even a semi-structure on the network by an index-like structure inefficient and/or impossible. We use percolation theory, an analytical tool borrowed from the complex system theory, to formalize and analyze such efficient flooding mechanism.

Probabilistic Indexing of QDNs for Efficient Approximate Query Processing

Considering a QDN as a database (with every node of the QDN as the potential entry point of the query), similar to traditional databases QDN should be "indexed" for efficient processing of the queries. To process approximate queries[c], we propose self-organizing the interconnection of the QDN based on the data content of the QDN nodes. With this organization, the network distance between every two nodes is positively correlated with the similarity of their data content with high probability. This approach results in an indexed network with distinguishable data localities, allowing efficient routing of the queries toward the nodes holding the result set of the query. The similarity measurements performed by each node while joining QDN to select an appropriate set of neighbors can be thought of as the off-line pre-computations required to create the index for efficient on-line query processing. Also, the topology of the generated interconnection should be compared with the tree-like topologies of the traditional hierarchical index structures in centralized databases. In addition to allowing efficient navigation/traversal of the dataset, this topology should support the dynamism of the dataset and the network, and more importantly, should avoid assuming a central entry point (the

root node in hierarchical indices) for the query, in order to balance the query load among all the nodes of the QDN.

It turns out that a probabilistic "small-world" model, which is a topology proposed to explain efficient communication in social networks, is a perfect candidate topology to index QDNs. With our *searchable* QDN model (Banaei-Kashani, 2003), we propose a self-organization mechanism that generates a QDN with small-world topology based on a recently developed small-world model (Watts, 2002). We complement the generated small-world network topology (i.e., the index) with a query forwarding mechanism (i.e., the index lookup technique) that effectively routes partial-match queries toward the QDN nodes that store the matching data items. Currently, we are focusing on extending this query routing technique to support more challenging queries such as range queries and nearest-neighbor queries.

Criticality-Based Probabilistic Flooding

Flooding is a common mechanism used in many networks, including QDNs, to broadcast a piece of information (e.g., an alert, or a search query) from a source node to other nodes of the network. With normal flooding, each node always forwards the received information to all its neighbors (i.e., directly connected nodes). In spite of many beneficial features, such as providing broad coverage and guaranteeing minimum delay, normal flooding is not a scalable communication mechanism, mainly because of the communication overhead it imposes to the system. To alleviate this problem, we introduce *probabilistic flooding* (Banaei-Kashani, 2003). With probabilistic flooding, unlike normal flooding, a node forwards the information to its neighbor probabilistically, with probability *p*. By changing the probability value *p*, we can control the effective connectivity of the network while information is forwarded. The idea is to tune the probability value *p* to a critical operation point (the phase transition point) such that statistically the network remains connected (to preserve full reachability) while redundant paths are eliminated. Percolation theory (Stauffer, 1992) is an analytical tool from the complex system theory extensively used to study probabilistic diffusion-like physical phenomena; e.g., diffusion of oil inside porous rocks in oil reservoir, a physical complex system. We use percolation theory to formalize the probabilistic flooding approach as a query-diffusion problem, and to find its critical (optimal) operating point rigorously. Our formal analysis shows that the critical value of *p* can be as low as 1%, which translates to 99% reduction in communication overhead of flooding, hence, scalable flooding.

CONCLUSION

In this article, we identified Querical Data Networks (QDNs) as a family of data networks, recently emerging as a new generation of distributed database systems with significantly less constraining assumptions. We envision a QDN as a distributed query processing system with a database-like architecture. In search of an effective approach to design and analyze Database-QDNs, we find the complex system theory, a theory that explains a family of systems with characteristics that bear significant similarity to those of QDNs, extremely helpful. As an instance application of this approach, we provide two parallel solutions for the QDN search problem inspired by models adopted from the complex system theory.

ACKNOWLEDGMENT

This research has been funded in part by NSF grants EEC-9529152 (IMSC ERC), IIS-0238560 (PECASE), IIS-0324955 (ITR), and unrestricted cash gifts from Google and Microsoft. Any opinions, Findings, and conclusions or recommenda-

tions expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

REFERENCES

Akyildiz, I. F., Su, W., Sankarasubramaniam, Y., & Cayirci, E. (2002). A Survey on Sensor Networks. *IEEE Communications Magazine, 40*(8), 102-114.

Avnur, R., & Hellerstein, J. (2000). Eddies: Continuously adaptive query processing. *Proceedings of the International Conference on Management of Data (SIGMOD), 2000,* (pp. 261-272).

Banaei-Kashani, F., & Shahabi, C. (2003b). Searchable querical data networks. In *Lecture notes in computer science, 2944,* 17-32. Berlin, Germany: Springer-Verlag.

Banaei-Kashani, F., & Shahabi, C. (2003a). Criticality-based analysis and design of unstructured peer-to-peer networks as complex systems. *Proceedings of the Third International Workshop on Global and Peer-to-Peer Computing (GP2PC) in conjunction with CCGrid, 2003,* 351-359.

Bar-Yam, Y. (1997). *Dynamics of complex systems.* New York, NY: Westview Press.

Daswani, N., Garcia-Molina, H., & Yang, B. (2003). Open problems in data-sharing peer-to-peer systems. *Proceedings of the International Conference on Database Theory (ICDT), 2003,* 1-15.

Estrin, D., Govindan, R., Heidemann, J., & Kumar, S. (1999). Next century challenges: Scalable coordination in sensor networks. *Proceedings of the International Conference on Mobile Computing and Networks (MobiCom), 1999,* 256-262.

Govindan, R., Hellerstein, J., Hong, W., Madden, S., Franklin, M., & Shenker, S. (2002). *The sensor network as a database* (Tech. Rep. No. 02-771). University of Southern California, Los Angeles, CA, 2002.

Harren, M., Hellerstein, J., Huebsch, R., Loo, B. T., Shenker, S., & Stoica, I. (2002). Complex queries in DHT-based peer-to-peer networks. *Proceedings of the International Workshop on Peer-to-Peer Systems (IPTPS), 2002.*

Heidemann, J., Silva, F., Intanagonwiwat, C., Govindan, R., Estrin, D., & Ganesan, D. (2001). Building efficient wireless sensor networks with low-level naming. *Proceedings of the Symposium on Operating Systems Principles (SOSP), 2001,* 146-159.

Ratnasamy, S., Francis, P., Handley, M., Karp, R., & Shenker, S. (2001). A scalable content addressable network. *Proceedings of the Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication (SIGCOMM), 2001,* 161-172.

Ratnasamy, S., Karp, B., Shenker, S., Estrin, D., Govindan R., Yin, L., & Yu, F. (2003). Data-Centric Storage in Sensornets with GHT, a Geographic Hash Table. *Mobile Networks and Applications, 8*(4), 427-442.

Schmidt, R., & Shahabi, C. (2002a). How to evaluate multiple range-sum queries progressively. *21st ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems (PODS), 2002,* 133-141.

Schmidt, R., & Shahabi, C. (2002b). Propolyne: A fast wavelet-based algorithm for progressive evaluation of polynomial range-sum queries. *Proceedings of the Conference on Extending Database Technology (EDBT), 2002,* 664-681.

Stauffer, D., & Aharony, A (1992). *Introduction to percolation theory* (2nd ed.). London, UK: Taylor and Francis.

Watts, D. J., Dodds, P. S., & Newman, M. E. J. (2002). Identity and search in social networks. *Science, 296,* 1302-1305.

KEY TERMS

Complex Systems: Complex Systems is a new field of science studying how parts of a complex system give rise to the collective behaviors of the system. Complexity (information-theoretical and computational) and emergence of collective behavior are the two main characteristics of such complex systems. Social systems formed (in part) out of people, the brain formed out of neurons, molecules formed out of atoms, the weather formed out of air flows are all examples of complex systems. The field of Complex Systems cuts across all traditional disciplines of science, as well as engineering, management, and medicine.

Content-Centric Networks: A content-centric network is a network where various functionalities such as naming, addressing, routing, storage, etc., are designed based on the content. This is in contrast with classical networks that are node-centric.

Distributed Hash Tables (DHTs): A distributed index structure with hash table-like functionality for information location in the Internet-scale distributed computing environments. Given a key from a pre-specified flat identifier space, DHT computes (in a distributed fashion) and returns the location of the node that stores the key.

Peer-to-Peer (P2P) Networks: A peer-to-peer network is a distributed, self-organized federation of *peer* entities, where the system entities collaborate by sharing resources and performing cooperative tasks for mutual benefit. It is often assumed that such a federation lives, changes, and expands independent of any distinct service facility with global authority.

Percolation Theory: Assume a grid of nodes where each node is occupied with probability *p* and empty with probability *(1-p)*. Percolation theory is a quantitative (statistical-theoretical) and conceptual model for understanding and analyzing the statistical properties (e.g., size, diameter, shape, etc.) of the clusters of occupied nodes as the value of *p* changes. Many concepts associated with complex systems such as clustering, fractals, diffusion, and particularly phase transitions are modeled as percolation problem. The significance of the percolation model is that many different problems can be mapped to the percolation problem; e.g., forest-fire spread, oil field density estimation, diffusion in disordered media, etc.

Sensor Networks: A sensor network is a network of low-power, small form-factor sensing devices that are embedded in a physical environment and coordinate amongst themselves to achieve a larger sensing task.

Small World Models: It is believed that almost any pair of people in the world can be connected to one another by a short chain of intermediate acquaintances, of typical length about six. This phenomenon is colloquially referred to as the "six degrees of separation," or equivalently, the "small-world" effect. Sociologists propose a number of topological network models, the small-world models, for the social network to explain this phenomenon.

ENDNOTES

[a] Go to New England Complex Systems Institute (http://necsi.org/) for more information about the complex system theory.

[b] One can consider peer tasks as *rules of federation*, which govern the QDN but do not violate autonomy of individual nodes.

[c] Considering the transience of the QDN structure and the dynamism of the dataset, *exact* query processing with zero false dismissal is not a practical option.

Chapter LXXXIV
On the Implementation of a Logic Language for NP Search and Optimization Problems

Sergio Greco
University of Calabria, Italy

Cristian Molinaro
University of Calabria, Italy

Irina Trubitsyna
University of Calabria, Italy

Ester Zumpano
University of Calabria, Italy

INTRODUCTION

It is well known that NP search and optimization problems can be formulated as DATALOG$^{\neg}$ (datalog with unstratified negation; Abiteboul, Hull, & Vianu, 1994) queries under nondeterministic stable-model semantics so that each stable model corresponds to a possible solution (Gelfond & Lifschitz, 1988; Greco & Saccà, 1997; Kolaitis & Thakur, 1994). Although the use of (declarative) logic languages facilitates the process of writing complex applications, the use of unstratified negation allows programs to be written that in some cases are neither intuitive nor efficiently valuable. This article presents the logic language NP Datalog, a restricted version of DATALOG$^{\neg}$ that admits only controlled forms of negation, such as stratified negation, exclusive disjunction, and constraints. NP Datalog has the same expressive power as DATALOG$^{\neg}$, enables a simpler and intuitive formulation for search and optimization problems, and can be easily translated into other formalisms. The example below shows how the vertex cover problem can be expressed in NP Datalog.

Example 1. Given an undirected graph $G = <N, E>$, a subset of the vertexes $V \subseteq N$ is a vertex

cover of *G* if every edge of *G* has at least one end in *V*. The problem can be formulated in terms of the NP Datalog query $<P_1,v(X)>$ with P_1 defined as follows:

$$v(X) \subseteq node(X)$$
$$edge(X,Y) \Rightarrow v(X) \vee v(Y),$$

where the output relation *v* gives a vertex cover, $\subseteq$ denotes the subset relation, and $\Rightarrow$ is used to define constraint. The first rule guesses a subset *v* of the relation *node* whereas the latter constraint verifies the vertex cover condition: It states that if *X* and *Y* are two connected nodes, then *X* or *Y* or both must be in the cover. The optimization problem computing a cover with minimum cardinality can be simply expressed by means of the query $<P_1, min|v(X)|>$.

The evaluation of logic programs with stable-model semantics can be carried out by means of current ASP (answer set programming) systems that compute the semantics of DATALOG$^{\neg}$ programs. An alternative solution consists of reducing problems expressed by means of logic formalisms (usually extensions of datalog) into equivalent SAT problems and evaluating the target problems by means of SAT solvers. In this article, a different solution, based on the rewriting of logic programs into constraint programming, is proposed. More specifically, the implementation of NP Datalog consists of translating NP Datalog queries into OPL (optimization programming language; Van Hentenryck, 1999; Van Hentenryck, Michel, Perron, & Regin, 1999), a constraint language well suited for solving both search and optimization problems, and then executing the target OPL code by means of the ILOG OPL Studio platform.

Example 2. The optimization query of Example 1 is translated into the following OPL code.

```
var boolean v[node];
minimize sum (x in node) v[x]
subject to {
      forall (<x,y> in edge)
          1 ⇒ (v[x] + v[y] > 0);
}
```

where the second statement expresses the optimization condition (minimizes the cardinality of *v*), while the last one corresponds to the vertex cover condition.

NP DATALOG

Familiarity is assumed with disjunctive logic programs, disjunctive deductive databases, (disjunctive) stable-model semantics, and computational complexity (Abiteboul et al., 1994; Gelfond & Lifschitz, 1991; Johnson, 1990). In this section, the language NP Datalog is presented. This language restricts the use of unstratified negation of DATALOG$^{\neg}$ without loss of its expressive power and can be executed more efficiently or easily translated into other formalisms. NP Datalog extends DATALOG$^{\neg s}$ (datalog with stratified negation) with two simple forms of unstratified negation embedded into built-in constructs: exclusive disjunction, used in partition rules, and constraint rules.

An NP Datalog partition rule is a disjunctive rule of the form

$$p_1(\mathbf{X}) \oplus \ldots \oplus p_k(\mathbf{X}) \leftarrow Body(\mathbf{X},\mathbf{Y}) \qquad (1)$$

or of the form

$$p_0(\mathbf{X}, c_1) \oplus \ldots \oplus p_0(\mathbf{X}, c_k) \leftarrow Body(\mathbf{X},\mathbf{Y}), \quad (2)$$

where (a) $p_0, p_1, \ldots, p_k$ are distinct IDB predicates not defined elsewhere in the program, (b) $c_1, \ldots, c_k$ are distinct constants, (c) *Body*(**X,Y**) is a conjunction of literals not depending on predicates defined by disjunctive rules, and (d) **X** and **Y** are vectors of variables. The intuitive meaning of these rules is that if the body is true, then exactly one head atom must be true too, hence the projection of the body relation on **X** is partitioned nondeterministically into *k* relations or *k* distinct sets of the same relation. Clearly, every rule of the form of Equation 2 can be rewritten into a rule of the form of Equation 1 and vice versa.

A generalized partition rule is a rule of the form

$$\oplus_{\mathrm{L}}\ \mathrm{p}(\mathbf{X}, \mathrm{L}) \leftarrow \mathrm{Body}(\mathbf{X},\mathbf{Y}),\ \mathrm{d}(\mathrm{L}),$$

where *p* is an IDB predicate symbol not defined elsewhere in the program, *Body*(**X,Y**) is a conjunction of literals not depending on predicates defined by disjunctive rules, and *d* is a database domain predicate specifying the domain of the variable *L*. The intuitive meaning of such a rule is that the projection of the relation defined by *Body*(**X**,**Y**) on **X** is partitioned into a number of subsets equal to the cardinality of the relation *d*.

The existence of subset rules is also assumed, that is, rules of the form

$$\mathrm{s}(\mathbf{X}) \subseteq \mathrm{Body}(\mathbf{X},\mathbf{Y}),$$

where *s* is an IDB predicate symbol not defined elsewhere in the program and *Body*(**X,Y**) is a conjunction of literals not depending on predicates defined by partition or subset rules. This rule guesses a subset of the projection of the body relation on **X**. Observe that a subset rule of the above form corresponds to the generalized partition rule with d = {0, 1}, while every generalized partition rule can be rewritten into a subset rule and constraints.

A constraint (rule) is of the form

$$\Leftarrow \mathrm{Body}(\mathbf{X}),$$

where *Body*(**X**) is a conjunction of literals. A constraint rule of the above form is satisfied if the conjunction *Body*(**X**) is false. Constraints can be rewritten using rules of the form $A_1 \vee \ldots \vee A_k \Leftarrow B_1, \ldots, B_m$ (or $B_1, \ldots, B_m \Rightarrow A_1 \vee \ldots \vee A_k$) to denote a constraint of the form $\Leftarrow B_1, \ldots, B_m, \neg A_1, \ldots, \neg A_k$ (i.e., negative literals are moved from the body to the head).

All the variables in the above rules are range restricted.

Definition 1. An NP Datalog program consists of three distinct sets of rules:

1. partition and subset rules defining guess (IDB) predicates,
2. standard stratified datalog rules defining standard (IDB) predicates, and
3. constraint rules,

where every guess predicate is defined by a unique subset or partition rule and does not depend on other guess predicates, and every recursive predicate does not depend on guess predicates.

Definition 2. An NP Datalog search query over a database schema *DS* is a pair $Q = \langle P, g(t)\rangle$ where *P* is an NP Datalog program and $g(t)$ is an IDB atom denoting the output relation. An NP Datalog optimization query is a pair $opt(Q) = \langle P, \mathrm{opt}|g(t)|\rangle$ where $opt \in \{min, max\}$.[1]

NP Datalog queries, expressing the vertex cover search and optimization problems, were presented in Example 1. The following example shows graph coloring problems, expressed by means of NP Datalog queries.

Example 3. A *k*-coloring of a graph *G* is an assignment of one of *k* possible colors to each node of *G* such that no two adjacent nodes have the same color. The problem can be expressed as $\langle P_3, col(X, C)\rangle$ where P_3 consists of

$$\oplus_{\mathrm{C}}\ \mathrm{col}(\mathrm{X},\mathrm{C}) \leftarrow \mathrm{node}(\mathrm{X}),\ \mathrm{color}(\mathrm{C})$$
$$\Leftarrow \mathrm{edge}(\mathrm{X},\mathrm{Y}),\ \mathrm{col}(\mathrm{X},\mathrm{C}),\ \mathrm{col}(\mathrm{Y},\mathrm{C}),$$

where the base relation *color* contains exactly *k* colors. The first rule guesses a *k*-coloring for the graph by assigning a unique color to each node of the graph. The second one is a constraint stating that two connected nodes *X* and *Y* cannot be colored with the same color *C*. The query modeling the minimum coloring problem (find a *k*-coloring with minimum *k*) is obtained from the *k*-coloring example by adding a rule that stores the used colors,

$$\mathrm{used\_color}(\mathrm{C}) \leftarrow \mathrm{col}(\mathrm{X},\mathrm{C}),$$

and replacing the query goal with *min*|*used_color*(*C*)|.

It has been shown that NP Datalog has the same expressive power as DATALOG$^{\neg}$ (Zumpano, Greco, Trubitsyna, & Veltri, 2004).

TRANSLATING NP DATALOG INTO OPL

Several languages for hard search and optimization problems have been designed and implemented. These include logic languages based on stable models (e.g., DLV, Smodels; Leone et al., 2002; Syrjanen & Niemela, 2001), constraint logic programming languages (e.g., ECLiPSe, SICStus Prolog, Mozart; *SICStus Prolog*, n.d.; Van Roy, 1999; Wallace & Schimpf, 1999), and constraint programming languages (e.g., ILOG OPL, Lingo; Finkel, Marek, & Truszczynski, 2004; Van Hentenryck, 1999; Van Hentenryck et al., 1999). The advantage of using logic languages based on stable-model semantics with respect to constraint programming is their ability to express complex NP problems in a declarative way. On the other hand, constraint programming languages are very efficient in the solution of optimization problems. As NP Datalog is a language to express NP problems, the implementation of the language can be carried out by translating queries into target languages specialized in optimization problems such as constraint programming. Thus, the implementation of NP Datalog is carried out by translating NP Datalog queries into OPL programs. OPL is a constraint language well suited for solving both search and optimization problems. OPL programs are computed by means of the ILOG OPL Studio system.

This section informally shows how NP Datalog queries are translated into OPL programs. Without loss of generality, for the sake of simplicity of presentation, it is assumed that NP Datalog programs satisfy the following conditions.

- Guess predicates are defined by either generalized partition rules or subset rules.
- Standard predicates are defined by a unique extended rule of the form $A \leftarrow body_1 \vee \ldots \vee body_m$, where $body_i$ is a conjunction of literals.
- Constraint rules are of the form $A \Leftarrow B$, where A is a disjunction of atoms and B is a conjunction of atoms.
- Constants appear only in atoms of the form $x \Theta y$, $\Theta \in \{>, <, \leq, \geq, =, \neq\}$.

NP Datalog programs have associated a database schema specifying the used database domains and for each base predicate, the domains associated with each attribute. Starting from the database schema, the schema of every derived predicate is deducted and new domains, obtained from the database domains, are introduced. For instance, the database schema associated with the graph used in the *k*-coloring problem of Example 3 is

DOMAINS = {node, color}.
PREDICATES = {edge(node, node)},

whereas the schema associated with the derived predicate *col* is *col*(*node*, *color*).

Thus, given a database *D* and a query $Q = <P, G>$, an OPL program equivalent to the application of the query *Q* to the database *D* is generated. First, it is shown how databases are translated and, next, the compilation of queries is presented.

Database Translation. A domain relation is translated into an enumerated type. The translation of a base relation *r* with arity $n>0$ consists of two steps: (a) declaring a new record type with *n* fields, where the type of the i^{th} field is the enumerated type associated with the domain of the i^{th} attribute of *r*, and (b) declaring a set of records of this type. For instance, the translation of the database containing the facts *node*(*a*), *node*(*b*), *node*(*c*), *node*(*d*), *edge*(*a*,*b*), *edge*(*a*,*c*), *edge*(*b*,*c*), and *edge*(*c*,*d*) consists of the following OPL declarations.

```
enum node {a, b, c, d};
struct edge_type {
     node s;
     node t;
};
setof(edge_type) edge = {<a,b>, <a,c>, <b,c>,
<c,d>};
```

Query Translation. First, as IDB predicates are associated with Boolean arrays, the following predefined type *Boolean* is declared.

```
range boolean 0..1
```

Then, for each IDB predicate p with arity k, a k-dimensional Boolean array of variables is introduced as follows:

var boolean p[D_1, …, D_k],

where $D_1, \ldots, D_k$ denote the domains on which the predicate p is defined. For instance, for the binary predicate *col* of Example 3, the declaration

```
var boolean col[node, color];
```

is introduced.

A search (resp. optimization) NP Datalog query Q is translated into an OPL search (resp. optimization) program by translating each rule of Q into corresponding OPL statements.

A subset rule of the form $s(X_1, \ldots, X_k) \subseteq body(X_1, \ldots, X_k, Y_1, \ldots, Y_n)$ is translated into an OPL statement specifying that if $s(X_1, \ldots, X_k)$ is true, then $body(X_1, \ldots, X_k, Y_1, \ldots, Y_n)$ must be true too for some values of $Y_1, \ldots, Y_n$. For instance, the subset rule $e(x,y) \subseteq edge(x,y)$ is translated into the following OPL statement.

```
forall (x in node, y in node)
     e[x, y] ⇒ (sum(<x,y> in edge) 1 > 0);
```

The above OPL constraint states that if $e[x,y]$ is true, then the pair $<x,y>$ must occur in relation *edge*, hence it ensures that the subset e is defined on the existing edges.

A generalized partition rule $\oplus_L s(X_1, \ldots, X_k, L) \leftarrow body(X_1, \ldots, X_k, Y_1, \ldots, Y_n), d(L)$, defining the guess predicate s, is translated into OPL instructions stating that (a) if the body is true, then $s(X_1, \ldots, X_k, L)$ must be true too for a unique value of L in the base relation d, and (b) if $s(X_1, \ldots, X_k, L)$is true, then the body must be true too for some values of $Y_1, \ldots, Y_n$. For instance, the partition rule of Example 3 is translated as follows.

```
forall (x in node)
     1 ⇒ (sum(c in color) col[x, c] = 1);
forall (x in node, c in color)
     col[x, c] ⇒  1;
```

Here, the first constraint guarantees for each node x the assignment of exactly one color c, whereas the second one ensures that the coloring is defined on the existing nodes. Note that this code can be optimized (e.g., the second constraint can be omitted as it is always satisfied).

A standard rule of the form $p(X_1, \ldots, X_k) \leftarrow Body_1(X_1, \ldots, X_k Y_1^1, \ldots, Y_{n\_1}^1) \vee \ldots \vee Body_m(X_1, \ldots, X_k, Y_1^m, \ldots, Y_{n\_m}^m)$ is translated into OPL instructions stating that the rule head is true if and only if the rule body is true. For instance, the standard rule of Example 3 is translated into the OPL code as follows.

```
forall(c in color)
     used_color[c] ⇔ (sum(x in node) col[x, c]
> 0);
```

The above constraint guarantees that a color c is used if and only if there exists a node x colored with c.

A constraint of the form $A \Leftarrow B$ is translated into an OPL statement declaring that if B is true, then A must be true too. For instance, the constraint of Example 3 is translated into the following statement.

```
forall (x in node, y in node, c in color)
          (((sum(<x,y> in edge) 1 > 0) * col[x, c]
* col[y, c]) > 0) ⇒ 0;
```

The constraint ensures that no two connected nodes *x* and *y* are colored with the same color *c*.

The number of (ground) constraints can be strongly reduced by applying simple optimizations to the OPL code (Greco, Molinaro, Trubitsyna, & Zumpano, 2006).

Example 4. Below is reported a compact and efficiently solvable OPL program expressing the minimum coloring problem. It is obtained by translating the NP Datalog query of Example 3 and by applying several optimizations.

```
var color col[node];
var boolean used_color[color];

minimize sum(c in color) used_color[c]
subject to {
        forall (c in color)
              used_color[c] ⇔ (sum(x in
    node) (col[x] = c) > 0);
        forall (<x,y> in edge)
              ((col[x] = col[y]) > 0) ⇒ 0;
};
```

A system prototype was carried out by implementing modules translating NP Datalog queries into OPL programs and using the OPL ILOG system to execute the target code. Several experiments show the effectiveness of this approach (Greco et al., 2006).

CONCLUSION

NP search and optimization problems can be formulated as DATALOG¬ queries under nondeterministic stable-model semantics. In order to enable a simpler and more intuitive formulation of these problems, the NP Datalog language, allowing search and optimization queries to be expressed using only simple forms of unstratified negations, has been proposed. This entry has presented the implementation of the language, which is based on the translation of NP Datalog queries into OPL programs that are evaluated by means of the ILOG OPL Studio. This approach combines an easy formulation of problems, expressed by means of a declarative logic language and an efficient execution of the ILOG solver.

REFERENCES

Abiteboul, S., Hull, R., & Vianu, V. (1994). *Foundations of databases.* Addison-Wesley.

Finkel, R. A., Marek, V. W., & Truszczynski, M. (2004). Constraint lingo: Towards high-level constraint programming. *Software: Practice and Experience* (pp. 1481-1504).

Gelfond, M., & Lifschitz, V. (1988). The stable model semantics for logic programming. *International Conference on Logic Programming* (pp. 1070-1080).

Gelfond, M., & Lifschitz, V. (1991). Classical negation in logic programs and disjunctive databases. *New Generation Computing, 9*(3/4), 365-385.

Greco, S., Molinaro, C., Trubitsyna, I., & Zumpano, E. (2006). Implementation and experimentation of the logic language NP datalog. *International Conference on Database and Expert Systems Applications* (pp. 622-633).

Greco, S., & Saccà, D. (1997). NP-optimization problems in datalog. *International Logic Programming Symposium* (pp. 181-195).

Johnson, D. S. (1990). A catalog of complexity classes. In J. van Leewen (Ed.), *Handbook of theoretical computer science* (Vol. 1, pp. 67-161). North-Holland.

Kolaitis, P. G., & Thakur, M. N. (1994). Logical definability of NP optimization problems. *Information and Computation, 115,* 321-353.

Leone, N., Pfeifer, G., Faber, W., Calimeri, F., Dell'Armi, T., Eiter, T., et al. (2002). The DLV system. *European Conference on Logics in Artificial Intelligence* (pp. 537-540).

SICStus prolog. (n.d.). Retrieved from http://www.sics.se/sicstus/

Syrjanen, T., & Niemela, I. (2001). The Smodels system. *International Conference on Logic Programming and Nonmonotonic Reasoning* (pp. 434-438).

Van Hentenryck, P. (1999). *The OPL optimization programming language.* MIT Press.

Van Hentenryck, P., Michel, L., Perron, L., & Regin, J. C. (1999). Constraint programming in OPL. *International Conference on Principles and Practice of Declarative Programming* (pp. 98-116).

Van Roy, P. (1999). Logic programming in Oz with Mozart. *International Conference on Logic Programming* (pp. 38-51).

Wallace, M., & Schimpf, J. (1999). ECLiPSe: Declarative specification and scaleable implementation. *International Workshop on Practical Aspects of Declarative Languages* (pp. 365-366).

Zumpano, E., Greco, S., Trubitsyna, I., & Veltri, P. (2004). On the semantics and expressive power of datalog-like languages for NP search and optimization problems. *ACM Symposium on Applied Computing* (pp. 692-697).

KEY TERMS

Answer Set Programming (ASP): A programming paradigm that uses logic to express and solve search problems. Given a search problem, a logic theory can be designed so that models of this theory represent problem solutions.

Constraint Logic Programming (CLP): A programming paradigm that extends logic programming by a mechanism for constraints modeling and processing.

Constraint Programming (CP): A programming paradigm that expresses a problem by means of a set of constraints that must be satisfied by the solutions of the problem.

Disjunctive Datalog Program: A set of (disjunctive datalog) rules of the form

$$A_1 \vee \ldots \vee A_k \leftarrow B_1, ..., B_m, not\ B_{m+1}, \ldots, not\ B_n, \quad k+n>0,$$

where $A_1, \ldots, A_k, B_1, \ldots, B_n$ are atoms of the form $p(t_1, ..., t_h)$, p is a predicate symbol of arity h, and the terms $t_1,..., t_h$ are constants or variables. A not-free ($\vee$-free) program is called positive (or normal). Predicate symbols are partitioned into two distinct sets: base predicates (also called EDB predicates) and derived predicates (also called IDB predicates). Base predicates correspond to database relations defined over a given domain and they do not appear in the head of any rule; derived predicates are defined by means of rules.

Optimization Programming Language (OPL): A modeling language to represent optimization problems.

SAT: The Boolean satisfiability problem. It consists of determining a satisfying variable assignment, V, for a Boolean function, f, or determining that no such V exists. SAT is one of the central NP-complete problems.

Stable-Model Semantics: Declarative semantics for logic programs with negation.

Chapter LXXXV
A Query-Strategy-Focused Taxonomy of P2P IR Techniques

Alfredo Cuzzocrea
University of Calabria, USA

INTRODUCTION

During the last years, there was a growing interest in peer-to-peer (P2P) systems, mainly because they fit a wide number of real-life ICT applications. Digital libraries are only a significant instance of P2P systems, but it is very easy to foresee how large the impact of P2P systems on innovative and emerging ICT scenarios, such as e-government and e-procurement, will be during the next years.

P2P networks are natively built on top of a very large repository of data objects (e.g., files) that is intrinsically distributed, fragmented, and partitioned among participant peers. P2P users are usually interested in (a) retrieving data objects containing information of interest, like video and audio files, and (b) sharing information with other (participant) users or peers. From the information retrieval (IR) perspective, P2P users (a) typically submit short, loose queries by means of keywords derived from natural-language-style questions (e.g., "find all the music files containing Mozart's compositions" is posed through the keywords *compositions* and *Mozart*), and (b), due to resource-sharing purposes, are usually interested in retrieving as a result a set of data objects rather than only one. Based on such set of items, well-founded IR methodologies like ranking can be successfully applied to improve system query capabilities, thus achieving performance better than that of more traditional database-like query schemes. Furthermore, the above-described P2P IR mechanism is self-alimenting as intermediate results can be then reused to share new information, or to set and specialize new search and query activities. In other words, from the database perspective, P2P users typically adopt a semistructured (data) model for querying data objects rather than a structured (data) model. On the other hand, efficiently accessing data in P2P systems, which is an aspect directly related to the above issues, is a relevant and still incompletely solved open research challenge.

Traditional functionalities of first-generation P2P systems are currently being extended by adding to their native capabilities (i.e., file sharing primitives and simple lookup mechanisms based on partial or exact match of search strings) useful (and more complex) knowledge representation and extraction techniques. Achieving the defini-

tion of new knowledge delivery paradigms over P2P networks is the underlying goal of this effort; in fact, the completely decentralized nature of P2P networks, which enable peers and data objects to come and go at will, allows us to (a) successfully exploit self-alimenting mechanisms of knowledge production, and (b) take advantages from innovative knowledge representation and extraction models based on semantics, metadata management, probability, and so forth. All considering, we can claim that, presently, there is a strong, effective demand for enriching P2P systems with functionalities that are proper of IS, such as knowledge discovery (KD) and IR-style data object querying, and cannot be supported by the actual data representation and query models of traditional P2P systems. More specifically, knowledge representation and management techniques mainly concern the modeling of P2P systems, whereas knowledge discovery techniques (implemented via IR functionalities) mainly concern the querying (i.e., knowledge extraction) of P2P systems.

Following this trend, a plethora of P2P IR techniques have been proposed recently, each of them focused on covering a particular or specific aspect of the KD phase. A meaningful way of studying P2P IR techniques under a common plan is looking at their query strategies used to retrieve information and knowledge. In fact, despite the implementation and architectural details, the underlying query strategy is the most relevant characteristic of any P2P IR technique, mainly from the database research perspective.

According to these considerations, in this article we provide a taxonomy of state-of-the-art P2P IR techniques, which emphasize the query strategy used to retrieve information and knowledge from peers, and put in evidence similarities and differences among the investigated techniques. This taxonomy helps us to keep track of the large number of proposals that have come up in the last years, and to support future research in this leading area.

BACKGROUND

The first experiences of P2P systems, such as Gnutella (*The Gnutella File Sharing System*, 2006), KaZaA (*The KaZaA File Sharing System*, 2006), and *Napster* (*The Napster File Sharing System*, 2006), which mainly focused on data management issues on P2P networks, were oriented toward designing techniques for which sharing data objects and generating large communities of participant peers were the most relevant goals. Under this assumption, two reference architectures have gained a leading role for P2P systems, each of them addressing two different ways of retrieving data objects by querying: unstructured P2P systems and structured P2P systems.

As regards the unstructured P2P systems, there are three main variants. In the first one (e.g., *Napster*, 2006), a centralized index storing a directory of all data objects currently available on the P2P network is located in a certain peer, whose identity is known to all the peers. When a participant peer p_i receives a request for a missing data object, it (a) performs a query against the peer containing the centralized index for retrieving the name of the (participant) peer p_j where the required data object is stored, and (b) redirects the request toward p_j. In the second variant (e.g., *Gnutella*, 2006), there is no centralized index as it can be the source of failures; each participant peer needs to maintain only information about his or her own data for supporting data object lookups, and information about neighboring peers for routing requests coming from other peers. Given such a scheme, a request for a missing data object is flooded from a peer p_i toward other peers via the neighboring peers of p_i. In the last variant (e.g., *KaZaA*, 2006), peers connect to a super-peer who builds an index over the data objects shared by his or her set of peers. In addition to this, each super-peer keeps information about neighboring super-peers in the system, and queries are routed among super-peers. Scalability is the most important drawback for unstructured P2P systems: In fact, when the number of participant peers grows, the described query mechanism can become very

inefficient as flooding the P2P network for each data object request can became (very) resource intensive.

In structured P2P systems, all the available data objects are indexed by a high-performance indexing data structure (such as distributed hash tables [DHTs]) that is distributed among the participant peers. Some examples of P2P systems adhering to such architecture are Chord (Stoica, Morris, Karger, Frans Kaashoek, & Balakrishnan, 2001), Content-Addressable Network (CAN; Ratnasamy, Francis, Handley, Karp, & Shenker, 2001), Tapestry (Zhao, Kubiatowicz, & Joseph, 2001), and XPath for P2P (XP2P; Bonifati & Cuzzocrea, 2006; Bonifati, Cuzzocrea, Matrangolo, & Jain, 2004). Compared to the previous class of P2P systems, structured P2P systems introduce the important advantage that, at query time, they do not flood the requests across the P2P network, but, indeed, the required data objects are quickly localized thanks to the distributed index. As a consequence, structured P2P systems are highly scalable. Nevertheless, an important limitation is represented by the need for updating the distributed index, particularly when highly dynamic P2P networks are handled as dynamics of peers quickly modify the network topology with a very high churn rate, which refers to the tendency of peers to join and leave the network (Gummadi, Dunn, Saroiu, Gribble, Levy, & Zahorjan, 2003).

P2P INFORMATION RETRIEVAL TECHNIQUES: A QUERY-STRATEGY-FOCUSED TAXONOMY

It is well established (e.g., Tsoumakos & Roussopoulos, 2003b; Zeinalipour-Yazti, Kalogeraki, & Gunopulos, 2004) that P2P systems are mainly characterized by their proper query strategies allowing us to retrieve useful knowledge in the form of data objects (e.g., files). According to this vision, in this section, we propose a query-strategy-focused taxonomy of state-of-the-art P2P IR techniques (summarized results are shown in Table 1).

Table 1. A query-strategy-focused taxonomy of IR techniques for P2P systems

Class	***Search Kind***	***P2P IR Techniques***
BST	KbP2PS	*Gnutella*, 2006; *KaZaA*, 2006; *Napster*, 2006
RST	KbP2PS	Kalogeraki, Gunopulos, & Zeinalipour-Yazti, 2002; Lv, Cao, Cohen, Li, & Shenker, 2002; Tsoumakos & Roussopoulos, 2003a
IST	KbP2PS	Gen Yee & Frieder, 2005; Meng, Yu, & Liu, 2002; Sripanidkulchai, Maggs, & Zhang, 2003; Zeinalipour-Yazti, Kalogeraki, & Gunopulos, 2005
SST	KbP2PS	Galanis, Wang, Jeffery, & DeWitt, 2003; Gong, Yan, Qian, & Zhou, 2005; Koloniari & Pitoura, 2004; Yang & Garcia-Molina, 2002
IndST	OIDbP2PS	Aberer, 2001; Bonifati & Cuzzocrea, 2006; Bonifati et al., 2004; Bremer & Gertz, 2003; Clarke, Sandberg, Wiley, & Hong, 2000; Crainiceanu, Linga, Gehrke, & Shanmugasundaram, 2004; Crespo & Garcia-Molina, 2002; Gibbons, Karp, Ke, Nath, & Seshan, 2003; Gupta, Agrawal, & El Abbadi, 2003; Loo, Huebsch, Hellerstein, Stoica, & Shenker, 2004; *Morpheus File Sharing System*, n.d.; Ratnasamy et al., 2001; Rhea et al., 2001; Sartiani, Manghi, Ghelli, & Conforti, 2004; Stoica et al., 2001; Zhao et al., 2001
DIRT	KbP2PS	Callan, 2000; Gauch, Wang, & Rachakonda, 1999; Ogilvie & Callan, 2001; Xu & Callan, 1998
SemST	KbP2PS	Cai & Frank, 2004; Crespo & Garcia-Molina, 2003; Cuzzocrea, 2005, 2006; Deerwester, Dumais, Furnas, Landauer, & Harshman, 1999; Golub & Loan, 1996; Halaschek, Aleman-Meza, Arpinar, & Sheth, 2004; Halevy, Ives, Mork, & Tatarinov, 2003; Kokkinidis & Christophides, 2004; Li, Lee, & Sivasubramaniam, 2003; Nejdl et al., 2003; Salton & Buckley, 1990; Salton, Wang, & Yang, 1975; Tang, Xu, & Dwarkadas, 2003; Zhang, Berry, & Raghavan, 2001; Zhu, Cao, & Yu, 2006

The first general classification distinguishes between keyword-based P2P (KbP2PS) and object-identifier-based P2P (OIDbP2PS) systems by looking at the atomic construct they use to drive the search mechanism. In KbP2PS, traditional keywords are used to drive the search through peers whereas in OIDbP2PS, object identifiers are implemented on peers to enhance IR performance by biasing the search toward specific subsets of peers (in particular, due to the decentralized nature of P2P systems, object identifiers are usually embedded into distributed indexing data structures such as DHTs). It should be noted that both KbP2PS and OIDbP2PS can be implemented on top of either unstructured or structured P2P systems, meaning that the search mechanism is independent of the specific system topology even if different performance is achieved. Second, state-of-the-art proposals are classified according to their query strategies concerning how to route (through peers) messages needed to answer queries, thus retrieving information and knowledge.

Basic Search Techniques

Among the basic search techniques (BSTs), the breadth first search (BFS) is one of the most popular ways of supporting IR over P2P networks. In BFS, a peer p_i receiving a query message q from a sender peer p_j first forwards q to all its neighboring peers other than p_j, and then searches its local repository for relevant matches. Furthermore, if a peer p_k reached by q finds a match in its repository, it sends across the network the hit message along with the identifiers needed to download from it the data objects of interest and the state of its network connectivity. Finally, if p_j receives hit messages from more than one peer, it may decide to download the retrieved documents from peers on the basis of their network connectivity states. BFS, which has been one of the first query strategies implemented in P2P networks, such as in Gnutella (*Gnutella*, 2006), KaZaA (*KaZaA*, 2006), and Napster (*Napster*, 2006), presents the advantage of being very simple so that no excessive computational overheads are introduced in the middleware software. Contrary to this, BFS performance is usually very poor due to inefficient network resource utilization that generates a lot of service messages across the P2P network: Peers with low bandwidth can become serious bottlenecks for such a search mechanism. However, equipping query messages with the time-to-live (TTL) parameter, which determines the maximum number of hops allowed for any query message, can limit network flooding and sensitively increase performance.

Random Search Techniques

Random search techniques (RSTs) represent a simple yet effective derivation from BST. Kalogeraki et al. (2002) propose a significant extension to the naïve version of BFS called random breadth first search (RBFS), which consists of propagating the query message from a peer to a randomly determined subset of its neighboring peers rather than all of them. A setting parameter establishes how many neighboring peers must be involved in the propagation of the query message (e.g., if the parameter is equal to 0.5, then the query message is propagated to half of all the neighboring peers, chosen at random). The major benefit of the RBFS approach consists of the fact that the performance of the BFS approach is dramatically improved while ensuring low computational overheads because the random choice does not require any global knowledge. On the other hand, RBFS being a probabilistic technique, it could happen that large segments of the P2P network are neglected by the random choice, thus reducing the efficiency of the query task. Lv et al. (2002) present the random walkers algorithm (RWA), according to which each peer forwards the query message (called, in this context, a walker) to another of its neighboring peers at random. To improve performance and reduce query time, the original idea of using one walker only is extended to the usage of $k > 1$ walkers that are consecutively sent from the sender peer. RWA resembles RBFS but in RBFS the query message is propagated to a subset of peers instead of only one at a time (as in RWA); as a consequence, in RBFS the number of

service messages across the P2P network can become exponential, whereas in RWA such a number is bounded by a linear complexity (Lv et al.). The adaptive probabilistic search (APS), proposed by Tsoumakos and Roussopoulos (2003a), is inspired by RWA with the difference that in APS each peer p_i implements a local data structure that captures the relative probability of each neighboring peer p_j to be chosen as the next hop for future requests. Furthermore, while RWA forwards the walkers at random, APS exploits knowledge derived from previous searches to model the behavior of walkers on a probabilistic basis. In Tsoumakos and Roussopoulos (2003a), experimental results presented by authors show that APS outperforms RWA.

Intelligent Search Techniques

Beyond the previous basic approaches, another line of research aims at integrating intelligent techniques, perhaps inherited from similar experiences in related but different scientific disciplines, into the P2P middleware as to enforce the quality of the search task. We name such a class of proposals as intelligent search techniques (ISTs). The intelligent search mechanism (ISM), proposed by Zeinalipour-Yazti et al. (2005), belongs to this technique class and represents a novel approach for supporting IR over P2P networks by minimizing the number of messages sent among the peers and minimizing the number of peers that are involved for each search request. To this end, ISM is composed of (a) a profile mechanism, according to which each peer builds a profile for each of its neighboring peer, (b) a query similarity function, which calculates the similarity queries have to a new query, (c) a relevance rank, which is a ranking technique for peers that takes as input the (neighboring) peer profiles, and produces as output a ranked list of (neighboring) peers used to bias the search toward the most relevant peers, and (d) a search mechanism, which implements the ISM search policy. In Zeinalipour-Yazti et al., the authors show how ISM works well over P2P networks when peers hold some specialized knowledge about the P2P environment, and when P2P networks have high degrees of query locality; in these particular conditions, ISM outperforms BFS as well as RBFS techniques. Other techniques that can be classified as belonging to the IST class are the framework proposed by Gen Yee and Frieder (2005), which combines ranking techniques and metadata management for efficiently supporting IR over P2P networks; the metasearch engines by Meng et al. (2002), mainly focused on source selection and the merging of results from independent sources; and the system proposed by Sripanidkulchai et al. (2003), which discriminates among peers based on their past behavior in order to form communities of peers having similar interests. The main limitation of IST is that, being usually implemented inside the P2P middleware directly, they could become resource consuming and, in general, do not scale well on large and dynamic P2P networks.

Statistics-Based Search Techniques

Statistics-based search techniques (SSTs) are another important result for IR over P2P networks: They use some aggregated statistics to forward queries toward a particular subset of peers; usually, statistics is maintained by mining results of past queries. Techniques belonging to such a class are the most-results-in-past (>RES) heuristic, proposed by Yang and Garcia-Molina (2002), where query messages are routed to those peers who returned the most results for the last *m* queries, *m* being a technique parameter (it should be noted that, in this case, the statistics employed is very simple, being based on a quantitative approach); Galanis et al.'s (2003) data summaries and histograms, which are built on each peer by means of data replication techniques in order to exploit such information at query time to bias the search toward the most relevant peers; Gong et al.'s (2005) bloom filters; and Koloniari and Pitoura's (2004) multilevel bloom filters, which are statistics-inspired compressed data structures able to summarize the data of the neighborhood of a given peer, and to guide query answering over P2P networks by means of probabilistic algorithms.

Even if statistics allows the performance of certain kinds of queries (e.g., range queries; Gupta et al., 2003) to be improved when compared against traditional approaches, maintaining summary data structures over P2P networks can become very resource intensive and thus unfeasible in real-life scenarios.

Index-Based Search Techniques. Index-based search techniques (IndSTs) efficiently exploit the hierarchical nature of structured P2P networks, and extensively use and take advantages from well-known data indexing solutions coming from the RDBMS (relational database management system) technology (e.g., B^+ trees and *R* trees). Among IndST proposals, we recall (a) Clarke et al.'s (2000) Freenet, which uses an intelligent indexing scheme based on the depth first search (DFS) mechanism to locate objects (note that Freenet has not been proposed for the context of P2P networks properly, but, indeed, for a general anonymous information storage and retrieval system), (b) Aberer's (2001) P-Grid, which provides extensions to traditional DHTs in order to efficiently support scalability, which is recognized as one of the critical factors for P2P systems, (c) Ratnasamy et al.'s (2001) CAN, which employs a *d*-dimensional Cartesian space to index resources on P2P networks, (d) Stoica et al.'s (2001) *Chord*, which uses a linear space of identifiers forming a ring, exploited to speed up lookup operations over P2P networks, (e) Zhao et al.'s (2001) Tapestry, mainly focused on supporting fault-tolerant functionalities over large P2P networks, (f) the hybrid technique for building and maintaining local indices proposed by Crespo and Garcia-Molina (2002), who present three different methods (namely, compound routing index [CRI], hop-count routing index [HRI], and exponentially aggregated routing index [ERI]) that generate indexing data structures able to store the direction toward relevant documents over P2P networks by exploiting original ideas about routing protocols developed and tested in Arpanet, (g) Gibbons et al.'s (2003) IrisNet, which is an architecture for supporting IR over P2P networks storing XML (extensible markup language) data indexed by XPath expressions; (h) the locality sensing hashing (LSH) technique by Gupta et al. (2003), who propose using horizontal partitions of relational tables and extensions to the original DHT for supporting approximate range query answering over P2P networks, (i) the Bremer and Gertz (2003) distributed indexes (namely, *P*-index, *A*-index, and *T*-index), which are oriented to build virtual XML repositories over P2P networks by encoding global path information, and efficiently supporting global query answering over them, (j) *P* trees by Crainiceanu et al. (2004), who propose augmenting the DHT with B^+ trees in order to support range query answering on relational P2P databases, (k) Loo et al.'s (2004) PierSearch, which exports RDBMS features in an Internet-scale P2P environment, (l) Sartiani et al.'s (2004) XPeer, which is targeted at XML data and uses full tree guides to perform query evaluation, and (m) XP2P, proposed by Bonifati and Cuzzocrea (2006) and Bonifati et al. (2004), where light-weight XPath expressions are encoded in few Kb by means of Rabin's fingerprints in order to build a (lightweight) distributed index for efficiently supporting lookup queries over structured P2P networks. Just like SST, even if query capabilities are improved and well supported, including new paradigms that were missing in first-generation P2P systems (such as range queries), IndST mainly suffers from scalability limitations, and updating distributed indexes over large P2P networks is still an open problem.

Distributed Information Retrieval Techniques

The distributed information retrieval (DIR) approach (Callan, 2000; Gauch et al., 1999; Ogilvie & Callan, 2001; Xu & Callan, 1998), first proposed in contexts different from P2P systems, assumes that peers have a global knowledge of the system, for example, statistical knowledge about the content of each data repository in the network, or intensional (i.e., schema-based) knowledge as in Cai and Frank (2004), Halaschek et al. (2004), Halevy et al. (2003), Nejdl et al. (2003), and Tang et al. (2003). In contrast, most of the actual

P2P IR techniques assume that peers only have a local knowledge of the system, e.g. knowledge about their neighboring peers. This is mainly due to the fact that distributed information retrieval techniques (DIRTs), being based on the assumption of holding (and maintaining) global views of the system through peers, could become very inefficient in large and dynamic P2P networks.

Semantics-Based Search Techniques

Semantics-based search techniques (SemSTs) are the new frontier for IR over P2P networks. Such techniques aim at adopting formal semantics to both model and query distributed resources over P2P networks in order to improve the capabilities of traditional resource-sharing P2P systems. The first advantage of SemST is the amenity of reusing and readopting well-founded results coming from semantic models and query languages. Another advantage consists of the possibility of meaningfully integrating IR techniques in P2P networks with leading new research trends like ontologies and the semantic Web. In literature, there are very few proposals addressing the described research challenges since most papers are still focused on query performances of unstructured and structured P2P systems. However, it is expected that integrating semantics in P2P networks will be one of the most relevant research topics for next-generation ICT applications.

First, Crespo and Garcia-Molina (2003) propose the notion of semantics overlay networks (SONs), which are an efficient way of grouping together peers that share the same schema information. Therefore, peers having one or more topics on the same thematic hierarchy belong to the same SON. This approach well supports query routing as every peer p_i can quickly identify peers containing relevant information, namely, the set $N(p_i)$, by avoiding network flooding. Here, being relevant means that a certain semantic relation exists between information held in p_i and information held in peers belonging to $N(p_i)$. Such semantic hierarchies are naturally represented (and processed) via the resource description framework (RDF) by also taking advantages from several declarative languages for querying and defining views over RDF bases, such as the RDF query language (RQL; Karvounarakis, Alexaki, Christophides, Plexousakis, & Scholl, 2002) and the RDF view language (RVL; Magkanaraki, Tannen, Christophides, & Plexousakis, 2003). In Crespo and Garcia-Molina (2003), the authors demonstrate that SON can significantly improve query performances while at the same time allowing users to decide what content to publish in their (peer) hosts, that is, how to form a SON. The SON initiative heavily influenced many P2P-focused research projects; among all, some of them are centered on query routing issues in SON by meaningfully using the potentialities of RDF constructs: RDFPeers by Cai and Frank (2004), SemDIS by Halaschek et al. (2004), Piazza by Halevy et al. (2003), Edutella by Nejdl et al. (2003), and self-organizing SON by Tang et al. (2003). All these initiatives have in common the idea of propagating queries across a (semantic) P2P system by means of semantics-based techniques such as correlation discovery, containment, and so forth mainly working on RDF-modeled networks of concepts.

Another significant proposal can be found in the ICS-FORTH SQPeer middleware proposed by Kokkinidis and Christophides (2004) who, starting from the same motivations of the SON initiative, propose more sophisticated information representation and extraction mechanisms by introducing (a) the concept of active RDF schemas for declaring the parts of a SON RDF schema that are currently of interest for a peer via an advertisement mechanism, and (b) the concept of semantic query pattern relying on query and view subsumption techniques that are able to guide query routing on the basis of semantics.

A possible limitation of SON is represented by the overwhelming volume of messages that can be generated for supporting data object replications on peers, as required by SON design guidelines (Crespo & Garcia-Molina, 2003). Thus, P2P applications running on top of the SON-based model for query routing incur excessive overheads on

network traffic. An interesting solution to this problem has been proposed by Li et al. (2003): They suggest using signatures on neighboring peers for directing searches along selected network paths and introduce some schemes to facilitate the efficient search of data objects. Signatures are a way of adding semantics to data by building a bit vector *V*; *V* is generated according to the following two steps: (a) Hash the content of a data object into bit strings, *BS*, and (b) apply a bitwise *or* operator on *BS*. The so-built bit strings are used at query time by performing a bitwise *and* operation on the search signature (i.e., the signature of the term used as a search key) and the data signature (i.e., the signature stored on the current peer). In Li et al., the authors show how some proposed flooding-based search algorithms allow the signatures of the neighboring peers to be efficiently exploited to enhance search results, and, in addition to this, an extensive experimental part clearly confirms the effectiveness of the neighborhood signature technique in comparison with existing P2P content search methods, including Gnutella (*Gnutella*, 2006), RWA (Lv et al., 2002), and the IndST of the systems Morpheus (*Morpheus*, 2006) and OceanStore (Rhea et al., 2001).

Latent semantic indexing (LSI), first proposed by Deerwester et al. (1999), is a state-of-the-art technique for supporting IR from collections of documents. It is based on the vector space model (VSM) proposed by Salton et al. (Salton et al., 1975) that represents a document collection as a term-by-document matrix $A_{t\times d}$ such that each element of $A_{t\times d}$ is a weight w_{ij} of the corresponding term t_i in the specific document d_j; w_{ij} is computed by taking into account the term frequency tf_{ij}, which captures the local relevance of t_i in d_j, and the inverted document frequency df_j, which models a sort of global statistic knowledge about d_j in the whole document collection (Salton et al.). Opposite of the original VSM approach, the main idea behind LSI is comparing users' queries and documents at the concept level (thus using semantics) rather than on the basis of simple keyword matching (as in VSM). In LSI, the VSM matrix $A_{t\times d}$ is factorized by the single value decomposition (SVD) technique, first proposed by Golub and Loan (1996), which, unfortunately, introduces an excessive computational cost when applied on huge document collections, as studied by Zhang et al. (2001). However, similar to DIRT, LSI requires that each peer or site holds global knowledge about the system, thus it becomes ineffective for large P2P networks. A possible solution to this problem can be found in the work of Zhu et al. (2006), who propose a novel query optimization scheme, called semantic dual query expansion (SDQE), where the lack of global knowledge is compensated by engaging ad hoc query optimization plans implemented on peers locally. SDQE is in turn based on the query expansion (QE) technique, initially proposed by Salton and Buckley (Salton & Buckley, 1990), which consists of refining users' queries by adding to them other terms related to those expressed by the original queries. Some works in the context of DIRT (e.g., Gauch et al., 1999; Ogilvie & Callan, 2001; Xu & Callan, 1998) show that DIRT performance can be improved thanks to QE. In other words, SDQE can be just considered as a sort of advanced query engine for DIRT, which allows us to mitigate the requirement of global knowledge posed by DIRT.

Finally, in Cuzzocrea (2005, 2006), we propose an innovative semantics-based framework for supporting KD- and IR-style resource querying on large-scale P2P XML repositories (e.g., those that one can find in corporate business-to-business and business-to-consumer e-commerce systems). In more detail, in Cuzzocrea (2006), we propose (a) modeling both XML repositories or documents and queries in terms of concepts they express by means of formal reasoning flat models, like lists, and hierarchical models, like graphs, and (b) applying ad hoc knowledge extraction algorithms that efficiently exploit such models. Our algorithms also enhance the semantic expressiveness of the reasoning task by exploiting the local knowledge given by mining, according to some meaningful two-dimensional abstractions, past (successful) query results flooded through neighboring peers (Cuzzocrea, 2006).

CONCLUSION

Querying P2P systems, which in this article has been investigated as a fundamental task for supporting advanced functionalities such as KD- and IR-style data and resource extraction, is an important research challenge for the database research community, still asking for solutions capable of capturing all the (complex) requirements posed by such a novel class of systems. As highlighted throughout the article, this challenge involves various aspects ranging from representation and reasoning issues to maintenance and query issues. At the same time, research topics discussed in this article play a leading role with respect to the efficiency and the effectiveness of a wide range of modern ICT applications, which also have a relevant impact on day-to-day real-life (e.g., e-government systems, e-procurement systems, etc). Given these considerations, we presented a query-strategy-focused taxonomy of state-of-the-art P2P IR techniques that put in evidence current research trends in supporting information and knowledge extraction from large, dynamic, data-intensive P2P systems.

FUTURE TRENDS

Adding preferences into query languages seems to be the next challenge for P2P IR research. Via preferences, users and applications can specify constraints over the final result, thus designing new paradigms for information and knowledge delivery over P2P networks. As an example, a user could request the list reporting administrative data on Central European suppliers that never sold parts in Italy and at the same time never imported parts from East Europe.

In order to efficiently support preference-aware IR functionalities, next-generation P2P systems must include inside their query layers novel models and algorithms capable of being aware about the concept of preference on the semantic plan as well as the query execution plan, thus overcoming limitations of actual P2P query engines. A possible solution could be obtained by integrating logical models and languages with consolidated P2P query routing algorithms, thus taking advantage in terms of flexibility and expressiveness of both the query definition and evaluation tasks.

Another interesting line of research for next-generation P2P IR techniques seems to be the probabilistic data processing initiative, which aims at embedding probability models and schemes inside current data representation and query evaluation solutions for P2P networks. This would allow us to meaningfully extend the capabilities of state-of-the-art P2P systems by including novel query paradigms and providing support for the querying of incomplete or inconsistent data and information sources, which are very popular in very large and highly dynamic P2P networks.

REFERENCES

Aberer, K. (2001). P-grid: A self-organizing access structure for P2P information systems. *Proceedings of the 6th International Conference on Cooperative Information Systems* (pp. 179-194).

Bonifati, A., & Cuzzocrea, A. (2006). Storing and retrieving XPath fragments in structured P2P networks. *Data & Knowledge Engineering*, 59(2), 247-269.

Bonifati, A., Cuzzocrea, A., Matrangolo, U., & Jain, M. (2004). XPath lookup queries in P2P networks. *Proceedings of the 6th ACM International Workshop on Web Information and Data Management, in conjunction with the 13th ACM International Conference on Information and Knowledge Management* (pp. 48-55).

Bremer, J.-M., & Gertz, M. (2003). On distributing XML repositories. *Proceedings of the 2003 ACM International Workshop on Web and Databases, in conjunction with the 2003 ACM International Conference on Management of Data* (pp. 73-78).

Cai, M., & Frank, M. (2004). RDFPeers: A scalable distributed RDF repository based on a structured peer-to-peer network. *Proceedings of the 13th*

International World Wide Web Conference (pp. 650-657).

Callan, J. (2000). Distributed information retrieval. In *Advances in information retrieval* (pp. 127-150). Kluwer Academic Publishers.

Clarke, I., Sandberg, O., Wiley, B., & Hong, T. W. (2000). Freenet: A distributed anonymous information storage and retrieval system. *Proceedings of the ICSI Workshop on Design Issues in Anonymity and Unobservability* (pp. 311-320).

Crainiceanu, A., Linga, P., Gehrke, J., & Shanmugasundaram, J. (2004). Querying peer-to-peer networks using p-trees. *Proceedings of the 2004 ACM International Workshop on Web and Databases, in conjunction with the 2004 ACM International Conference on Management of Data* (pp. 25-30).

Crespo, A., & Garcia-Molina, H. (2002). Routing indices for peer-to-peer systems. *Proceedings of the 22nd IEEE International Conference on Distributed Computing Systems* (pp. 23-34).

Crespo, A., & Garcia-Molina, H. (2003). *Semantic overlay networks for P2P systems* (Tech. Rep.). Stanford University, Computer Science Department.

Cuzzocrea, A. (2005). Towards a semantics-based framework for KD- and IR-style resource querying on XML-based P2P information systems. *Proceedings of the 2005 IEEE/WIC/ACM International Conference on Web Intelligence* (pp. 58-61).

Cuzzocrea, A. (2006). On semantically-augmented XML-based P2P information systems. In *Lecture notes in artificial intelligence: Vol. 4027. Proceedings of the 7th International Conference on Flexible Query Answering Systems* (pp. 441-457).

Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K., & Harshman, R. (1999). Indexing by latent semantic analysis. *Journal of the American Society for Information Science, 41*(6), 391-407.

Galanis, L., Wang, Y., Jeffery, S. R., & DeWitt, D. J. (2003). Locating data sources in large distributed systems. *Proceedings of the 29th International Conference on Very Large Data Bases* (pp. 874-885).

Gauch, S., Wang, J., & Rachakonda, S. M. (1999). A corpus analysis approach for automatic query expansion and its extension to multiple databases. *ACM Transactions on Information Systems, 17*(3), 250-269.

Gen Yee, W., & Frieder, O. (2005). On search in peer-to-peer file sharing systems. *Proceedings of the 20th ACM Symposium on Applied Computing* (pp. 1023-1030).

Gibbons, P. B., Karp, B., Ke, Y., Nath, S., & Seshan, S. (2003). IrisNet: An architecture for a world-wide sensor web. *IEEE Pervasive Computing, 2*(4), 22-33.

The Gnutella file sharing system. (2006). Retrieved from http://gnutella.wego.com

Golub, G. H., & Loan, C. F. V. (1996). *Matrix computation.* The John Hopkins University Press.

Gong, X., Yan, Y., Qian, W., & Zhou, A. (2005). Bloom filter-based XML packets filtering for millions of path queries. *Proceedings of the 21st IEEE International Conference on Data Engineering* (pp. 890-901).

Gummadi, P. K., Dunn, R. J., Saroiu, S., Gribble, S. D., Levy, H. M., & Zahorjan, J. (2003). Measurement, modeling, and analysis of a peer-to-peer file-sharing workload. *Proceedings of the 19th ACM Symposium on Operating Systems Principles* (pp. 314-329).

Gupta, A., Agrawal, D., & El Abbadi, A. (2003). Approximate range selection queries in peer-to-peer systems. *Proceedings of the 1st Biennial Conference on Innovative Data Systems Research.* Retrieved from http://www.db.cs.wisc.edu/cidr/cidr2003/program/p13.pdf

Halaschek, C., Aleman-Meza, B., Arpinar, I. B., & Sheth, A. P. (2004). Discovering and ranking

semantic associations over a large RDF metabase. *Proceedings of the 30th International Conference on Very Large Data Bases* (pp. 1317-1320).

Halevy, A. Y., Ives, Z. G., Mork, P., & Tatarinov, I. (2003). Piazza: Data management infrastructure for semantic Web applications. *Proceedings of the 12th International World Wide Web Conference* (pp. 556-567).

Kalogeraki, V., Gunopulos, D., & Zeinalipour-Yazti, D. (2002). A local search mechanism for peer-to-peer networks. *Proceedings of the 11th ACM International Conference on Information and Knowledge Management* (pp. 300-307).

Karvounarakis, G., Alexaki, S., Christophides, V., Plexousakis, D., & Scholl, M. (2002). RQL: A declarative query language for RDF. *Proceedings of the 11th International World Wide Web Conference* (pp. 592-603).

The KaZaA file sharing system. (2006). Retrieved from http://www.kazaa.com

Kokkinidis, G., & Christophides, V. (2004). Semantic query routing and processing in P2P database systems: The ICS-FORTH SQPeer middleware. *Proceedings of the 2004 International Workshop on Peer-to-Peer Computing and Databases, in conjunction with the 9th International Conference on Extending Database Technology* (pp. 486-495).

Koloniari, G., & Pitoura, E. (2004). Content-based routing of path queries in peer-to-peer systems. *Proceedings of the 9th International Conference on Extending Database Technology* (pp. 29-47).

Li, M., Lee, W.-C., & Sivasubramaniam, A. (2003). Neighborhood signatures for searching P2P networks. *Proceedings of the 7th IEEE International Database Engineering and Applications Symposium* (pp. 149-159).

Loo, B. T., Huebsch, R., Hellerstein, J. M., Stoica, I., & Shenker, S. (2004). Enhancing P2P file-sharing with an Internet-scale query processor. *Proceedings of the 30th International Conference on Very Large Data Bases* (pp. 432-443).

Lv, Q., Cao, P., Cohen, E., Li, K., & Shenker, S. (2002). Search and replication in unstructured peer-to-peer networks. *Proceedings of the 16th ACM International Conference on Supercomputing* (pp. 84-95).

Magkanaraki, A., Tannen, V., Christophides, V., & Plexousakis, D. (2003). Viewing the semantic Web through RVL lenses. *Proceedings of the 2nd International Semantic Web Conference* (pp. 96-112).

Meng, W., Yu, C., & Liu, K.-L. (2002). Building efficient and effective metasearch engines. *ACM Computing Surveys, 34*(1), 48-84.

Morpheus file sharing system. (n.d.). Retrieved from http://www.musiccity.com

The Napster file sharing system. (2006). Retrieved from http://www.napster.com

Nejdl, W., Wolpers, M., Siberski, W., Schmitz, C., Schlosser, M., Brunkhorst, I., et al. (2003). Super-peer-based routing and clustering strategies for RDF-based P2P networks. *Proceedings of the 12th International World Wide Web Conference* (pp. 536-543).

Ogilvie, P., & Callan, J. (2001). The effectiveness of query expansion for distributed information retrieval. *Proceedings of the 10th ACM International Conference on Information and Knowledge Management* (pp. 183-190).

Ratnasamy, P., Francis, P., Handley, M., Karp, R., & Shenker, S. (2001). A scalable content-addressable network. *Proceedings of the 2001 ACM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications* (pp. 161-172).

Rhea, S., Wells, C., Eaton, P., Geels, D., Zhao, B., Weatherspoon, H., et al. (2001). Maintenance-free global data storage. *IEEE Internet Computing, 5*(5), 40-49.

Salton, G., & Buckley, C. (1990). Improving retrieval performance by relevance feedback. *Journal of the American Society for Information Science, 41*(4), 288-297.

Salton, G., Wang, A., & Yang, C. S. (1975). A vector space model for information retrieval. *Journal of American Society for Information Science, 18*(11), 613-620.

Sartiani, C., Manghi, P., Ghelli, G., & Conforti, G. (2004). XPeer: A self-organizing XML P2P database system. *Proceedings of the 2004 International Workshop on Peer-to-Peer Computing and Databases, in conjunction with the 9th International Conference on Extending Database Technology* (pp. 456-465).

Schmidt, A., Waas, F., Kersten, M., Carey, M., Manolescu, I., & Busse, R. (2002). XMark: A benchmark for XML data management. *Proceedings of the 28th International Conference on Very Large Data Bases* (pp. 974-985).

Sripanidkulchai, K., Maggs, B., & Zhang, H. (2003). Efficient content location using interest-based locality in peer-to-peer systems. *Proceedings of the 22nd Annual Joint Conference of the IEEE Computer and Communications Societies* (pp. 81-87).

Stoica, I., Morris, R., Karger, D., Frans Kaashoek, M., & Balakrishnan, H. (2001). Chord: A scalable peer-to-peer lookup service for Internet applications. *Proceedings of the 2001 ACM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications* (pp. 149-160).

Tang, C., Xu, Z., & Dwarkadas, S. (2003). Peer-to-peer information retrieval using self-organizing semantic overlay networks. *Proceedings of the 2003 ACM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications* (pp. 175-186).

Tsoumakos, D., & Roussopoulos, N. (2003a). Adaptive probabilistic search for peer-to-peer networks. *Proceedings of the 3rd IEEE International Conference on Peer-to-Peer Computing* (pp. 102-109).

Tsoumakos, D., & Roussopoulos, N. (2003b). A comparison of peer-to-peer search methods. *Proceedings of the 2003 ACM International Workshop on Web and Databases, in conjunction with the 2003 ACM International Conference on Management of Data* (pp. 61-66).

Xu, J., & Callan, J. (1998). Effective retrieval with distributed collections. *Proceedings of the 21st ACM International Conference on Research and Development in Information Retrieval* (pp. 112-120).

Yang, B., & Garcia-Molina, H. (2002). Efficient search in peer-to-peer networks. *Proceedings of the 22nd IEEE International Conference on Distributed Computing Systems* (pp. 5-14).

Zeinalipour-Yazti, D., Kalogeraki, V., & Gunopulos, D. (2004). Information retrieval techniques for peer-to-peer networks. *IEEE CiSE Magazine*, 12-20.

Zeinalipour-Yazti, D., Kalogeraki, V., & Gunopulos, D. (2005). Exploiting locality for scalable information retrieval in peer-to-peer systems. *Information Systems, 30*(4), 277-298.

Zhang, X., Berry, M. W., & Raghavan, P. (2001). Level search schemes for information filtering and retrieval. *Information Processing and Management, 37*(2), 313-334.

Zhao, Y. B., Kubiatowicz, J., & Joseph, A. (2001). *Tapestry: An infrastructure for fault-tolerant wide-area location and routing* (Tech. Rep. No. UCB/CSD-01-1141). University of California, Berkeley, Computer Science Division.

Zhu, X., Cao, H., & Yu, Y. (2006). SDQE: Towards automatic semantic query optimization in P2P systems. *Information Processing and Management, 42*(1), 222-236.

KEY TERMS

Peer: A site composing the P2P network.

Peer-to-Peer Network: A network environment where each site belonging to such network acts both as client (i.e., requiring services to server

sites) and as server (i.e., providing services to client sites).

Peer-to-Peer Protocol: The P2P network layer implementing communication and transmission functionalities of the network.

Resource: A data object (e.g., file) storing information of interest.

Structured P2P Network: A P2P network where the search mechanism is implemented via RDBMS-inspired indexing data structures (e.g., B^+ trees and R trees).

Super-Peer: In an unstructured P2P network, a peer indexing data and resources located in a domain of peers it controls.

Unstructured P2P Network: A P2P network where the search mechanism is implemented via flooding the network from a peer to another.

Chapter LXXXVI
Pervasive and Ubiquitous Computing Databases: Critical Issues and Challenges

Michael Zoumboulakis
University of London, UK

George Roussos
University of London, UK

INTRODUCTION

The concept of the so-called Pervasive and Ubiquitous Computing was introduced in the early nineties as the third wave of computing to follow the eras of the mainframe and the personal computer. Unlike previous technology generations, Pervasive and Ubiquitous Computing recedes into the background of everyday life: "it activates the world, makes computers so imbedded, so fitting, so natural, that we use it without even thinking about it, and is invisible, everywhere computing that does not live on a personal device of any sort, but is in the woodwork everywhere" (Weiser 1991). Pervasive and Ubiquitous Computing is often referred to using different terms in different contexts. Pervasive, 4G mobile and sentient computing or ambient intelligence also refer to the same computing paradigm. Several technical developments come together to create this novel type of computing, the main ones are summarized in Table 1 (Davies and Gellersen 2002; Satyanarayanan 2001).

Table 1. Elements of pervasive and ubiquitous computing

1. Physical and Virtual Integration	**Software Architectures** Adaptable, large scale, complex software infrastructures and development
Sensing Information gathering in the physical world and its representation in the virtual world	**Connectivity** High-speed, low power wired and wireless communications systems
Actuation Decisions in the virtual world and their tangible results in the physical world	**3. Deployment**
Awareness and Perception Using sensed data to maintain a higher-level model of the physical world and argue about	**Scalability** System adaptation to cater for massive scale in terms of space, devices and users
Ambient Displays Display information from the virtual world on physical artifacts	**Reliability** Security and redundancy technologies for continuous operation
World Modeling Representations of physical spaces	**Maintenance** Structured maintenance processes for reliability and updates
2. System Components	
Platforms Mobile, wearable or implantable hardware devices with small form factor	**Evaluation** Methods to evaluate the effectiveness of information systems outside the laboratory
Sensors and Actuators Hardware and software platforms for sensing and actuation	

BACKGROUND

One of the major challenges in turning the Pervasive and Ubiquitous Computing vision into reality is the development of distributed system architectures that will support effectively and efficiently the ability to instrument the physical world (Estrin *et al* 2002, National Research Council 2001). Such architectures are being developed around two core concepts: self-organizing networks of embedded devices with wireless communication capabilities and data-centricity. To augment physical artifacts with computational and communications capabilities it is necessary to enable miniaturized hardware components capable of wireless communication. However, these same characteristics that allow for instrumentation of physical objects also impose significant constraints. Systems architectures require significant changes due to the severely limited resources available on these devices. One possible solution is offered by the emergence of data-centric systems. In this context, data-centric refers to in-network processing and storage, carried out in a decentralized manner (Estrin *et al* 2000).

One objective of data-centricity is to let systems exploit the anticipated high node densities to achieve longer unsupervised operating lifetimes. Indeed, smaller form factor wireless sensor nodes have limited resources and often cannot afford to transfer all the collected data to the network edge and forward to centralized information processing systems. A practical example of the data-centric approach for network routing is directed diffusion (Intanagonwiwat *et al* 2000). This mechanism employs in-network processing by routing data along aggregation paths and thus removing the need for an address-centric architecture. It exploits data naming as the lowest level of system

organization and supports flexible and efficient in-network processing.

In summary, databases have a dual role to play in Pervasive and Ubiquitous Computing: in the short-term they need to provide the mapping between physical and virtual entities and space in a highly distributed and heterogeneous environment. In the longer term, database management systems need to provide the infrastructure for the development of data-centric systems. Each of the two phases is discussed in turn in the following sections.

DATABASES IN PERVASIVE AND UBIQUITOUS COMPUTING

As noted earlier, a key requirement of Pervasive and Ubiquitous Computing is the use of contextual information to adapt system behavior on-the-fly and deliver services that fit the specific user situation, a feature that has become known as context-awareness (Abowd and Mynatt 2000). Context-aware systems need to combine three elements: details of the physical environment must be sourced from embedded sensors, user profiles must be retrieved from distributed repositories holding fragments of identity information and these data must be correlated to a domain-specific knowledge model that can be used to reason about the specific situation. In these circumstances databases must support two aspects of the virtual-physical integration: on the one hand, to provide mappings between physical artifacts and associated digital resources and on the other, to hold profile, historical and state data. This functionality is best illustrated in a case study for example, ubiquitous retail (Kourouthanassis and Roussos 2003).

Database Systems Challenges

The introduction of Pervasive and Ubiquitous Computing technologies in a retail environment can help develop a number of novel applications that can transform the shopping experience and at the same time create value for retailers. Some of the possibilities that appear attractive to consumers include the smart shopping cart, in-store navigational assistance, shelf displays and offers and promotions targeted to the individual. At the same time, retailers can use these applications to gather data to help optimize the supply chain and increase profit margins, while at the same time improving consumer satisfaction. A core ingredient for such ubiquitous retail systems is the augmentation of consumer products with information processing capabilities and the development of supporting infrastructures.

To this end, several new technologies have been recently introduced. The Electronic Product Code (EPC) is a globally unique identifier that is attached to a particular product item stored in a radio frequency identification (RFID) tag. Unlike barcodes that are used to identify product classes, the EPC distinguishes a particular product instance, for example a specific can of Cola rather the class of Cola cans, and may be used to trace additional information about it. This information is retrieved by querying the Object Naming Service (ONS), which matches the EPC to the production server that holds information about the item in Product Markup Language (PML), for example its ingredients and date of production (Mealling 2003). Using these technologies, a smart shopping cart will identify the products placed into it using the EPC, retrieve additional information about the product via ONS and PML and provide personalized feedback to the consumer. For example, product information can be checked against the user profile retrieved from the consumer's identity management service provider for specific medical counter indications, for example lactose intolerance.

In that sense RFID has become a central technology with many applications in Pervasive and Ubiquitous Computing. The success of RFID is partly based on its simplicity and low-cost factor.

Today RFID-related technologies are used for Supply Chain Automation, Tracking (of assets, people, luggage, livestock, and so forth.), Medical Applications, Manufacturing, Retail, Timing (for example in Sport competitions), Robotics (Rousos et al 2007), Transport (for example Transport for London in conjunction with a major bank introduced a debit/credit card with an RFID embedded chip. The card can be used for travelling on public transport and for paying for low-value goods or services using the "wave-and-pay™" system), etc.

These examples highlight some of the immediate challenges of Pervasive and Ubiquitous Computing for database technology. Firstly, database management must address the demands of a highly heterogeneous environment both from a technical and an organizational point of view. Indeed, data resources are highly geographically distributed and different service providers control access to specific segments of the data space. Moreover, data generation and query insertion may move rapidly from one location to another to the extent that almost every object in the system is transient. Finally, these requirements put novel demands on system architectures and we will discuss each of these issues in turn in the remainder of this section.

Pervasive and Ubiquitous Computing databases need to cater for the physical distribution of data production, data consumption and data storage over wide geographic regions. Indeed, geographic distribution affects both data sources and applications, which are mobile to a very high degree. In the Pervasive and Ubiquitous retail scenario introduced previously a product item for example a single can of Cola, is a data source that may travel thousands of miles from its production plant until consumed and discarded. In the same scenario, consumers would wish to access personalized information services irrespective of their location in the world.

In addition to geographic distribution, Pervasive and Ubiquitous Computing infrastructures are also logically distributed since they integrate multiple independently controlled components and subsystems. It is not rare that one subsystem may be in operation in some form for decades but would still have to interoperate with newly developed technologies. For example, the shopping cart application discussed earlier retrieves data from the user's identity management service provider, the ONS system, the manufacturer PML servers and the retailer databases and still has to perform without significant delays for the user while also providing up to date information. This data source heterogeneity demands that management and acceleration architectures must adapt to accommodate application requirements.

Further, the origin of a particular query may also have considerable impact on the organization and performance of a distributed query-processing architecture. To be sure, distribution of queries and the rate at which are initiated will affect system performance. For example, reading the EPC of a new item placed inside a smart shopping cart will generate a new query to the PML server at the rate at which the consumer adds new items in the cart. To improve performance, some systems may correlate data distribution with query distribution to control their overhead. Alternatively, systems may restrict the set of available queries and thus limit their impact on system throughput. One solution to this problem is the construction lower-cost overlay networks for processing queries. Such overlays currently referred to as Service-Oriented Architectures, allow for data sources and sinks to connect to the system via the closest access point.

Pervasive and Ubiquitous Computing devices will also typically communicate over unreliable networks. Thus, failure should be a feature of the system and should be taken into account in designing appropriate query-processing architectures. For example, all objects in the database may be persistent so that when a client reconnects to the network the system may replicate the object closer to the new location of the client and thus reduce

latency. This feature would also support correlation of application queries and the data they carry, as is the case of a client that initiates a specific query for data but also becomes a source for the data after the query is satisfied.

Finally, developing optimized architectures for Pervasive and Ubiquitous Computing systems requires a better understanding of typical workload patterns. Consider the case of the smart shopping cart, which generates data and queries for data. In a realistic scenario of a supermarket chain, the pattern of the generated workload is rather unique and would induce considerable stress on existing database system architectures primarily due to the high utilization of scarce resources. Because of the limited experience we have with realistic workloads it may be inappropriate to optimize systems architectures. At the same time, the stress that Pervasive and Ubiquitous Computing systems will put on public resources, for example the ONS, will demand that massively distributed infrastructures are developed at a scale orders of magnitude larger to that available today.

Sensor Databases

A particularly important development for databases in Pervasive and Ubiquitous Computing is the emergence of the so-called sensor databases. Recent advances in micro-electro-mechanical systems (MEMS) and wireless communication, coupled with significant improvements in electronics manufacturing processes have enabled the development of low-cost, low-power, multifunctional sensor and actuator nodes (Akyildiz *et al* 2002). Such nodes are small in size and can communicate untethered in relatively short distances and thus it has become practical for these devices to be embedded into an increasing range of physical artifacts. Moreover, sensor and actuator nodes can be linked together in dense, self-configurable and adaptively coordinated networks often called sensor and actuator networks. These systems are also referred to as Smart Dust, wireless networks of sensors and actuators, motes, embedded wireless networks or smart-its. The name Smart Dust seems to have particular appeal with the term "dust"' justified on the basis that many of these nodes are expected to be less than one cubic millimeter in volume (currently $2.5mm^3$ sensors are in production, with $0.15mm^3$ nodes in laboratory testing).

Because individual nodes are embedded in objects, sensor and actuator networks are tightly coupled to the physical world. In this sense, they represent a realization of ubiquitous computing's Holy Grail to provide a ``connection of things in the world with computation'' (Weiser 1991). Although the same goal can be achieved using alternative technologies, sensors and actuators offer a competitive candidate for the deployment of pervasive, self-organizing networks of artifacts. Indeed, their cost and performance characteristics combined with their infrastructure-free operation and their low or no dependence on supporting computational and communications fabric offers distinct advantages in deployment and operation.

Yet, such flexibility does not come without constraints. Sensor and actuator nodes are also highly resource constrained and often create massively concurrent, complex, networked, computational systems. Moreover, they are integrated into objects and systems that are likely to function for long periods of time unsupervised and used by individuals who are not technology experts. Finally, they assemble information processing and communication systems with significantly greater size and scale compared to those that exist today.

Sensor and actuator networks operate unsupervised with individual nodes making decisions independently in a decentralized manner. Applications frequently operate in one of either modes:

i. In event-driven applications, for example detection of forest fires, the system remains inactive until an event is generated in one

of the nodes. Then, the event propagates through the system and causes the activation of appropriate behavior. This is semantically similar to Event-Condition-Action (ECA) rules in conventional databases, although in many sensor network scenarios events are occurring very rarely.

ii. In demand-driven applications, for example inventory tracking, activity is initiated in response to external requests, usually in the form of queries.

Both cases include a data transmission step, which may be either proactive or reactive. That is, data transmission may be in response to a particular query inserted to the network by an external device or it may be triggered by a condition that is checked at a particular node subset inside the network itself. Furthermore, in both cases system behavior is defined in terms of high-level statements of logical interests over the entire network and they might be represented either in an appropriate query definition language (Gehrke and Madden 2004) or as event-condition-action (ECA) triplets (Zoumboulakis and Roussos 2004).

The query-based approach to sensor databases views a large collection of potentially heterogeneous nodes as a single logical database unit. Users can submit arbitrary queries to this logical database and it is the task of the query processor to formulate a plan to process the query in the best possible way. Connectivity is often intermittent and node capabilities may vary. These parameters have to be taken into account when answering a query: for example if a user submits a simple `SELECT` query that asks for the temperature samples from at least three nodes in a small geographic region, then it is more sensible to disseminate the query to the nodes that have the highest energy reserves available. In addition there may be a division of responsibilities among sensor nodes: a number of nodes in a region may be merely collecting data while the remaining nodes process the collected data.

Examples of selected sensor network query processors have been implemented in Tiny DB (Madden *et al* 2003), Cougar (Yao and Gerhke 2002), Abstract Regions with support for geographic routing (Welsh 2004), ICEDB with support for intermittently-connected query processing (Zhang et al 2007), Ken with support for model-based approximate querying.(Chu et al 2006), REED which is an improvement over Tiny DB with support for event-filtering, and so forth.

The model-based approach to query processing is particularly interesting since it reduces the workload for the sensor nodes in the network. The basic principle in model-based querying is that users' queries are answered with probabilistic guarantees; the probabilities for these guarantees are calculated based on data previously seen. A simple example of this is an environmental monitoring application recording temperature and relative humidity. These attributes do not tend to fluctuate rapidly. If a user's query requests temperature readings at 30 second intervals for the duration of 6 hours, then a model-based approach can use previously gathered data, build a probabilistic model using this data and answer the query using the model thus reducing the need for directly querying the sensor network (the network need only be queried when the model can not meet the requirements of the users' queries – for instance a user in her query might stipulate that she only desires readings that are accurate with 95 per cent confidence; if this confidence can not be met by the model then the sensor network is queried). Day/night patterns together with seasonal variations affect temperature. If previous data is taken into account (for example last year's data) then the model can provide answers with reasonable accuracy. There is an example of this approach (Desphande et al 2004) that uses a Bayesian framework to constantly update probabilities in light of new data. The previously mentioned system called Ken is an improvement of this notion.

A second database technology that provides an appropriate computational model for event-based sensor and actuator network applications is event-condition-action (ECA) rules (Zoumboulakis and Roussos 2004) and more generally event-based approaches. This approach suits scenarios where there are large periods of stability interrupted by short periods of activity – for example a pervasive health monitoring system such as Biosensornet (Biosensornet 2007) where a patient's health is monitored in his/her sensor equipped home. In this setting an event could be the patient not using the fridge for a day – captured by the sensor fitted on the refrigerator door – or not taking his/her medications at a given time. This type of event can trigger a notification to be sent to a social worker who may decide to call in order to check the patient's status.

Many event-based systems are built on the publish/subscribe paradigm where an interested user (referred to as subscriber or consumer) opts to receive notifications for a specific type of an event occurring at some source (referred to as publisher or producer). Specifically for sensor networks the delivery of notifications may be asynchronous due to connectivity constraints – for example there might not be a path between the producer and the consumer at the time of the event. An interesting approach to this problem is proposed by the previously mentioned REED model (and a similar approach is also described by Thome 2005) – events are generated and their data is stored at an event staging area; users' subscriptions are stored in a table and the event storage area is queried using join semantics.

Both query-based and event-based technologies viewed as a computational paradigm for sensor and actuator networks are successful in freeing developers from lower level concerns by transferring responsibility of data management, collection and processing to the system (Bonnet *et al* 2001). Despite their early successes, there are still several challenges associated with the design and development of such sensor databases:

- *Node resource management.* Sensor and actuator nodes are resource-constrained and any system should carefully manage their consumption, especially power. Communication and sensing tend to dominate demands on the battery while computation places a relative lower burden.
- *Network topology management.* Sensor and actuator networks have a fundamentally transient nature due to moving nodes, signal degradation due to changing interference patterns and nodes running out of power.
- *Data aggregation.* Due to the high cost of communication and the possible very high data collection rates it is desirable to summarize or abstract the recorded data as they propagate through the network. For example, decomposable operators may be computed on the fly by intermediate nodes and only the aggregates forwarded.
- *Simple and expressive language.* To preserve node resources the query or the ECA language used to interact with the sensor network should be simple enough to support a lightweight implementation and expressive enough to be useful in building applications.
- *Management tools.* In addition to an appropriate conceptual framework a system becomes useful when it also offers a collection of tools to assist users in management and interaction.

One implication of these requirements is that it is often necessary for the nodes to perform collaborative signal processing to effectively analyze the sensed data. Furthermore, the high probability of node failures implies that sensor data will be noisy and thus managing sensor data uncertainty should be adequately addressed. Another critical concern is the designing of efficient routing mechanisms that cater for the particular requirements of query processing and reactive functionality. As a result it is often necessary to

develop database systems that bypass the traditionally distinct network layers.

A further implication is that query dissemination or correspondingly construction of an event channel for ECA rule execution often requires two phases. First, the preprocessing of a routing tree that defines the pathways along which data travels. This routing tree is constructed online as nodes forward notifications to their selected neighbors in the network. When signals are generated at sensor nodes the recorded data is possibly aggregated with information received by other sensors and subsequently forward to parent nodes until data reaches its intended destination. The details of such routing algorithms are beyond the scope of this article but the interested reader may consult Gherke and Madden (2004) for full details.

FUTURE TRENDS

In the next decade databases employed in support of Pervasive and Ubiquitous Computing systems will be extended to cater for the geographically decentralized operation both in terms of data generation and application access, the transient nature of pervasive computing objects and entities and the fundamentally heterogeneous nature of such system and the need for correlation and coordination of physical and virtual resources. New systems architectures will be developed to support Pervasive and Ubiquitous Computing systems and the novel workloads that they bring with them.

A particularly interesting Pervasive and Ubiquitous Computing system that is expected to attract considerable interest over the next years is sensor databases. Sensor databases consist of numerous tiny embedded computing devices capable of wireless communication. Such systems have extremely limited resources and must take advantage of in-network processing and storage to survive for longer periods of time. Database management systems for sensor network offer distinct advantages form a user interaction perspective but also face considerable challenges for efficient system operation.

CONCLUSION

Pervasive and Ubiquitous Computing offers the promise of pervasive information and communications infrastructures that provide their services whenever required while at the same time taking into account the particular context of the user. Pervasive and Ubiquitous Computing also blurs the boundaries between the physical and the virtual worlds through sensing and actuation technologies. To realize this vision, developments in several technical areas are required and database management systems have a critical role to play in supporting Pervasive and Ubiquitous computing services.

REFERENCES

Abadi, D. et al. (2005). REED: Robust, Efficient Filtering and Event Detection in Sensor Networks. *In Proceedings of VLDB 2005,* Trodheim, Norway.

Abowd, G., & Mynatt, E. (2000). Charting past, present, and future research in ubiquitous computing. *ACM Transactions on Computer-Human Interaction, 7*(1), 29-58.

Akyildiz, I. F., Su, W., Sankarasubramaniam, Y., & Cayirci, E. (2002). Wireless Sensor Networks: A Survey. *Computer Networks, 38,* 393-422.

Biosensornet: *Autonomic Biosensor Networks for Pervasive Healthcare,*http://www.doc.ic.ac.uk/ mss/Biosensornet.htm 2007

Bonnet, P., Gehrke, J. E., & Seshadri, P. (2001). Towards Sensor Database Systems. *Proceedings of the Second International Conference on Mobile Data Management*, Hong Kong, January 2001.

Chu D. et al. (2006). *Approximate Data Collection in Sensor Networks using Probabilistic Models,* ICDE.

Davies, N., & Gellersen, H. (2002) Beyond prototypes: Challenges in Deploying Ubiquitous Systems. *IEEE Pervasive Computing, 1*(1), 29-35.

Deshpande A. et al. (2004). *Model-Driven Data Acquisition in Sensor Networks, VLDB.*

Estrin, D., Culler, D., Pister, K., & Sukhatme, G. (2002). Connecting the Physical World with Pervasive Networks. *IEEE Pervasive Computing, 1*(1), 59-69.

Estrin, D., Govindan, R., Heidemann, J., & Kumar, S. (1999). Next Century Challenges: Scalable Coordination in Sensor Networks. *Proceedings of the ACM/IEEE International Conference on Mobile Computing and Networking. Seattle, Washington*, USA, (pp. 263-270).

Gehrke, J., & Madden, S. (2004). Query Processing in Sensor Networks. *IEEE Pervasive Computing, 3*(1), 46-55.

Intanagonwiwat, C., Govindan, R., & Estrin, D. (2000). Directed Diffusion: A Scalable and Robust Communication Paradigm for Sensor Networks. *Proceedings of the Sixth Annual International Conference on Mobile Computing and Networking* (MobiCOM '00), August 2000, Boston, MA.

Kourouthanassis, P., & Roussos, G. (2003). Developing Consumer-Friendly Pervasive Retail Systems. *IEEE Pervasive Computing, 2*(2), 32-39.

Krishnamachari, B., Estrin, D., & Walker, S. (2002). The Impact of Data Aggregation in Wireless Sensor Networks. *International Workshop on Distributed Event-Based Systems*, Vienna, Austria, July.

Madden, S., Franklin, M. J, Hellerstein, J. M., & Hong, W. (2003). The Design of an Acquisitional Query Processor for Sensor Networks. *Proceedings of the 2003 ACM SIGMOD international conference on Management of data*, San Diego, CA, (pp. 491-502).

National Research Council (2001). Embedded, Everywhere: A Research Agenda for Networked Systems of Embedded Computers. *National Academy Press, Washington DC, USA.*

Mealling, M. (2003). Object Name Service (ONS) 1.0. *Auto-ID Center, Boston MA, USA.*

Roussos G., Papadogkonas D., Taylor J., Airantzis D., Levene M., & Zoumboulakis M. (2007). Shared Memories: A Trail-based Coordination Server for Robot Teams. *First International Conference on Robot Communication and Coordination (IEEE Robocomm),* 15-17 October, Athens, Greece.

Sadagopan, N., Krishnamachari, A., & Helmy, A. (2004). *Active Query Forwarding in Sensor Networks.* Ad-Hoc Networks, in press.

Satyanarayanan, M. (2001). Pervasive Computing: Vision and Challenges. *IEEE Personal Communications, 8*(4), 10-17.

Thome B., Gawlick D., & Pratt M. (2005). Event Processing with an Oracle Database. *In Proceedings of SIGMOD* (pp.863-867), Baltimore, Maryland.

Weiser, M. (1991). The computer of the 21st century. *Scientific American, 265*(3), 66–75.

Welsh, M. (2004). Exposing Resource Tradeoffs in Region-Based Communication Abstractions for Sensor Networks. In SIGCOMM, 34(1).

Yao, Y., & Gehrke, J. E. (2002). The Cougar Approach to In-Network Query Processing in Sensor Networks. *Sigmod Record, 31*(3).

Zhang Y. et al. (2007). ICEDB: Intermittently-Connected Continuous Query Processing. *In Proceedings of ICDE.*

Zoumboulakis, M., Roussos, G., & Poulovassilis, A. (2004). Asene: Active Sensor Networks. *In-*

ternational Workshop on Data Management for Sensor Networks VLDB 2004, Toronto, Canada, 30 August.

KEY TERMS

Data Aggregation: Data aggregation is the process in which information is gathered and expressed in a summary form. In case the aggregation operator is decomposable partial aggregation schemes may be employed in which intermediate results are produced that contain sufficient information to compute the final results. If the aggregation operator is non-decomposable then partial aggregation schemes can still be useful to provide approximate summaries.

In-Network Processing: In-network processing is a technique employed in sensor database systems whereby the data recorded is processed by the sensor nodes themselves. This is in contrast to the standard approach, which demands that data is routed to a so-called sink computer located outside the sensor network for processing. In-network processing is critical for sensor nodes because they are highly resource constrained, in particular in terms of battery power and this approach can extend their useful life quite considerably.

Pervasive and Ubiquitous Computing: Pervasive and Ubiquitous computing is a vision of the future and a collection of technologies where information technology becomes pervasive, embedded in the everyday environments and thus invisible to the users. According to this vision, everyday environments will be saturated by computation and wireless communication capacity and yet they would be gracefully integrated with human users.

Sensor and Actuator Networks: An ad-hoc, mobile, wireless network consisting of nodes of very small size either embedded in objects or freestanding. Some nodes have sensing capabilities, which frequently include environmental conditions, existence of specific chemicals, motion characteristics and location. Other nodes have actuation capabilities, for example they may provide local control of mechanical, electric or biological actuators and optical long-range communications.

Sensor Databases: The total of stored sensor data in a sensor and actuator networked is called a sensor database. The data contains both metadata about the nodes themselves, for example a list of nodes and their related attributes, such as their location, and sensed data. The types and the quantity of sensed data depend on the particular types of sensors that participate in the network.

Sensor Query Processing: Sensor query processing is the design of algorithms and their implementations used to run queries over sensor databases. Due to the limited resources of sensor and actuator nodes query processing must employ in-network processing and storage mechanisms

Service-Oriented Architectures: A service is an information technology function that is well defined, self-contained, and does not depend on the context or state of other services. A service-oriented architecture is an approach to building information technology systems as a collection of services which communicate with each other. The communication may involve either simple data passing between services or it could involve infrastructure services which coordinate service interaction. SOA is seen as a core component of Ubiquitous Computing infrastructures.

Chapter LXXXVII
Business-to-Business (B2B) Integration

Christoph Bussler
Merced Systems, Inc.

BUSINESS-TO-BUSINESS (B2B) INTEGRATION TECHNOLOGY

Businesses world-wide started exchanging electronic business messages with each other around 1970. This coincides with the emergence of wide-area computer networks. Immediately businesses realized the potential in sending electronic messages instead of paper letters in order to conduct business electronically with each other. Computer networks provided a significant increase of transmission speed, less failures due to message loss and direct processing of messages upon receipt without error prone manual transcripts from paper to computer terminals or vice versa. Overall, business interactions became a lot more reliable and efficient.

The direct processing capability of electronic messages enabled the seamless integration of the messaging environment and the back end application systems. The electronic integration of businesses was achieved this way and the technological basis was called Business-to-Business (B2B) integration technology. However, B2B integration technology was not affordable to every business independent of its size due to its high

deployment and maintenance cost. Therefore, only large businesses with an information technology (IT) department could afford electronic B2B integration. Smaller businesses continued to rely on paper letters or fax transmissions; later on email was used, too, in order to send and receive business data.

Today it is commonly accepted to exchange electronic messages between businesses as the complexity of the technology and the cost are both reducing significantly. Current trends are to make B2B technology even more accessible to every business independent of its size through ubiquitous Web technology in combination with the standardized markup language XML. This new technology is called Web Services and all software vendors readily provide solutions (Alonso et al. 2004), (Bussler 2003).

HISTORICAL BACKGROUND

Value Added Networks (VANs) started providing the transport of electronic messages between businesses around 1970. VANs are dedicated networks and since their service is very reliable they still are extremely popular today and widely used. VANs do not support direct communication between businesses. Instead, businesses have to upload and download messages asynchronously from VANs. VANs provide a persistent mailbox system where every partner has a dedicated mailbox storing the electronic messages addressed to it. Based on their behavior VANs are an asynchronous network for exchange of messages. In addition ('value-added') they provide services like storage, backup, and billing. This technical implementation allows businesses to send and receive the messages independent of each other's communication availability. This is very important as the availability of the business partner is not necessary in order to communicate business data.

Through the emergence of VANs Business-to-Business (B2B) integration was born (Bussler 2003). VANs made a big difference in competitiveness since businesses could rely on the extreme high speed of electronic data transmission compared to paper-based communication through postal services. Different industries embarked on B2B integration through at that time new form of communication.

It became immediately apparent that it is not advantageous at all for businesses to define their own message formats and send them over VANs. This would require a business to deal with all the different message formats defined by all its business partners if every of it would provide its own message format. A by far and significantly better approach was to standardize the message formats across businesses so that a business could use the same message formats across all its business partners from the same industry. In specific industries message definition standards have been developed for over 30 years now. An example for the supply-chain industry is EDI (ASC 2007), one for the banking industry is SWIFT (SWIFT 2007) and one for the insurance industry is (ACORD 2007). These standards are called B2B standards and enable the fast integration of a business with other businesses in the same industry. A business can comply with the standard and by that it is ensured that the message exchange with the other partners is interoperable. The standardized message formats ensure interoperability. Of course, one business might be involved in several industries and in this case the business has to support several standards. However, in general it is limited to one standard per industry.

While the message formats have been standardized, the technical software implementations for B2B communication have not. Every VAN is providing its own communication software for uploading and downloading the messages from the mailboxes. In order to allow messages to be automatically processed by back end application

systems, businesses had to integrate their VANs' communication software into their information system environment. This back end application system connectivity is necessary for retrieving and storing the data from the back end application systems that is communicated through the communication software. This meant that each business had to do some custom integration implementation in order to make the B2B communication with its partners work.

In the late 1980s and early 1990s first B2B integration products appeared on the software market as standalone software products. These products were off-the-shelf software that did not require any custom coding by the businesses any more in order to connect to the VAN's software or the back end application system software. Instead, it could automatically link the VANs' communication software and the back end application systems (like for example enterprise resource planning (ERP) systems) within the businesses. Establishing B2B integration became more and more like a turn-key software solution instead of a custom coding task.

INTEGRATION TECHNOLOGY

The precondition for an off-the-shelf B2B integration product is a conceptualization of the underlying functionality so that a general implementation is possible [Bussler 2003]. The main concepts are:

- B2B protocols
- Back end application adapters
- Data transformation and mediation
- Process management
- Process mediation
- Integration partner management

Each of these concepts will be discussed next in more detail in order to show how the conceptualization led to generic implementations of the functionality allowing off-the-shelf B2B integration products.

B2B protocols are communication protocols that define message standards as well as the exchange sequence of messages. A B2B protocol is a formal definition that allows interoperability across networks on a business data content level. With standardization it is possible to design and implement generic protocol engines that can execute different B2B protocols and therefore a generic implementation is possible. There is a multitude of existing B2B protocol standards today and the important ones from a historical perspective have been introduced above. Currently more are developed, see the section 'Current Developments' in the following.

The second important area is the connectivity to back end application systems. While the number of back end application systems is huge, there are only a few patterns to interact and interface with back end application systems. The main patterns are direct invocation, message-based invocation, interface table based invocation and screen scraping. Once these patterns crystallized out, standard development started and based on these standards it was possible to build a generic connector framework for B2B integration technology. This automatic connectivity to back end application systems is achieved through specialized software adapters that adapt the B2B integration product's interface to the particular interface of the back end application system (Apte 2002). An adapter enables the B2B integration technology to insert as well as to extract data from the back end application system. The benefit for the businesses was that they did not have to perform any custom integration implementation any more, but could use prepackaged B2B integration software instead. That allowed buying the integration functionality as software products without going through software development processes in the internal IT organizations. A standard called J2EE Connector Architecture (JCA 2007) allows software vendors to build standardized adapters.

This contributes to the turn-key nature of integration products.

The third important conceptualization that occurred was the explicit notion of data mediation or transformation (synonyms) (Bussler 2003) (Omelayenko and Fensel 2001) (Rahm and Bernstein 2001). As the same business data is represented in general differently by B2B protocols and the interfaces of back end application systems, it is necessary to transform the business content between the different formats. This is not only syntactical transformation, but also semantic transformation like replacing values, looking up values or even recalculating values like in the case of money transactions. An example of a data definition mismatch is that an address can be defined as a single string or a structured record where each address element like city or zip code is stored in a separate field. Not only the structure of the data might have to be changed (e.g. from a one string representation into a structured record), but also the content representation (e.g., 'Ireland' in one representation has to be replaced by 'IRL' in another one). Once B2B standards appeared as well as transformation languages like XSLT it was possible to build generic transformation engines. The data mapping has to happen in such a way that the meaning of the data (data semantics) is not changed at all, i.e., the data semantics has to be absolutely preserved. Otherwise the data might be misinterpreted in the back end application systems leading to system or business logic errors. Data semantics preserving transformation is a major research topic currently and the field of semantics is addressing this problem, see the section 'Future Trends' below.

Another important conceptualization took place for process management. B2B protocols define the messages as well as the particular exchange sequence of these messages. This ensures that two or more communicating organizations are interoperable. However, once messages are received, they require further processing. In the simplest case this is sending the message after data transformation directly to one back end application system. In many cases, though, more processing steps have to take place than just inserting a message into a back end application system. Examples are authorizations, validations, or even replicating the data into several back end systems. With the emergence of process management as a separate functionality (in analogy to databases) it was possible to provide a generic process engine in context of B2B integration technology.

Process mediation is an area that recently started to become a separate area of conceptualization. Not only data structures and vocabularies can mismatch, but also the message exchange sequences. For example, while a back end application system might send one message and expects a return message back, a B2B interaction might have more messages including acknowledgement messages in order to establish an exactly once transmission. These differences in message exchanges are overcome by process mediation (Fensel and Bussler 2002). Process mediation ensures that all communicating parties receive the messages in the sequence they require. The area of process mediation is another important area that is currently researched and works described in 'Future Trends' are currently researching into this functionality.

Integration partner management is an essential aspect of B2B integration. When businesses send and receive messages they have to make sure that they know their communication partners. Otherwise they might accept messages that are not sent by a contractually bound partner. In this case messages might be processed that should not. On the other hand side, message must only be sent to partners, not to any other organization. In order to define partners as well as their specific properties standards like Collaboration Partner Profile (CPP) and Collaboration Partner Agreements (CPA) are established (ebXML 2007). These standards allow a generic implementation of integration partner management that is basis for off-the-shelf product development.

These six areas are the most important concepts that have to be implemented by B2B integration technology as a generic set of components together establishing generic B2B integration products. The availability of software products for B2B integration lowered their price significantly, mainly, because custom coding was not necessary any more. This enabled smaller businesses to participate in B2B integration and soon more businesses made use of the B2B integration technology than ever before.

CURRENT DEVELOPMENTS

Once the conceptualization took place B2B integration products were built. However, at the time, the implementation itself was proprietary in the sense that no standard technology was available for implementing the B2B integration products themselves. This changed significantly with recent developments of Web Service technology and XML in context of the Internet as a communication platform. In addition, based on the acceptance of XML as a standard, new B2B protocols are developed. The current activities in these areas are introduced next.

In the mid 1990s XML (XML 2007) emerged as a viable language technology and was picked up by software developers around the world. At the same time first B2B integration products based on XML as the message syntax were built and appeared on the market. The promise was that the new technology is going to make the B2B integration task a lot easier and a lot cheaper. Using XML as a different technology basis ensured that interoperability is easier to achieve, however, the various component like data transformation or back end application adapters had to be moved to XML first before B2B integration technology as a whole could benefit from it.

The Internet is available as communication platform and many B2B integration products started using the Internet instead of VANs. That proved difficult since the open and unreliable Internet required the development of secure and reliable communication protocols. This is in contrast to VANs that are proprietary networks and their access is restricted to the VANs' customers; the access to the VANs is controlled as compared to the Internet. In addition, in case of the Internet there was no asynchronous behavior provided by the network any more. Instead, businesses have to communicate directly with each other and the communication partners have to be both available in order for the communication to be successful. This requires sufficient uptime to avoid communication errors due to unavailability. From a B2B integration technology point of view more communication protocols are available now for organizations and so it is easier to achieve B2B connectivity as organization can choose the best possible connectivity for them.

Recently new B2B integration standards appeared like RosettaNet (RosettaNet 2007), cXML (cXML 2007) and ebXML (ebXML 2007). These use XML as the syntax for defining message types. However, XML did not deliver on the promise that standards became better or easier to use. The promise was delivered on the assumption that one commonly accepted message syntax makes a significant difference. Instead, the contrary happened and quite unexpectedly. Because it was so easy to define message types in XML, too many were created quickly contradicting the purpose of standards as such. Instead of less variety of message types, a huge number appeared in each business domain. The flurry of proposed XML message standards made potential customers wonder about how these are going to be managed in the long run. Consequently, customers held back in investing into XML-based B2B technology and continued to use VANs instead.

XML and the formats used earlier for B2B standards do not capture the semantics of the data, just the structure (syntax) of data. Because of this situation it is not possible for the B2B integration technology to semi-automatically or fully

automatically perform the mediation. Instead, a developer has to develop the rules that define the mediation by hand. While this is a possible solution, manual encoding of the rules lead often to severe mistakes in the B2B communication. The work introduced in the next section addresses the semantic interoperability problem.

Independent of its usefulness, once XML was 'discovered' it became clear that XML messages can be sent through HTTP connections. The sending of XML message through Internet technology is called Web Services. Prominent standards in the space are SOAP (SOAP 2007) and WSDL (WSDL 2007). Both together allow defining basic and single message exchanges between a client and a server. Soon it was discovered that single message exchanges are insufficient for serious communication patterns between businesses. Several so-called process standards try to address this like (BPEL 2007) (ebXML BP 2007) (WSCDL 2007) as well as many research groups.

In summary, current developments provide additional technologies for B2B integration and these technologies are being picked up right now for B2B integration. At the same time, semantics is not directly addressed by the new technologies and current work in the area of semantics is addressing these deficiencies. The next section will focus on this area and show current activities.

FUTURE TRENDS

The advent of Semantic Web technology promises to improve the overall situation significantly, especially providing solutions for the mediation problem. Ontology languages like RDF (RDF 2007), RDF/S (RDFS 2007) and OWL (OWL 2007) allow capturing much more semantics about data than XML or other non-semantic languages. Message formats can be defined with ontology languages not only describing the syntax but also the semantics. Once the semantics is formally captured, the mediation problem can be addressed a lot better. This is possible since the semantics is formally denoted and the integration technology can infer from this if two data definitions match or not. If they do, transformation between the two can be automatically achieved. If not, a human can be brought into the loop helping to overcome the semantic mismatch. However, solving the transformation problem based on ontologies is a very recent development and no products can be bought yet that have this higher-level functionality.

A significant effort applying Semantic Web technology in order to solve the integration problems is the Web Service Modeling Ontology (WSMO) (WSMO 2007). This ontology, which is developed by a significant number of organizations, uses Semantic Web Technology to define a conceptual model to solve the integration problem that encompasses B2B integration. The main conceptual elements proposed are ontologies for defining data and messages, mediators for defining transformation, goals for dynamically binding providers and web services for defining the interfaces of communicating services. In order to define a specific B2B integration, the formal Web Service Modeling Language (WSML) (WSML 2007) is defined. This can be processed by a WSML parser. In order to execute B2B integration the Web Service Modeling Ontology Execution environment (WSMX) (WSMX 2007) is developed. It serves as a runtime environment for integrations defined through WSML.

However, in 2007, the future is not here yet. While a lot of progress has been made in the Semantic Web space, the progress is not yet at a point where B2B integration becomes easier to establish. To cater for the communication aspect, the field of Semantic Web Services (SWS) emerged with the major efforts being OWL-S (OWL-S 2007) and WSMO (WSMO 2007). A W3C standard called SAWSDL was recently released as a recommendation (SAWSDL 2007) that proposed how initial SWS elements can find their way into the main stream Web Service technology domain. However, at this point it is too early to report

on a significant uptake of this new technology by established businesses for production use. In (Shadbolt et al. 2006) and (Lassila and Hendler, 2007) the authors give a good impression of the state of the progress made in this space.

CRITICAL ISSUES OF B2B INTEGRATION TECHNOLOGIES

Despite the wide-spread use of B2B integration in industry and government organizations, B2B integration technology is still an area of major product development efforts. Software vendors like BEA (BEA 2007), IBM (IBM 2007), Microsoft (Microsoft 2007) and Oracle (Oracle 2007) continue to develop new products providing B2B integration functionality.

In recent years the scientific community picked up the topic of integration in context of Web Services (Web Services 2007) and the Semantic Web activities (Fensel and Bussler 2002), see also future trends above.

Independent of the specific approaches in industry and academia, critical issues as listed in Table 1 have to be addressed for the technology to work and for the research to make significant contributions. These issues are not solved yet despite many research project and software products continue to be worked on.

CONCLUSION

Business-to-Business (B2B) Integration is absolutely essential for businesses and organizations to not only stay competitive but also keep or even gain market share. Furthermore, the trend is going towards connecting all enterprises electronically for the benefit of the enterprises as well as customers. Once all business are connected all business interactions can be implemented as business messages making the interactions as efficient as possible.

Ultimately the Semantic Web technology will make the task of integrating businesses easy due

Table 1. A summary of critical issues

Choreography Message exchange pattern that a business exposes and that a business partner has to follow or agree upon for successful exchanging business messages	**Mediation** Semantics-preserving transformation of a message in a message format into a separate message following a different message format
Communication software Software that implements the communication of messages between businesses based on a specific network protocol like TCP/IP	**Message security** Insurance that message cannot be read by unauthorized businesses, that message are not replayed, deleted or modified while in transmission
Conversation Series of message exchanges between two or more businesses in order to achieve a business goal	**Non-repudiation** Security functionality that ensures that the sender and the receiver cannot claim to not have received or sent a specific business message
Error management Handling of error messages that indicate a message exchange error by executing subsequent actions to rectify the error situation into a consistent business state	**Orchestration** Sequenced invocation of several business partners by a given business partner for it to achieve its business goals
Exactly-once message transmission Assurance that each sent message is received either once or not at all by a business partner in presence of a non-reliable and non-transactional network protocol	**Partner identification** Unique identification of business partners including the assurance that they do not assume a wrong identity

to the semantic description of the message formats as well as choreographies and orchestration. Once semantically described the goal of self-composing enterprises becomes feasible.

REFERENCES

ACORD (2007). ACORD Corporation. www.acord.org.

Alonso, G., Casati, F., Kuno, H., & Machiraju, V. (2004). *Web Services - Concepts, Architectures and Applications.* Springer-Verlag.

Apte, A. (2002). *Java Connector Architecture: Building Enterprise Adaptors.* SAMS Publishing.

ASC (2007). The Accredited Standards Committee (ASC) X12. www.x12.org

BEA (2007). BEA Systems, Inc. www.bea.com.

BPEL (2007). OASIS Web Services Business Process Execution Language. www.oasis-open.org/committees/tc_home.php?wg_abbrev=wsbpel

Bussler, C. (2003). *B2B Integration*. Springer-Verlag.

cXML (2007). Commerce XML. www.cxml.org/

ebXML (2007). Electronic Business using eXtensible Markup Language (ebXML). www.ebxml.org

ebXML BP (2007). OASIS ebXML Business Process. www.oasis-open.org/committees/tc_home.php?wg_abbrev=ebxml-bp

Fensel, D., Bussler, C. (2002). The Web Service Modeling Framework WSMF. *Electronic Commerce Research and Applications*, Vol 1, No. 2. Elsevier Science

IBM (2007). International Business Machines Corporation. www.ibm.com

JCA (2007). J2EE Connector Architecture. java.sun.com/j2ee/connector/index.jsp

Lassila, O., Hendler, J. (2007). Embracing Web 3.0. IEEE Internet Computing, May/June 2007

Microsoft (2007). Microsoft Corporation. www.microsoft.com

Omelayenko, B., & Fensel, D. (2001). An Analysis of Integration Problems of XML-Based Catalogues for B2B E-commerce. In *Proceedings of the 9th IFIP 2.6 Working Conference on Database (DS-9), Semantic Issues in e-commerce Systems.* Hong Kong, China, 2001

Oracle (2007). Oracle Corporation. www.oracle.com

OWL (2007). Web Ontology Language. www.w3.org/2001/sw/WebOnt/

OWL-S (2007) OWL-S. www.daml.org/services/owl-s/

Rahm, E., Bernstein, P. (2001). A Survey of Approaches to Automatic Schema Matching. *The VLDB Journal.*

RDF (2007). Resource Description Framework. www.w3.org/RDF/

RDFS (2007). Resource Description Framework Schema. www.w3.org/TR/rdf-schema/

RosettaNet (2007). Rosettanet. www.rosettanet.org

SAWSDL (2007). www.w3.org/TR/sawsdl/

Shadbolt, N., Hall, W., Berners-Lee, T. (2006) The Semantic Web Revisited. *IEEE Intelligent Systems*, Vol. 21, Issue 3, May/June 2006

SOAP (2007). Simple Object Access Protocol (SOAP) 1.2. www.w3.org/TR/soap12-part1/ and www.w3.org/TR/soap12-part2/

SWIFT (2007). Society for Worldwide Interbank Financial Telecommunication (SWIFT). www.swift.com

Web Services (2007). Web Services Activity. www.w3.org/2002/ws/

WSCDL (2007). Web Services Choreography. www.w3.org/2002/ws/chor/

WSDL (2007). Web Service Description Language (WSDL) 1.1. www.w3.org/TR/wsdl

WSML (2007). Web Service Modeling Language. www.wsmo.org/wsml

WSMO (2007). Web Service Modeling Ontology. www.wsmo.org

WSMX (2007). Web Service Modeling Execution Environment. www.wsmx.org

XML (2007). Extensible Markup Language. www.w3c.org/XML/

KEY TERMS

B2B Integration Technology: A software system that provides business-to-business integration functionality by sending and receiving messages and retrieving and storing them in back end application systems.

Back End Application System: A software system that manages business domain specific data for businesses like Enterprise Resource Planning (ERP) systems.

Business Partner: A business partner is a company or an organization with which to engage in a business relationship like a buyer, a seller or a shipper.

Choreography: Choreography is the message exchange behavior that a business exposes in order to participate in a business relationship based on electronic message exchanges.

Mediation: Mediation is the semantics-preserving transformation of a message in one message format into a message of a different message format that represents the same information.

Message Standard: A message standard defines the attributes as well as possible values that a business has to use in order to support interoperable electronic message exchange.

Orchestration: A business uses orchestration in order to define the electronic message interaction with other business partners in order to fulfill its obligations.

Value Added Network: A value added network (VAN) is a company that provides network capabilities to business partners that want to exchange electronic messages. A VAN provides also storage, back-up, security and other services in order to provide more message exchange functionality.

Chapter LXXXVIII
Enterprise Application Integration (EAI)

Christoph Bussler
Merced Systems, Inc.

ENTERPRISE APPLICATION INTEGRATION (EAI) TECHNOLOGY

As long as businesses only have one enterprise application or back end application system there is no need to share data with any other system in the company. All data that has to be managed is contained within one back end application system and its database. However, as businesses grow, more back end application systems find their way into their information technology infrastructure managing different specialized business data, mainly introduced due to the growth. These back end application systems are not independent of each other; in general they contain similar or overlapping business data or are part of business processes. Keeping the data in the various application systems consistent with each other requires their integration so that data can be exchanged or synchronized. The technology that supports the integration of various application systems and their databases is called Enterprise Application Integration (EAI) technology. EAI technology is able to connect to back end application systems in order to retrieve and to insert data. Once connected, EAI technology supports the definition

of how extracted data is propagated to back end application systems solving the general integration problem.

BACKGROUND

Typical examples of back end application systems that are deployed as part of a company's information technology (IT) infrastructure are an Enterprise Resource Planning (ERP) system or a Manufacturing Resource Planning (MRP) system. In the general case, different back end application systems store potentially different data about the same objects like customers or machine parts. For example, a part might be described in an ERP system as well as in a MRP system. The reason for the part being described in two different back end application systems is that different aspects of the same part are described and managed. In fact, this means that the not necessarily equal representation of the object exists twice, once in every system. If there are more than two systems, then it might be very well the case that the same object is represented several times. Any changes to the object have to be applied to the representation of the object in all systems that contain the object. And, since this distributed update cannot happen simultaneously (in the general case), during the period of applying the change the same object will be represented differently until the changes have been applied to all representations in all back end application systems. It therefore can very well be the case that during an address update of a customer object the customer has two addresses. Some objects representing the customer have already the new address while others still have the old address. This situation exists until the distributed update is complete. Furthermore, in most cases there is no record of how many systems represent the same object. It might be the case and actually often it is the case that a change is not applied to all objects because it is not known which back end application system has a representation of the object in the first place. Only over time these cases will be detected and rectified, mainly through the resolution of error situations.

In summary, the same object can be represented in different back end application systems, the updates to an object can cause delays and inconsistencies, and locations of object representations can be unknown due to missing object registries.

A second use case is that precisely the same object is replicated in different back end application systems. In this case the update of the object in one system has to be applied to all the other systems that store the same object. The objects are replica of each other since all have to be updated in the same way so their content is exactly the same. Only when all the objects are updated they are consistent again and the overall status across the back end application systems is consistent again. In the replicated case it must not be possible that the same object exposes different properties like in the address example above.

A third use case is that applications participate in common business processes. For example, first a part is being purchased through the ERP system and upon delivery it is entered and marked as available in the MRP system. The business process behind this is consisting of several steps, namely purchase a part, receive the part, make the part available, and so on. In this case the back end application systems do not share common data, but their data state depends on the progress of a business process and it has to update the back end application systems accordingly. The data will change their state according to the progress of the business process. In this sense they share a common business process, each managing the data involved in it.

All these three use cases, while looking quite different from each other, have to be implemented by companies in order to keep their business data consistent. EAI technology (Bussler 2003)

(Hohpe and Woolf 2003) allows to accomplish this at it provides the necessary functionality as described next.

ENTERPRISE APPLICATION INTEGRATION TECHNOLOGY

Enterprise application integration technology addresses the various scenarios that have been introduced above by providing the required functionality. In the following the different functionalities will be introduced step by step. First, EAI technology provides a reliable communication mechanism so that any communication of data between back end application systems is reliable. This ensures that no communication is lost. This is often achieved by sending messages through persistent and transactional queuing systems (Gray and Reuter 1993). Queuing systems store messages persistently and transactionally achieving an exactly-once message communication.

EAI technology uses adapters (J2EE Connector Architecture 2007) in order to connect to the proprietary interfaces of back end application systems. An adapter knows the proprietary interface and provides a structured interface to the EAI technology. This allows EAI technology to connect to any back end application system for which an adapter is available. If no adapter is available for a particular back end application system then a new one has to be built. Some EAI technologies provide an adapter development environment allowing new adapters to be constructed. Adapters can retrieve data from back end application systems as well as store data within back end application systems. They therefore enable the communication of the EAI technology with the back end application systems.

EAI technology by means of connecting to all required back end application systems with adapters can propagate data changes to all of the back end application systems. For example, this allows ensuring that all representations of an object in several back end application systems can be changed as required. EAI technology is required to process changes in an order preserving fashion to ensure object consistency. For example, an update of an object followed by a read of the object value must be executed in the given order. Not following the order would return an incorrect value (i.e., the value before instead of after the update). Any negative consequences of delay and inconsistencies are avoided by ensuring that every request is processed in the order of arrival.

In early implementations of EAI technology publish/subscribe technology was used to establish the communication rules (Eugster et al. 2003). A back end application system that has interest in specific objects and changes to them subscribes to those. Every back end system publishes changes like object creation, update or deletion. If there is a subscription that matches a publication, the same changes are applied to the subscribing back end application system.

While publish/subscribe technology is quite useful, it is not able to implement business process integration (Bussler 2003). In this case several back end application systems have to be called in a specific order as defined by a business process. Sequencing, conditional branching and other control flow constructs define the invocation order of the back end application systems dynamically. Publish/subscribe cannot express this type of multi-step execution and more expressive technology is required.

EAI technology incorporated workflow management systems in order to enable more complex and multi-step executions. Business process based integration requirements are addressed through workflows. Through a workflow specification the overall invocation order can be established and complex processes like human resource hiring or supply-chain management are possible. Workflow steps interact with back end application systems through adapters whereby the workflow steps themselves are sequenced by workflow definitions.

While business process based integration is a significant step in the development of EAI technology, it is not sufficient in heterogeneous environments. For example, different back end application systems follow in general different data models. If this is the case the same type of data is represented in different ways. For example, one back end application system might implement an address as a record of street name, street number, city and zip code while another back end application system might implement an address as two strings, "address line 1" and "address line 2". If data is transmitted from one back end application system to another one the address has to be mediated from one representation to the other representation for the data to be understood by the receiving back end application system.

CURRENT DEVELOPMENTS AND FUTURE TRENDS

The latest development in EAI technology takes data heterogeneity into consideration and provides mediation technology. For example, Microsoft's Biztalk Server (Microsoft 2007) has a mapping tool that allows the graphical definition of mapping rules that can transform a message in one format into a message of a different format. Tools from other vendors follow the same approach. The research community is also working on the data mediation problem and an overview is given in (Rahm and Bernstein 2001).

Also, this new generation realizes that back end application systems expose not only data but also public processes that define interaction sequences of data (Bussler 2003). A public process allows to determine which messages a back end application systems sends or receives and in which order. This allows ensuring that all messages are exchanged appropriately with the back end application system.

Web Services are a relatively new approach to communicate data and message over the Internet (Alonso et al. 2004). Web Services are starting to be used in EAI technologies in several places. One is for the communication between back end application systems and the EAI technology itself. The other place is to define the interface of back end application systems as Web Services. In this case all back end application systems have a homogeneous interface technology they expose and an EAI technology does not have to deal with the huge variety of interfaces of back end application systems any more.

Architectural approaches based on Web Service technology are emerging. For example, BEA (BEA 2007) provides a whole suite of technologies that support the EAI problem based on Web Service technology. Other vendors like IBM (IBM 2007), Microsoft (Microsoft 2007) or Oracle (Oracle 2007) are also offering SOA based products. The architecture underlying Web Service based approaches is called Service-Oriented Architecture (SOA) and standardization work is ongoing, too (OASIS SOA RM 2007).

Further out are Semantic Web technology based approaches. They are in a research state today but can be expected to be available in commercial technology later on. A significant effort applying Semantic Web technology in order to solve the integration problems is the Web Service Modeling Ontology (WSMO) (WSMO 2007). This ontology, which is developed by a significant number of organizations, uses Semantic Web Technology to define a conceptual model to solve the integration problem that encompasses EAI integration. The main conceptual elements proposed are ontologies for defining data and messages, mediators for defining transformation, goals for dynamically binding providers and web services for defining the interfaces of communicating services. In order to define a specific EAI integration, the formal Web Service Modeling Language (WSML) (WSML 2007) is defined. This can be processed by a WSML parser. In order to execute B2B integration the Web Service Modeling Ontology Execution environment (WSMX)

Table 1. A Summary of Critical Issues

Autonomy Back end applications are autonomous in their state changes and EAI technology must be aware that since it cannot control the application's state changes	**Inconsistency** If a delay happens while changing different representations of the same object inconsistencies can occur if these different representations are accessed concurrently
Communication EAI technology must provide reliable and secure communication between back end application systems in order to ensure overall consistency	**Mediation** Due to heterogeneity of back end application systems objects are represented in different data models that have to be mediated if objects are sent between applications
Delay Changes of objects represented in several back end application systems might not happen instantaneously due to the time required to do the update and so a delay can occur between the update of the first and the last change	**Process management** Multi-step business processes across back end application systems require process management to implement the particular invocation sequence logic
Distribution Back end application systems are distributed in the sense that each has its own separate storage or database management system with each being separately controlled and managed	**Publish/subscribe** Back end application systems can declare interest in object changes through subscriptions that are matched with publications of available object changes
Duplication Objects that have to reside in several back end application systems might require duplication in order to ensure back end application state consistency	**Reliability** The integration of back end application systems must be reliable in order to neither loose data nor accidentally introduce data by retry logic in case of failures
Heterogeneity Back end application systems are implemented based on their own data model and due to the particular management focus the data models differ in general not complying with a common standard	**Replication** Replication is a mechanism that insures that changes of an object are automatically propagated to duplicates of this object. The various representations act as if they are one single representation
	Security Communication of data between back end application systems through EAI technology must ensure security to avoid improper access

(WSMX 2007) is developed. It serves as a runtime environment for integrations defined through WSML. Another Semantic Web technology based approach is OWL-S (OWL-S 2007).

Very recently a first standard in the Semantic Web Service space was established as a W3C recommendation. It is called SAWSDL and can be retrieved at (SAWSDL 2007). The main concept behind this standard is the augmentation of the WSDL approach for describing Web Service interfaces.

CRITICAL ISSUES OF EAI TECHNOLOGIES

The critical issues of Enterprise Application Integration are listed in Table 1. Current generations of EAI technology have to address these issues in order to be competitive and considered appropriate implementations to the EAI problem. The same applies to research. Recently research as well as new products focuses on Web Services as a means to integrate applications. However, Web Services are only a new technology and

all the requirements discussed above and issues listed below still apply. Semantic Web Services (WSMO 2007) addresses the requirements and issues taking all aspects into consideration.

CONCLUSION

Enterprise Application Integration (EAI) technology is essential for enterprises with more than one back end application system. Current EAI technology is fairly expressive being able to handle most of the integration tasks. Newer developments like Web Services (Web Services 2004) and Semantic Web Services (WSMO 2004) (OWL-S 2007) will significantly improve the situation by introducing semantic descriptions making integration more reliable and dependable.

REFERENCES

Alonso, G., Casati, F., Kuno, H., & Machiraju, V. (2004). *Web Services - Concepts, Architectures and Applications.* Springer-Verlag.

BEA (2007). *BEA Systems, Inc.* www.bea.com.

Bussler, C. (2003). *B2B Integration.* Springer-Verlag.

Gray, J., & Reuter, A. (1993). *Transaction Processing: Concepts and Techniques.* Morgan Kaufmann.

Eugster, P., Felber, P., Guerraoui, R., & Kermarrec, A.-M. (2003). The many faces of publish/subscribe. *ACM Computing Surveys (CSUR), 35*(2).

Hohpe, G., & Woolf, B. (2003). *Enterprise Integration Patterns: Designing, Building, and Deploying Messaging Solutions.* Addison-Wesley Professional.

IBM (2007). International Business Machines, Corp. www.ibm.com.

J2EE Connector Architecture (2004). J2EE Connector Architecture. java.sun.com/j2ee/connector/

Microsoft (2007). Microsoft Corporation. www.microsoft.com.

OASIS SOA RM (2007) www.oasis-open.org/committees/tc_home.php?wg_abbrev=soa-rm

Oracle (2007). Oracle Corporation. www.oracle.com.

OWL-S (2007). OWL-S. www.daml.org/services/owl-s/.

Rahm, E., & Bernstein, P. (2001). A survey of approaches to automatic schema matching. *The VLDB Journal, 10,* 334-350.

SAWSDL (2007). www.w3.org/2002/ws/sawsdl/.

Web Services (2007). *Web Service Activity.* www.w3.org/2002/ws/

WSML (2007). *Web Service Modeling Language.* www.wsmo.org/wsml

WSMO (2007). *Web Service Modeling Ontology.* www.wsmo.org

WSMX (2007). *Web Service Modeling Execution Environment.* www.wsmx.org

KEY TERMS

Adapters: Adapters are intermediate software that understand proprietary back end application interfaces and provide easy access interfaces for EAI technology integration.

Back End Application Systems: Software systems managing business data of corporations.

EAI Technology: Software systems that provide business-to-business integration functionality by sending and receiving messages and

retrieve and store them in back end application systems.

Process Management: Process management allows defining specific invocation sequences of back end application systems in context of EAI.

Publish/Subscribe: Publish subscribe is a technology whereby interest can be declared and is matched upon the publication of object changes.

Workflow Technology. Workflow technology is a software component that provides the languages and interpreters to implement process management.

Chapter LXXXIX
The Role of Rhetoric in Localization and Offshoring

Kirk St.Amant
East Carolina University, USA

INTRODUCTION

Globalization is increasingly integrating the world's economies and societies. Now, products created in one nation are often marketed to a range of international consumers. Similarly, the rapid diffusion of online media has facilitated cross-border interactions on social and professional levels. Differing cultural expectations, however, can cause miscommunications within this discourse paradigm. Localization – customizing a communiqué to meet cultural expectations – has thus become an important aspect of today's global economy. This essay examines localization in offshoring practices that could affect database creation and maintenance.

BACKGROUND

To understand localization, one must understand how rhetoric, or the way in which information is presented, can vary along cultural lines. Each culture has a set of rhetorical expectations, or conditions, for how to convey ideas effectively (Kaplan, 2001; Woolever, 2001). The more closely

a message meets the rhetorical expectations of a cultural group, the more likely members of that group will consider that message credible or usable (Bliss, 2001). If one does not meet a culture's rhetorical expectations, then the related group is likely to view a message as non-credible and will be less inclined to consider it. Moreover, if non-credible messages are associated with a particular product, audiences might consider that item as not worth purchasing (Ulijn & Strother, 1995).

Rhetoric and Verbal Communication

Differing rhetorical expectations means information considered credible by one cultural group might be deemed suspect or unusable by another (Woolever, 2001; Ulijn & St.Amant, 2000). Language is perhaps the most obvious factor related to credibility in cross-cultural exchanges. That is, if one wishes to develop informative materials for another culture, then concepts must be presented in the language used by that group. (If one wishes to target information for an audience in France, one should use the French language when presenting ideas.)

Using the correct language, however, is often not enough, for cultural groups can have different norms for how ideas should be expressed within a language (Ulijn, 1996; Kaplan, 2001; Driskill, 1996). These expectations often reflect deep-seated values or societal rules (Neuliep, 2000; Ferraro, 2002). It is thus often difficult for the members of one culture to anticipate the rhetorical expectations another cultural group associates with credible presentations.

These cultural-rhetorical differences, moreover, can assume a variety of forms. Some cultures tend to prefer more linear/focused presentations in which connections between ideas and conclusions are explicitly stated (Campbell, 1998; Ulijn & St.Amant, 2000). Other cultures, however, might prefer more indirect presentations in which individuals seem to go off on tangents or avoid directly stating facts or conclusions (Woolever, 2001; Ulijn & St.Amant, 2000; Campbell, 1998). These variations can cause misperceptions or confusion when different cultural groups interact. As Ulijn and St.Amant (2000) note, many Western cultures prefer a more direct presentation of information. In contrast, many Eastern cultures use a more indirect approach when sharing ideas. As a result, the indirect style used by Eastern cultures is often viewed as evasive or dishonest by Westerners who expect presenters to "get to the point." Conversely, many Easterners tend to view the direct presentation style of Western cultures as rude, for by directly stating information (stating the obvious), an individual is patronizing the audience. In such cases, failing to address the rhetorical expectations of the "other" culture can undermine the credibility of persons interacting in cross-cultural exchanges.

Rhetoric and Visual Communication

Interestingly, cultural rhetorical expectations are not restricted to verbal presentations. Rather, they also affect how different groups perceive and respond to visual displays. In some instances, the cultural expectations of what features an item – or visual representations of an item – should possess can differ from country to country. Such differences can affect how audiences perceive the credibility and the acceptability of visual displays (Kamath, 2000; Neuliep, 2000).

For example, the perception of a mailbox being a box that sits atop a post and that has a "red flag" on the side of it is, essentially, a U.S. one (Gillette, 1999). In other cultures, a mailbox might be a small door in a wall or even a cylindrical metal container. These design discrepancies could cause confusion when individuals use images to share information across cultures. Consider the following situation: Persons from different nations come to a web portal and expect to find a "mail" function on that portal. To address this

expectation, the portal's designers have included an "access mail" icon into the portal's design. The image used for this icon, however, is a U.S.-style mailbox. Unfortunately, this choice of image renders that depiction unrecognizable to persons from different cultures – cultures in which mailboxes have very different characteristics. Those individuals might then consider the associated web portal non-credible, for they perceive it as lacking the key design feature of a mail option. In this way, cultural differences can affect sites that use the "wrong" kind of image.

The presence or absence of a design feature, moreover, can affect the credibility of an image or of an overall website. Cultures, for example, can associate different meanings with the same color (Conway & Morrison, 1999; Ferraro, 2002). These associations could affect how individuals from different cultures perceive the meaning of a particular image. In the United States, for example, a blue ribbon usually indicates first place, while the same color ribbon in the United Kingdom often indicates second place. (In the U.K., a red ribbon often signifies "first place/winner.") The different associations related to the color blue could affect how persons from the U.S. vs. the U.K. view a "blue ribbon product" (first rate vs. second rate). In terms of web design, one might consider how the blue ribbon image associate with the Electronic Frontier Foundation's (EFF) Blue Ribbon Campaign for Free Speech Online – an icon proudly displayed by many U.S. websites – could affect how persons from the U.K. perceive the quality of information found on such sites (i.e., second rate information).

The expectations cultures can have for verbal and visual communication can affect the success of cross-cultural exchanges. For these reasons, individuals can benefit from practices that address cultural differences on both a verbal and a visual level. Localization offers a solution to such situations.

MAIN FOCUS OF THE CHAPTER

Localization is a process in which professionals design or revise materials to meet the rhetorical expectations of a particular cultural group (Esselink, 2000; Yunker, 2003). Often abbreviated as L10n (for the 10 letters between the "L" and the "n" in "localization"), localization generally involves one of two scenarios.

Post-Production Localization

In this scenario, an organization creates a product for a specific cultural audience (Esselink, 2000; Yunker, 2003). Both the product and its related documentation are then designed to meet the rhetorical expectations that particular culture associates with credibility. Over time, however, the company decides to market the product in other nations. Yet the design of the original item and its related materials might conflict with the rhetorical expectations of other cultural audiences. At this point, the organization has localizers re-design the product to make it appear credible to users from other cultures (Esselink, 2000; Yunker, 2003). Such a process involves converting information from the rhetorical styles used in *source* (original) materials to those of a different cultural audience (known as the *target audience*).

Such after-the-fact localization involves factors of translation (language) and visual design – both in terms of layout and image use. The text is often translated into the language of the desired target audience, and visual and design factors are revised or replaced to match the expectations that same audience (Esselink, 2000; Yunker, 2003). Ideally, this process is a relatively simple "find and replace" activity in which items in source materials are replaced with culture-specific ones designed for other audiences. Unfortunately, certain factors can affect the ease with which localizers accomplish such a process.

One of the more interesting problems in this context is *text expansion*. The idea is information that can be conveyed with a single word in one language might require multiple words to convey the same meaning in another language (Esselink, 2000; Yunker, 2003). The English-language expression "overtime pay" (2 words), for example, is often translated as "remuneration des herues supplementaires" (4 words) in French. Even in cases where a single word is used to convey the same concept in different languages, the length of the related word could vary considerably. The English word "Help" (4 letters), for example, can be translated into "Assistance" (10 letters) in French.

Text expansion is important, for it can affect the design of a product. Software, for example, might use drop down menus to access certain functions. The width of a menu, however, is often configured to be as wide or narrow as the longest line of text in that menu. The formatting of a drop down menu might therefore require redesign to accommodate the text expansion related to insert corresponding information presented in another language. Changes resulting from text expansion could also require individuals to re-configure an overall interface to accommodate all of the textual features associated with a particular program (Esselink, 2000; Yunker, 2003). Similarly, website features such as menu buttons or menu options might require redesign to accommodate text expansion (St.Amant, 2003). In such cases, the size and structure of menu bars or a web page might also need to be re-configured to accommodate such changes.

Other instances might require the localizer to remove or replace certain design features, such as pictures or icons, to address the sensibilities of different target cultures (Esselink, 2000; Yunker, 2003). The removal or replacement of such items, however, could create gaps in the overall interface or change the layout of or the flow of text within an interface depending on the size of the images that are removed or replaced (St.Amant, 2003). Additionally, the expectations of a target culture might be so different that creating a localized item involves an entire redesign of an overall product. These factors are further complicated by the fact that localization is often done on relatively tight schedules and with limited budgets. In such cases, localizers often find themselves balancing issues of quality with those of cost and time.

Simultaneous Localization

In a more ideal situation, localization occurs at the start of a product's development (Esselink, 2000; Yunker, 2003). In these cases, localizers do more than "revise" source texts created for one specific culture. Rather, localizers simultaneously generate original source materials for different cultural audiences. Such a process allows for the coordinated release of a product into different overseas markets. While such processes seem time and cost intensive, the quality of the resulting materials would generally be better than items localized after the fact.

Another benefit of simultaneous localization is localizers might notice design factors (e.g., the construction of an interface) that would render a product less usable by a certain cultural audience (Esselink, 2000; Yunker, 2003). The localizer could then present this problem to the actual designer, programmer, or engineer creating the product. Such consultations allow developers to revise a product during vs. after the initial design process and alleviates many problems related to re-engineering completed products.

In both scenarios (i.e., post-production and simultaneous localization), localization facilitates the flow of information from one cultural group to another by revising materials to meet different rhetorical expectations. While localization practices often focus on products or product-related

services, localization can be adapted to address processes. Perhaps the most influential reason for such an adaptation is the spread of a production approach known as offshoring.

FUTURE TRENDS

Growth in Global Online Access

At present, there are some 1.3 billion persons around the world who have access to the online environment (Internet Usage Statistics, 2008). While this number is only 1/6 of the world's population, international online access continues to increase with amazing speed. The number of Internet users in Australia, for example, grew by almost 350,000 between June and August of 2004 (Active Internet users July, 2004). In Canada, spending on IT infrastructure is expected to grow by $4 billion (US) between 2004 and 2008 (Canada Gets with IT, 2004) while in South Korea, roughly 75% of the population has access to broadband (Forsberg, 2005; Borland & Kanellos, 2004). This easy and cheap access to broadband has led to a rapid growth in the number of online gamers in South Korea and to the development of the nation's online educational institutions (Lessons from Afar, 2003; Borland & Kanellos, 2004).

The most astounding growth, however, is taking place in developing nations. The number of Internet connections in India, for example, was projected to grow by 28 percent between 2006 and 2007 (Burns, 2007), while online access in China grew from 2.1 million users in 1999 to 210 million in 2007 (The Flies Swarm In, 2000; Internet Usage in Asia, 2008). In Africa, some 44 million persons have regular Internet access, and Africa's number of internet users has grown by almost 880% between 2000 and 2007 (Internet Usage Statistics for Africa, 2008). Additionally, the number of individuals going online in Eastern Europe has grown exponentially in recent years, and Lithuania, Latvia, Poland, and Russia have all experience 300+ % growth in online access since 2000 (Burns, 2005).

Offshoring Examined

The increase in global access has prompted many companies to explore database creation and management processes involving online communication technologies. One of the most interesting approaches is *offshoring* – a process in which organizations use online media to exchange information and electronic products (e.g., software) with employees in other countries. Generally, companies in industrialized nations export work to employees in developing nations. These employees can often perform skilled IT work for a fraction of what it would cost in industrialized countries, and online media allow these workers to interact directly and quickly with one another and with the company sponsoring a project. These factors also mean

- Parts of a database creation or maintenance process can be performed simultaneously in different locations
- Materials related to database creation or management can be forwarded from one time zone to another and allow work to continue without pause

Ideally, offshoring creates quality database products that can be completed more quickly and cheaply than if the item had been produced domestically. These benefits have inspired private companies and municipal governments to use offshoring for a variety of database-related activities (The New Geography, 2003; Relocating the Back Office, 2003; Engardio, 2006). Trends in information technology and in demographics furthermore indicate offshoring practices will increase in coming years as more developing nations seek to enter this highly profitable area (St.Amant, 2007).

Successful offshoring involving database creation and management, however, depends upon effective communication. In offshoring projects, ignored or misinterpreted information could result in employees performing a process incorrectly. Moreover, the speed with which such processes take place, when combined with the physical and cultural distance separating participants, means mistakes might go unnoticed until it is too late. Information therefore needs to be conveyed in a rhetorically effective manner that encourages participants to make use of it. Localization, in turn, becomes an important process, for it increases the chances workers from different cultures will use essential materials.

Localization and Offshoring Practices

Localization can take place at two levels within offshoring. The first level is interface design and involves the medium through which individuals interact. In many cases, participants collaborating on offshoring projects come to a central online location (e.g., a company portal or web page) to collect, present, or exchange information related to a particular product or production process. These interfaces, however, must be designed to encourage use if all involved parties perform their tasks correctly. Localization can facilitate such information exchanges by creating interfaces that meet the credibility expectations of different cultures and thus encourages effective use of such interfaces.

The second level of localization involves the use of online media (e.g., email, chats, and bulletin boards) to share information. In these exchanges, participants need to know two important pieces of rhetorical information:

1. How to draft online messages that recipients from other cultural groups will consider credible and worth responding to
2. How to interpret verbal messages constructed according to different cultural rhetorical norms

In both cases, localizers can provide participants with communication "cheat sheets" that address these factors. The first cheat sheet would explain how to draft messages that addresses the rhetorical expectations of other cultural groups involved in an offshoring process. The second cheat sheet would tell users how to interpret the meaning individuals from other cultures convey via a particular rhetorical structure. By teaching others how to address and interpret such rhetorical factors, localization enhances communication activities within offshoring. Such approaches contribute to the success of projects involving offshoring.

CONCLUSION

The global nature of modern business means individuals must now consider interactions in terms of international audiences. These audiences can have different expectations of how information should be presented. Fortunately, localization can facilitate cross-cultural interactions by converting materials from one cultural rhetorical style to another. While localization has traditionally dealt with products, the shift to offshoring allows localization to address processes as well. Thus, by understanding what localization is and how it works, organizations can increase their successes in a variety of cross-cultural communication situations related to database creation and maintenance.

REFERENCES

Active Internet users by country, July 2004. (2004, August 25). *ClickZ*. Retrieved July 10, 2008, from http://www.clickz.com/stats/sectors/geographics/article.php/3397231

Bliss, A. (2001). Rhetorical structures for multilingual and multicultural students. In C. G. Panetta (Ed.), *Contrastive rhetoric revisited and redefined* (pp. 15-30). Mahwah, NJ: Lawrence Erlbaum Associates.

Borland, J. & Kanellos, M. (2004, July 28). South Korea leads the way. *News.Com*. Retrieved November 2, 2005, from http://news.com.com/South+Korea+leads+the+way/2009-1034_3-5261393.html

Burns, E. (2007, July 5). Urban Indian web users. *ClickZ*. Retrieved April 3, 2008, from http://www.clickz.com/showPage.html?page=3626341

Campbell, Charles P. (1998). Rhetorical ethos: A bridge between high-context and low-context cultures? In S. Niemeier, C. P. Campbell, & R. Dirven (Eds.), *The Cultural Context in Business Communication* (pp. 31-47). Philadelphia, PA: John Benjamin.

Canada gets with IT. (2004, September 29). *Insurance-Canada.ca*. Retrieved February 5, 2005, from http://www.insurance-canada.ca/ebusiness/canada/eMarketer-Canada-IT-409.php

Conway, W. A. & Morrison, T. (1999). The color of money. *Global business basics*, Retrieved December 10, 1999 from, http://www.getcustom.com/omnibus/iw0897.html

Driskill, L. (1996). Collaborating across national and cultural borders. In D. C. Andrews (Ed.), *International dimensions of technical communication* (pp. 21-44). Arlington, VA: Society for Technical Communication.

Drucker, P. (2001, Nov. 1). The next society: A survey of the near future. *The Economist*, pp. 3-5.

Engardio, P. (2006). Let's offshore the lawyers. *BusinessWeek*, Retrieved April 3, 2008 from http://www.businessweek.com/magazine/content/06_38/b4001061.htm?chan=search

Esselink, B. (2000). *A practical guide to localization*. Philadelphia, PA: John Benjamins.

Ferraro, G. (2002). *Global brains: Knowledge and competencies for the 21st century*. Charlotte, NC: Intercultural Associates.

Forsberg, B. (2005, March 13). The future is South Korea: Technology firms try out latest in world's most wired society. *San Francisco Chronicle*. Retrieved November 3, 2005, from http://www.sfgate.com/cgi-bin/article.cgi?f=/c/a/2005/03/13/BROADBAND.TMP

Gillette, D. (1999, December). Web design for international audiences. *Intercom*, pp. 15-17.

Internet usage in Asia. (2008). *Internet World Stats*. Retrieved April 3, 2008, from http://www.internetworldstats.com/stats3.htm

Internet usage statistics. (2008). *Internet World Stats*. Retrieved April 3, 2008, from http://www.internetworldstats.com/stats.htm

Internet usage statistics for Africa. (2008). *Internet World Stats*. Retrieved April 3, 2008, from http://www.internetworldstats.com/stats1.htm

Kamath, G. R. (2000, May). The India paradox. *Intercom*, pp. 10-11.

Kaplan, R. B. (2001). Foreword: What in the world is contrastive rhetoric? In C. G. Panetta (Ed.), *Contrastive rhetoric revisited and redefined* (pp. vii-xx). Mahwah, NJ: Lawrence Erlbaum Associates.

Lessons from afar. (2003, May 8). *The Economist*. Retrieved September 12, 2005, from http://www.economist.com/business/globalexecutive/displaystory.cfm?story_id=1762562&tranMode=none

Lui, K. M. & Chan, K. C. C. (2003). Inexperienced software team and global software team. In A. Gunasekaran, O. Khalil, and S. M. Rahman (Eds.), *Knowledge and Information Technology Management: Human and Social Perspectives* (pp. 305-323). Hershey, PA: Idea Group Publishing.

Neuliep, J. W. (2000). *Intercultural communication: A contextual approach*. Boston: Houghton Mifflin.

Relocating the Back Office. (2003, 11 Dec.). *The Economist,* Retrieved December 20, 2003 from, http://www.economist.com/displaystory.cfm?story_id=2282381

St.Amant, K. (2007). Outsourcing: Perspectives, practices, and projections, *IEEE Transactions on Professional Communication*, *50*, 81-84.

St. Amant, K. (2003). Designing web sites for international audiences, *Intercom*, 15-18. The flies swarm in. (2000). *The Economists.* Retrieved April 3, 2008, from http://www.economist.com/world/displaystory.cfm?story_id=E1_GPPG

The new geography of the IT industry. (2003, 17 July). *The Economist,* Retrieved December 20, 2003, from http://www.economist.com/displaystory.cfm?story_id=S%27%29HH%2EQA%5B%21%23%40%21D%0A

Ulijn, J. M. (1996). Translating the culture of technical documents: Some experimental evidence.

In D. C. Andrews (Ed.), *International dimensions of technical communication* (pp. 69-86).Arlington, VA: Society for Technical Communication.

Ulijn, J. M. & St.Amant K. (2000). Mutual intercultural perception: How does it affect technical communication–some data from China, the Netherlands, Germany, France, and Italy. *Technical Communication*, *47*(2), 220-37.

Ulijn, J. M., & Strother, J. B. (1995). *Communicating in business and technology: From psycholinguistic theory to international practice.* Frankfurt, Germany: Peter Lang.

Wired China (2000, July 22). *The Economist,* pp. 24-28.

Woolever, K. R. (2001). Doing global business in the information age: Rhetorical contrasts in the business and technical professions. In C. G. Paneta (Ed.), *Contrastive Rhetoric Revisited and Redefined* (pp. 47-64). Mahwah, NJ: Lawrence Erlbaum Associates.

Yunker, J. (2003). *Beyond borders: Web globalization strategies.* Boston: New Riders.

KEY TERMS

Localization: the process of revising materials designed for one culture to meet the communication expectations of a different culture.

Offshoring: production process in which individuals in other nations perform work for an organization.

Online Media: communication technologies that use the Internet or the World Wide Web to present or exchange information.

Rhetoric: the manner in which information is presented.

Source/Source Text/Source Materials: original materials designed for a specific cultural group.

Target Culture: the culture for which one revises source materials in the localization process.

Text Expansion: when one language requires more words to express the same meaning than does another language.

Chapter XC
Adaptive XML-to-Relational Storage Strategies

Irena Mlynkova
Charles University, Czech Republic

INTRODUCTION

Without any doubt, the eXtensible Markup Language (XML) (Bray et al., 2006) is currently one of the most popular formats for data representation. Its wide popularity naturally invoked an enormous endeavour to propose faster and more efficient methods and tools for managing and processing of XML data. Soon it was possible to distinguish several different directions. The four most popular ones are methods which store XML data in a classical file system, methods which store and process XML data using a relational database management system, methods which exploit a pure object-oriented approach and native methods that use special indices, numbering schemas and/or data structures proposed or suitable particularly for tree structure of XML data.

Naturally, each of these approaches has particular advantages or disadvantages. In general, especially the popularity of file system-based and pure object-oriented methods is low. The former ones suffer from inability of querying without any additional pre-processing of the data, whereas the latter approach fails especially in finding a corresponding efficient and comprehensive tool.

The highest-performance techniques are the native ones, since they are proposed particularly for XML processing and do not need to artificially adapt existing structures to a new purpose. Nevertheless, the most practically used methods exploit features of relational databases. Although the scientific world has already proven that native XML strategies perform much better, they still lack one important aspect – a reliable and robust implementation verified by years of both theoretical and practical effort.

Currently there exists a plenty of existing works concerning database-based[a] XML data management. All the major database vendors support XML and even the SQL (Structured Query Language) standard has been extended by a new part SQL/XML (ISO/IEC 9075-14, 2003) which introduces new XML data type and operations for XML data manipulation. But, although the amount of existing works is enormous, there is probably no universally efficient strategy. Each of the existing approaches is suitable for selected applications, but, at the same time, there can usually be found cases when it can be highly inefficient. An illustrative example is updatability of data, where the efficient storage strategies significantly differ if the feature is required or not.

The aim of this text is to provide an overview of existing XML-to-relational storage strategies. We will overview their historical development and provide a more detailed discussion of the currently most promising ones – the adaptive methods. Finally, we will outline possible future directions.

BACKGROUND

The main concern of the database-based XML techniques is the choice of the way XML data are stored into relations, so-called *XML-to-relational mapping* or *schema decomposition to relations.* The strategy in first approaches to XML-to-relational mapping, so-called *generic* (e.g. Florescu et al., 1999), was based purely on the data model of XML documents. The methods were able to store any kind of XML data since they viewed XML documents as general labelled trees. But, the efficiency of query evaluation was quite low due to numerous join operations or the increase of efficiency was gained at the cost of increase of space overhead.

Hereafter, the scientists came with a natural idea to exploit structural information extracted from XML schemas of XML data, usually expressed in DTD (Document Type Description) (Bray et al., 2006) or XML Schema (Thompson et al., 2004; Biron et al., 2004) language. All the so-called *schema-driven* approaches (e.g. Shanmugasundaram et al., 1999) were based on the same idea that the structure of the target relational schema can be created according the structure of the source XML schema. Assuming that a user specifies the XML schema as precisely as possible to specify the related data, we can get also more precise relational XML schema. The problem is that DTDs are usually too general. The extensive examples are recursion or * operator which, in general, enable to specify infinitely deep or wide XML documents. According to analyses of real-world XML data (Mlynkova et al., 2006) in both the cases the respective XML documents are much simpler and, thus, the effort spent on processing all the complex schema constructs is useless.

A slight solution to the problem brought schema-driven approaches exploiting information extracted from XML Schema definitions (e.g. Mlynkova et al., 2004). The XML Schema language enables to describe the structure of XML data more precisely, especially in case of data types. But, although its expressive power is higher, most of the new constructs can be considered as "syntactic sugar". Hence, exploitation of XML Schema constructs does not have a key impact on efficiency of XML processing.

A different type of improving of the fixed mapping strategies brought *constraint-preserving* methods (e.g. Chen et al., 2003; Davidson et al.,

2007) which focus on preservation of constraints specified in the source XML schema, such as, e.g., key and foreign key constraints, functional dependencies or semantic constraints, in the target relational schema. Such approaches enable to reduce redundancy as well as improve efficiency of selected queries and, especially, update operations. But, in general, the methods suffer from the same disadvantages as all the described methods.

Consequently, it became obvious that a *fixed* universally efficient approach does not exist and to achieve a higher performance it is necessary to adapt the relational schema to requirements of the current application. Consequently, the *adaptive* methods have appeared and they are still considered as the most efficient approaches to the XML-to-relational mapping. The first and natural idea of adaptability was to leave all the decisions in hands of a user who specifies both the target relational schema and the required mapping. We speak about so-called *user-defined* methods (see Amer-Yahia, 2003). But the problem is that these methods require a user mastering two complex technologies. Consequently, two different types of approaches have appeared. In case of *user-driven* methods (Amer-Yahia et al., 2004; Balmin et al., 2005; Mlynkova, 2007) a user is provided with a fixed mapping strategy and can specify just local changes where appropriate. On the other hand, *cost-driven* approaches (Klettke et al., 2000; Ramanath et al., 2003; Xiao-ling et al., 2003; Zheng et al., 2003) search a space of possible XML-to-relational mapping strategies and choose the one which conforms to the current application the most. The application is usually specified using a sample set of XML documents and XML queries.

ADAPTIVE APPROACHES

As we have described, the existing representatives of adaptive methods can be divided into cost-driven and user-driven. Both approaches are based on the idea of exploiting additional user-given information and they appropriately adapt the target database schema. In the former case it is extracted from a sample set of XML documents and/or XML queries, in the latter case it is specified by user-given annotations, i.e. the user directly specifies the required changes of a default mapping.

Cost-Driven Techniques

Cost-driven techniques can choose the best storage strategy for a particular application automatically, without any interference of a user. Apart from various parameters of particular algorithms, the user can influence the mapping process through the provided XML schema, set of sample XML documents and XML queries. But after providing the input information, the algorithms cannot be influenced further.

We can divide the existing cost-driven approaches into two types – *single-candidate* and *multiple-candidate*. Single-candidate approaches involve a straightforward mapping algorithm which provides a single output relational schema on the basis of input data. Multiple-candidate approaches also process the input data, but before providing the resulting relational schema, they evaluate multiple candidate solutions and choose the one which suits the considered application the most.

One of the first attempts of a cost-driven adaptive approach and, at the same time, probably the only known representative of single-candidate approaches is a method called *Hybrid object-relational mapping* (Klettke et al., 2000). It is based on the observation that if XML documents are mostly semi-structured, a classical decomposition of unstructured or semi-structured XML parts into relations leads to inefficient query processing caused by plenty of join operations. Hence, the algorithm stores well structured parts into relations and semi-structured parts using an *XML*

data type, which supports path queries and XML-aware full-text operations. The main concern is to identify the structured and semi-structured parts of the input XML schema, whereas the proposed method is based on the analysis of their amount and way of occurrence in the sample XML documents and XML queries.

On the other hand, each of the existing multiple-candidate techniques (Ramanath et al., 2003; Xiao-ling et al., 2003; Zheng et al., 2003) can be characterized by:

- an initial input XML schema S_{init},
- a set of XML schema transformations $T = \{t_1, t_2, \ldots, t_n\}$, where $\forall i : t_i$ transforms a given schema S into a schema S',
- a fixed XML-to-relational mapping function f_{map} which transforms a given XML schema S into a relational schema R,
- a set of input sample data D_{sample} consisting of a set of sample XML documents D and XML queries Q which characterize the future application and
- a cost function f_{cost} which evaluates the efficiency of a given relational schema R with regard to the set D_{sample}.

Examples of schema transformations can be *inlining/outlining* of an element into/out of the table of its parent element, or storing a whole schema fragment into a single BLOB column.

The required result is an optimal relational schema R_{opt}, i.e. a schema, where $f_{cost}(R_{opt}, D_{sample})$ is minimal.

A naive, but illustrative, cost-driven storage strategy is based on the idea of "brute force". It first generates a set of possible XML schemas Σ using transformations from set T and starting from initial schema S_{init}. Then it searches for schema $s \in \Sigma$ with minimal cost $f_{cost}(f_{map}(s), D_{sample})$ and returns the corresponding optimal relational schema $R_{opt} = f_{map}(s)$. It is obvious that the complexity of such algorithm strongly depends on the set T. It can be proven that even a simple set of transformations causes the problem of finding the optimal schema to be NP-hard (Xiao-ling et al., 2003). Thus, in fact, the existing techniques search for a suboptimal solution using various heuristics, greedy strategies, approximation algorithms, terminal conditions etc.

User-Driven Techniques

Undoubtedly, the most flexible adaptive approaches to XML-to-relational mapping are user-defined ones which leave the whole process in hands of a user who defines both the target database schema and the required mapping. Probably due to simple implementation they are especially popular and supported in most commercial database systems. The problem is that such approach assumes users skilled in two complex technologies –relational databases and XML. Furthermore, for more complex applications the design of an optimal relational schema is generally not an easy task.

The main difference of user-driven methods is that the user can influence a default fixed mapping strategy using annotations which specify the required mapping for particular schema fragments. The set of allowed mappings is naturally limited, but still enough powerful to define various mapping strategies. In other words, the user helps the mapping process, he/she does not perform it directly.

Each of the techniques is characterized by:

- an initial XML schema S_{init},
- a set of fixed XML-to-relational mappings $f_{map}^{1}, f_{map}^{2}, \ldots, f_{map}^{n}$,
- a set of annotations A, each of which is specified by name, target, allowed values and function and
- a default mapping strategy f_{def} for not annotated fragments.

Examples of annotations are usually similar to schema transformations of cost-driven strate-

gies. But, in general, an annotation can specify any kind of a fixed storage strategy.

The existing approaches can be also divided according to strategy used to provide the resulting relational schema. We differentiate so-called *direct* and *indirect* mapping.

The approaches exploiting direct mapping (Amer-Yahia et al., 2004; Balmin et al., 2005) are based on a straightforward algorithm. Firstly, the initial schema S_{init} is annotated with user-required mapping strategies from A. Secondly, the correctness of the annotations is checked. And finally, the XML schema is mapped – for annotated schema fragments the specified mapping strategies f_{map}^{1}, f_{map}^{2}, ..., f_{map}^{n} are used, for not annotated schema fragments f_{def} is used. The existing approaches differentiate mainly in the supported mapping strategies.

Indirect mapping strategies try to exploit the user-provided information as much as possible. They are based on the idea that the annotations can not only be directly applied on particular schema fragments, but the information can be exploited for mapping the remaining schema fragments. The first and probably still the only representative of indirect user-driven strategies is system *UserMap* (Mlynkova, 2007). The indirect mapping proposed in the system is based on a simple observation that the user-provided schema annotations can be viewed as hints how to store particular schema fragments. And if we know the required mapping strategy for a particular schema fragment, we can assume that structurally (or semantically) similar schema fragments should be stored in a similar way. Hence, the system iteratively searches for similar schema fragments to find more suitable mapping strategy.

FUTURE TRENDS

Although each of the existing approaches brings certain interesting ideas and optimizations, there is still a space of possible future improvements. Apart from minor open issues, such as speeding up the search strategy of cost-driven methods, simplifying user interaction of user-driven strategies or possibility of combining of the approaches, the most striking disadvantage of adaptive methods is the problem of possible changes of both XML queries and XML data that can lead to crucial worsening of their efficiency.

The solution to this problem – i.e. a system that is able to adapt dynamically – is obvious and challenging, but it is not an easy task. The first related problem is how to find the new mapping strategy efficiently. Naturally, we could repeat the whole search strategy after a certain amount of operations over the current relational schema has been performed, but this naive strategy would be quite inefficient. A better solution seems to be exploitation of a search strategy that does not have to be restarted when the related information change, i.e. a strategy which can be applied on dynamic systems, such as, e.g., ACO (Ant Colony Optimization) meta-heuristic (Dorigo et al., 2006).

The dynamic system should also avoid total reconstructions of the whole relational schema and corresponding necessary reinserting of all the stored data or such operation should be done only in very special cases and not often. On the other hand, this "brute-force" approach can serve as a good inspiration. It is possible to suppose that changes especially in case of XML queries will not be radical, but will have a gradual progress. Thus the changes of the relational schema will be mostly local and we can apply the expensive reconstruction just locally. Furthermore, we can again exploit the idea of pattern matching and try to find the XML pattern defined by the modified schema fragment in the rest of the schema.

Another question is how often should be the relational schema reconstructed. The natural idea is of course "not too often". But, a research can be done on the idea of performing gradual minor changes. It is probable that such approach will lead to less expensive (in terms of reconstruction)

and, at the same time, more efficient (in terms of query processing) system. The former hypothesis should be verified; the latter one can be almost certainly expected. The key issue is how to find a reasonable compromise.

CONCLUSION

The main goal of this text was to describe and discuss the current state of the art and open issues of database-based XML-processing methods with the focus of the currently most efficient representatives, so-called adaptive approaches. First, we have provided a motivation why this topic should be ever studied and why adaptive methods are so promising. Then we have provided an overview and classification of the existing approaches and their features. And, finally, we have discussed the corresponding open issues and their possible solutions.

ACKNOWLEDGEMENT

This work was supported in part by Czech Science Foundation (GACR), grant number 201/06/0756.

REFERENCES

Amer-Yahia S. (2003). *Storage Techniques and Mapping Schemas for XML*. Technical Report TD-5P4L7B, AT&T Labs-Research.

Amer-Yahia, S., Du, F., & Freire, J. (2004). A Comprehensive Solution to the XML-to-Relational Mapping Problem. In *WIDM'04: Proceedings of the 6th Annual ACM International Workshop on Web Information and Data Management,* (pp. 31-38), New York, NY, USA: ACM Press.

Balmin, A., & Papakonstantinou, Y. (2005). Storing and Querying XML Data Using Denormalized Relational Databases. *The VLDB Journal, 14*(1), 30-49.

Biron, P. V., & Malhotra, A. (2004). *XML Schema Part 2: Datatypes (Second Edition).* W3C Recommendation. www.w3.org/TR/xmlschema-2/.

Bray, T., Paoli, J., Sperberg-McQueen, C. M., Maler, E., & Yergeau, F. (2006). *Extensible Markup Language (XML) 1.0 (Fourth Edition).* W3C Recommendation.http://www.w3.org/TR/REC-xml/.

Chen, Y., Davidson, S., Hara, C., & Zheng, Y. (2003). RRXS: Redundancy Reducing XML Storage in Relations. In *VLDB'03: Proceedings of the 29th international Conference on Very Large Data Bases, 29,* 189–200, Berlin, Germany. VLDB Endowment.

Davidson, S., Fan, W., & Hara, C. (2007). Propagating XML Constraints to Relations. *J. Comput. Syst. Sci., 73*(3), 316–361.

Dorigo, M., Birattari, M., & Stutzle, T. (2006). *An Introduction to Ant Colony Optimization.* Technical Report 2006-010, IRIDIA, Bruxelles, Belgium.

Florescu, D., & Kossmann, D. (1999). Storing and Querying XML Data using an RDMBS. *IEEE Data Eng. Bull., 22*(3), 27–34.

ISO/IEC 9075-14:2003. *Part 14: XML-Related Specifications (SQL/XML).* International Organization for Standardization.

Klettke, M., & Meyer, H. (2000). XML and Object-Relational Database Systems – Enhancing Structural Mappings Based on Statistics. In *Lecture Notes in Computer Science, 1997,* 151–170. Springer-Verlag.

Mlynkova, I. (2007). A Journey towards More Efficient Processing of XML Data in (O)RDBMS. In *CIT'07: Proceedings of the 7th IEEE International Conference on Computer and Information*

Technology, (pp. 23–28), Aizu-Wakamatsu City, Fukushima, Japan. IEEE Computer Society.

Mlynkova, I., Toman, K., & Pokorny, J. (2006). Statistical Analysis of Real XML Data Collections. In *COMAD'06: Proceedings of the 13th International Conference on Management of Data*, (pp. 20–31), New Delhi, India. Tata McGraw-Hill Publishing Company Limited.

Mlynkova, I., & Pokorny, J. (2004). From XML Schema to Object-Relational Database – an XML Schema-Driven Mapping Algorithm. In *ICWI'04: Proceedings of the 3rd IADIS International Conference WWW/Internet*, (pp. 115–122), Madrid, Spain. IADIS.

Pemberton, S. et al. (2002) XHTML *1.0 The Extensible HyperText Markup Language (Second Edition)*. W3C. http://www.w3.org/TR/xhtml1/.

Ramanath, M., Freire, J., Haritsa, J., & Roy, P. (2003). Searching for Efficient XML-to-Relational Mappings. In *XSym'03: Proceedings of the 1st International XML Database Symposium, 2824*, 19–36, Berlin, Germany. Springer-Verlag.

Shanmugasundaram, J., Tufte, K., Zhang, C., He, G., DeWitt, D. J., & Naughton, J. F. (1999). Relational Databases for Querying XML Documents: Limitations and Opportunities. In *VLDB'99: Proceedings of 25th International Conference on Very Large Data Bases*, (pp. 302-314), San Francisco, CA, USA. Morgan Kaufmann Publishers.

Thompson, H. S., Beech, D., Maloney, M., & Mendelsohn, N. (2004). *XML Schema Part 1: Structures (Second Edition)*. W3C Recommendation. www.w3.org/TR/xmlschema-1/.

Xiao-ling, W., Jin-feng, L., & Yi-sheng. D. (2003). An Adaptable and Adjustable Mapping from XML Data to Tables in RDB. In *Proceedings of the VLDB 2002 Workshop EEXTT and CAiSE 2002 Workshop DTWeb*, (pp. 117–130), London, UK. Springer-Verlag.

Zheng, S., Wen, J., & Lu, H. (2003). Cost-Driven Storage Schema Selection for XML. In *DASFAA'03: Proceedings of the 8th International Conference on Database Systems for Advanced Applications*, (pp. 337–344), Kyoto, Japan. IEEE Computer Society.

KEY TERMS

Adaptive XML-to-Relational Mapping: An XML-to-relational mapping which is based on exploitation of additional information, such as, e.g., sample XML documents, sample XML queries, user-specified requirements etc. and respectively adapts the target relational schema.

Cost-Driven XML-to-Relational Mapping: An XML-to-relational mapping which searches a space of possible fixed XML-to-relational mapping methods and chooses the one which conforms to the current application, i.e. XML documents and XML queries, the most.

Fixed XML-to-Relational Mapping: An XML-to-relational mapping which is based on a fixed set of rules how to create the target database schema.

Generic XML-to-Relational Mapping: An XML-to-relational mapping which is based purely on a selected kind of data model of XML documents.

Schema-Driven XML-to-Relational Mapping: An XML-to-relational mapping where the target relational schema is defined according to the structure of the source XML schema of XML data.

User-Defined XML-to-Relational Mapping: An XML-to-relational mapping where the user specifies both the target relational schema and the required mapping manually.

User-Driven XML-to-Relational Mapping: An XML-to-relational mapping where the user is provided with a fixed mapping strategy and can specify its local changes where appropriate.

XML-to-Relational Mapping: A method which specifies how XML data are stored into relations of a relational database management system. It involves a definition of the target database schema and a way how the data are stored into its relations. Related problems are data retrieval and data updates, but they are usually determined directly by the storage strategy.

ENDNOTE

[a] In the rest of the text the term "database" represents a relational database management system (RDBMS).

Chapter XCI
Innovative Access and Query Schemes for Mobile Databases and Data Warehouses

Alfredo Cuzzocrea
University of Calabria, Italy

1. INTRODUCTION

Thanks to the explosion of the wireless technology, mobile environments are becoming the leading software platforms for extracting knowledge and interacting with enterprise information systems. **Data and services availability** at all times is the major benefit coming from such deployment scenario, but new research challenges pose serious limitations concerning data engineering issues. In fact, although if one can suppose that re-writing and re-adapting data structures, algorithms, and data reliability/dependability schemes is the natural way to support efficient **data management on mobile environments**, new issues and old limitations arise, particularly for what concerns with **data availability and consistency** in wireless network environments.

Furthermore, for a **mobile application**, design and deployment requirements strongly depend on its nature. For a **service-oriented application**, efficient service activation, and fast availability of data on which services run play a dominant role; on the contrary, for a **data-intensive application**, distributed storage, indexing, and querying are the most important (and problematic) issues. However, for any mobile application, ***location-aware services*** *providing* (including data services) is the common deployment requirement to be satisfied. According to these considerations, every mobile software application should be able to provide data and services in *any place* and, above all, in *any time*.

Database Systems (*DBS*) and *Data Warehouse Systems* (*DWS*), which are usually built on the top of very large (and, very often, heterogeneous) data sources, are ready to gain innovative and important improvements by integrating the wireless environment into their application scope. In fact, these systems, which, without loss of generality, we can name as ***Data-Intensive Systems*** (*DIS*), can find in the mobile ones the natural front-end devices for making their impact on a large variety of applications (falling in even emerging contexts

such as *Business Intelligence* (*BI*)) very successful. In fact, the development and, above all, the usage of many commercial DIS have shown that an on-site, just-in-time, fast, even approximate, computation (achievable through a wireless network infrastructure) can be often more profitable than a distant-in-time, data-intensive computation (achievable through a wired network infrastructure). In other words, mobile environments can reasonably be considered as a *plus-value* for data-intensive applications and systems.

In this respect, the data model assumes a critical role for DIS. As an example, the *multidimensional data model* has became a leading solution for DIS, thanks to its capability of representing and processing data according to a multidimensional and multi-resolution vision of data. This model has been adopted in several systems, with real benefits, such as *e*-banking systems, trading-on-line systems, basket analysis systems etc. ***OnLine Analytical Processing*** (OLAP) (Gray et al., 1997) is the leading technology for the multidimensional data model. Many of the above-mentioned systems integrate in their core layer an OLAP engine that offers storage and indexing functionalities, query and integration capabilities over multidimensional data.

Similarly to traditional contexts, data accessing and querying are the most relevant aspects to be considered, as they heavily affect the performance of knowledge extraction tasks defined on top of them. A very important solution to this problem is represented by the idea of using ***data compression techniques***, which allow us to improve the performance of data access and query activities, as well as to reduce the cost of transmitting data that, due to specific technological constraints, assumes a critical role in mobile environments. On the other hand, the data compression idea is not new in the context of mobile computing, and it has been adopted in others experiences having some relationships with our investigated scenario, i.e. data-intensive mobile applications and systems. For instance, (Franz & Kistler, 1997) proposes the *slim binary representation*, a technique for compressing mobile code on the basis of adaptive compression of *syntax trees*. This technique allows the performance of code running on handheld devices to be improved significantly. Also, *data summarization*, which is the extraction of a text summary from a given text under the constraint of maintaining an "equal" information content, is a research topic related to data compression issues. In (Buyukkokten et al., 2001), authors propose data summarization techniques for supporting efficient browsing of Web hypermedia via handheld devices.

All considering, we can claim that *data compression represents an innovative access and query scheme for mobile databases and data cubes*. This intuition is also well-motivated by considering the nature of analysis supported by DIS against huge amounts of data. This analysis is usually *qualitative*, meaning that in the query and report phases, decimal precision is not need as, for instance, managers and analysts are often more interested in performing *trend* analysis rather than *punctual* analysis on business data. Following these considerations, providing *approximate answers* to queries against massive databases and data warehouses is more convenient and feasible rather than computing exact answers, mainly because the latter are resource-intensive in terms of both spatial and temporal computational needs. This evidence is also the fundamental motivation of the proliferating of a large number of ***approximate query answering techniques*** (e.g., (Poosala & Ioannidis, 1997; Gibbons & Matias, 1998; Acharya et al., 1999; Vitter et al., 1998; Ioannidis & Poosala, 1999; Poosala & Ganti, 1999; Gunopulos et al., 2000; Chaudhuri et al., 2001; Garofalakis & Kumar, 2004; Cuzzocrea, 2005a; Cuzzocrea, 2005b; Cuzzocrea, 2006; Cuzzocrea & Wang, 2007)) observed in the last years, whose main goal is devising representation techniques and query algorithms for obtaining fast and approximate answers from huge amounts of data.

2. BACKGROUND

A *database* $\mathcal{D}$ is a tuple $\mathcal{D} = \langle \mathcal{W}, \mathcal{I}, \mathcal{F} \rangle$ such that (*i*) $\mathcal{W}$ is the *schema* of $\mathcal{D}$; (*ii*) $\mathcal{I}$ is the *instance* of $\mathcal{D}$, i.e. its realization in terms of collections of tuples adhering to $\mathcal{W}$; (*iii*) $\mathcal{F}$ is the collection of *functional dependencies* defined over $\mathcal{W}$. In turn, $\mathcal{W}$ is a collection of *relation schemas* $\mathcal{W} = \{\mathcal{T}_0, \mathcal{T}_1, \ldots, \mathcal{T}_P\}$, with $P = |\mathcal{W}| - 1$, such that $\mathcal{T}_i$, with $0 \leq i \leq P$, is defined as a tuple $\mathcal{T}_i = \langle \mathcal{K}, \mathcal{A}_{i,0}, \mathcal{A}_{i,1}, \ldots, \mathcal{A}_{i,G} \rangle$, with $G = |\mathcal{T}_i| - 1$, such that $\mathcal{K}$ is the *key* of $\mathcal{T}_i$, and $\mathcal{A}_{i,j}$ is the *j-th attribute* of $\mathcal{T}_i$. A functional dependence is expressed as a *logical rule* over attributes of one or more relation schemas. The instance of a relation scheme $\mathcal{T}_i$ is named as *relation*, and denoted by $\mathcal{R}_i$.

A *data cube* $\mathcal{L}$ is a tuple $\mathcal{L} = \langle C, \mathcal{J}, \mathcal{H}, \mathcal{M} \rangle$, such that: (*i*) C is the data domain of $\mathcal{L}$ containing (OLAP) *data cells*, which are the basic SQL aggregations of $\mathcal{L}$ computed against the relational data source S alimenting $\mathcal{L}$; (*ii*) $\mathcal{J}$ is the set of *dimensions* of $\mathcal{L}$, i.e. the *functional attributes* (of S) with respect to which the underlying OLAP analysis is defined (in other words, $\mathcal{J}$ is the set of attributes with respect to which relational tuples in S are aggregated); (*iii*) $\mathcal{H}$ is the set of *hierarchies* related to the dimensions of $\mathcal{L}$, i.e. hierarchical representations of the functional attributes shaped-in-the-form-of general trees; (*iv*) $\mathcal{M}$ is the set of *measures* of $\mathcal{L}$, i.e. the *attributes of interest* (of S) for the underlying OLAP analysis (in other words, $\mathcal{M}$ is the set of attributes with respect to which SQL aggregations stored in data cells of $\mathcal{L}$ are computed).

In the following, we provide an analysis of issues of mobile databases and data warehouses. Even if many and well-founded issues have been recognized in the context of mobile databases and data warehouses (Barbarà, 1999), in our opinion, *frequent disconnections* that can happen when a mobile device runs in an online manner (i.e., connected to the WLAN) are among the most remarkable ones. In fact, (*i*) every today mobile device does not maintain a stable connection with the application/data server, and, due to technological drawbacks, (*ii*) wireless connection is often down. Moreover, only few citizen environments have actually an extended wireless network; among these, we recall universities, airports, big commercial centers etc. However, the actual trend is a resolute increasing diffusion, and foreseeing a large, *per-user* diffusion of mobile devices is very easy. The diffusion of wireless networks should be, as a consequence, very impressive during the next years.

In more detail, this scenario is that drawn by the modern ***ubiquitous and pervasive computing*** paradigm (Perich et al., 2004). This paradigm introduces significantly new challenges not addressed by the first mobile data management solutions, which were mainly focused on "basic" data management issues such as data representation, data storing, data transport protocols etc.

Variability of the network configuration is another issue of mobile databases and data warehouses that is directly related to the previous one. For instance, WLANs can operate in both infrastructure-based and ad-hoc modes. In the former case, wireless devices interact with an access point, which acts as bridge to a wired network. In the latter case, wireless devices interact with other neighboring devices directly, thus dynamically building the so-called *Mobile Ad-hoc NETworks* (*MANETs*) (NIST, 2005). From the data management point of view, such a variability impose us to cope with unrecognized aspects. For instance, dealing with *disconnection-aware data protocols and techniques* is a very exciting research challenge. First solutions to this issue proposed proxy-based approach (Bharadvaj et al., 1998; Joshi, 2000), extremely lightweight database clients (Bobineau et al., 2000), techniques based on partial replication of the main database on the wired side (Imielinski et al., 1997; Tait et al., 1995), techniques based on continuously broadcasting into the environment selected data (Acharya et al., 1995; Goodman et al., 1997), decentralized solutions inspired from well-known distributed

and federated databases research results (Oezsu & Valduriez, 1999) etc.

Asymmetry in the communications is another not negligible issue for mobile databases and data warehouses. Upstream communication channel has very often a much smaller bandwidth than the downstream communication channel, thus an even strong asymmetry can originate during data transport. Particularly, this asymmetry makes the achievement of effectiveness in data-intensive mobile applications very difficult, as accessing massive data sets via a wireless connection can be very problematic.

Power and screen size limitations are secondary issues that contribute to make data-intensive mobile applications minus effective. The resulting reduced usability can become a serious drawback for a wide class of modern DIS, as, typically, such applications require many user transactions before to produce the desiderata results, or before to complete knowledge extraction processes. To become convinced of this, it is suffice to think about modern *e*-government and *e*-procurement (wired) Web scenarios transposed on the *mobile Web*.

Data dissemination is a specific technique for DIS in the mobile context. It consists in broadcasting data to the clients, instead of waiting for the clients to request specific data items. The goal is to prevent requests in order to minimize *data access issues*, and optimize distribution data loads across the wireless network consequently.

Data consistency deals with guaranteeing consistency of data under (mobile) transactions that can admit disconnections. Data consistency is a very problematic issue that, however, could be addressed starting from relevant research results achieved in the last decade in the *data integration* context (e.g. (Lenzerini, 2002)).

Location-aware query answering investigates the way of answering queries on the basis of *current* positions of mobile users and devices, in order to reduce the spatio-temporal complexity of evaluating queries.

Finally, *innovative interfaces design* aims at finding new development patterns based on non-traditional interaction schemes, thus dealing with *visualization/interaction issues* between mobile users and mobile databases and data warehouses.

3. DATA-INTENSIVE MOBILE APPLICATIONS AND SYSTEMS

MoGATU (Perich et al., 2004) is a distributed framework that enables serendipitous querying and profile-driven data management in (data-intensive) pervasive environments. Such framework treats all mobile devices as equal semiautonomous peers. It can be classified as a *chained architecture* with *random replication* and *local incremental policy*. The framework abstracts mobile devices in terms of (*i*) *Information Providers*, (*ii*) *Information Consumers*, (*iii*) and *Information Managers*. Additionally, the framework employs *Communication Interfaces* for supporting multiple networking technologies and *Profiles* for enabling a proactive behavior. In MoGATU, data access and query issues are addressed by means of a *best-effort approach*, where a *data routing algorithm* allows closer devices to provide answers to queries placed by peers. To efficiently support this mechanism, MoGATU relies on *proactively cached information* that meaningfully exploit intelligent data dissemination strategies. This innovative query mechanism realizes the serendipitous feature of MoGATU, what means that data of the pervasive environment are accessed and queried without any fixed scheme or routing policy, but, indeed, based on *current* (even changing) profiles and contexts. The profile-driven data management refers instead the MoGATU's feature based on which users, devices and data are enriched by a powerful semantics-based language that allows us to model different "behaviors" and "beliefs" of these interacting entities so that the underlying

query mechanism can be driven by dynamics and contexts of the pervasive environment, with also the amenity of making use of innovative reasoning techniques founding on the concepts of presence of data and data caching.

(Gao and Hurson, 2005) proposes an interesting *location-dependent* query proxy mechanism for mobile environments where a caching scheme provides efficient processing for queries that exhibit semantic similarity and spatial locality. In more detail, this scheme is based on the *validity region* (VR) concept according to which database servers distributed over the wireless network provide to mobile devices their *validity information* conditionally, or do not provide it all. This allows us to significantly reduce query processing and data storage overheads as devices can simply query VR metadata published by database servers before to start or execute a user transaction.

Generic and Progressive Algorithms for Continuous mobile queries (GPAC) are proposed in (Mokbel & Aref, 2005). These algorithms can be applied to traditional mobile data (e.g., stored in mobile relational databases) as well as to mobile data generated by *moving objects*, whose data management is of relevant interest actually. Also, GPAC support even complex classes of queries, such as *continuous range-queries* and *k-nearest-neighbor* (*k-NN*) *queries*, which both expose a non-trivial complexity in terms of both spatial and temporal needs. The main idea of GPAC is to employ an *anticipation and incremental paradigm*, whose experimental evaluation presented in (Mokbel & Aref, 2005) shows an efficient performance gain over traditional approaches. Specifically, anticipation is obtained via caching query answers based on locality and frequency of accesses, and incremental evaluation is obtained via high-performance physical pipeline query operators.

A very interesting scheme for multidimensional data access in mobile networks is presented in (Liu et al., 2005). This scheme is based on *distributed caching*, and aims at reducing response time needed to evaluate complex multidimensional queries, such as range- and *k-NN* queries, by means of selecting the *cacheable region* for each query, and indexing the collection of derived (multidimensional) regions via high-performance *R**-trees. Experiments given in (Liu et al., 2005) confirms the effectiveness of the proposed caching policy over massive, distributed repositories of multidimensional data.

OLAP-enabling **mobile systems** are a particular class of systems inside the more general one of the data-intensive mobile systems. Specifically, such systems represent the most convenient application interfaces to mobile data warehouses (Sampaio et al., 2003; Stanoi et al., 1999), and allows us to efficiently exploit the amenity of gathering useful knowledge from distributed data sources through the abstraction of dealing with (OLAP) *dimensions* and *measures* (Gray et al., 1997). Some preliminary experiences of the so-called *mobile OLAP* can be found in (Rodriguez-Martinez & Rossopoulos, 2000; Sharaf & Chrysanthis, 2002). (Rodriguez-Martinez & Rossopoulos, 2000) describes a generic approach for developing database middleware on distributed data sources, and propose an effective realization of it, namely *MOCHA*; these guidelines for database middleware can also be efficiently adopted in mobile environments. On the other hand, (Sharaf & Chrysanthis, 2002) proposes a mobile OLAP model tailored for wireless environments, where power consumption is also taken into account, along with an on-demand scheduling algorithm that minimizes access time (thus, energy consumption) of mobile agents implementing OLAP aggregations near the (mobile) data sources.

Finally, *CubeView* (Maniatis et al., 2005) is a mobile system which allows accessing, browsing and querying wireless-accessible OLAP servers efficiently. To this end, CubeView makes use of the so-called *Cube Presentation Model* (*CPM*), a novel presentational model for "non-traditional" OLAP screens, like those of handheld devices. CPM is in turn based on the geometrical represen-

tation of a data cube and its human perception in the (multidimensional) space. *CubeView* software architecture and front-end tool are very similar to those of Hand-OLAP.

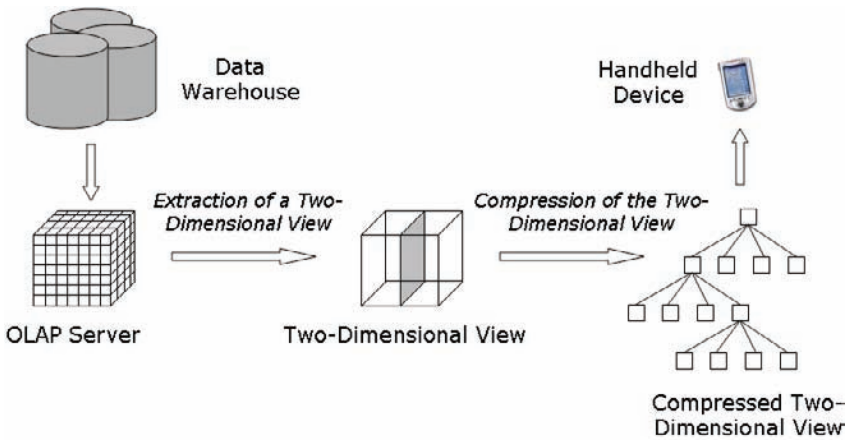
4. THE HAND-OLAP SYSTEM

In mobile OLAP systems, which can be reasonably considered as an emerging instance of mobile database and data warehouse systems, users access corporate OLAP servers via handheld devices. Similarly to the main context, mobile devices are usually characterized by specific properties (e.g., small storage space, small size of the display screen, discontinuance of the connection to the WLAN etc) that are often incompatible with the need of browsing and querying summarized information extracted from massive multidimensional data cubes made accessible through wireless networks. In such application scenarios, compressing multidimensional data cubes into two-dimensional OLAP views represents an effective solution yet an enabling technology for mobile OLAP environments, as, contrarily to what happens for hyper-spaces, handheld devices can easily visualize two-dimensional spaces on conventional (e.g., 2D) screens. This property, along with the realistic need of compressing data to be transmitted and processed by handheld devices, makes perfect sense to our idea of using data compression techniques as a way of efficiently accessing and querying massive data sets.

Inspired by the considerations above, Hand-OLAP (Cuzzocrea et al., 2003) allows OLAP users to extract, browse and query compressed two-dimensional views (which are computed via the technique (Buccafurri et al., 2003)) coming from a remote OLAP server (see Figure 1). Specifically, according to the guidelines of algorithms proposed in (Buccafurri et al., 2003), Hand-OLAP is targeted at supporting *range-queries* (Ho et al., 1997), a very popular class of queries useful to extract summarized knowledge from data cubes in the vest of aggregate information. The basic idea which Hand-OLAP is based on is: rather than querying the original multidimensional data, it may be more convenient to generate a compressed view of them, store the view into the handheld device, and query it locally, even if the WLAN is off, thus obtaining approximate answers that are perfectly suitable for OLAP goals (e.g., see (Cuzzocrea, 2005a)).

According to well-known design patterns, Hand-OLAP is a multi-tier system, and every software layer corresponds to a specific applica-

Figure 1. Hand-OLAP overview



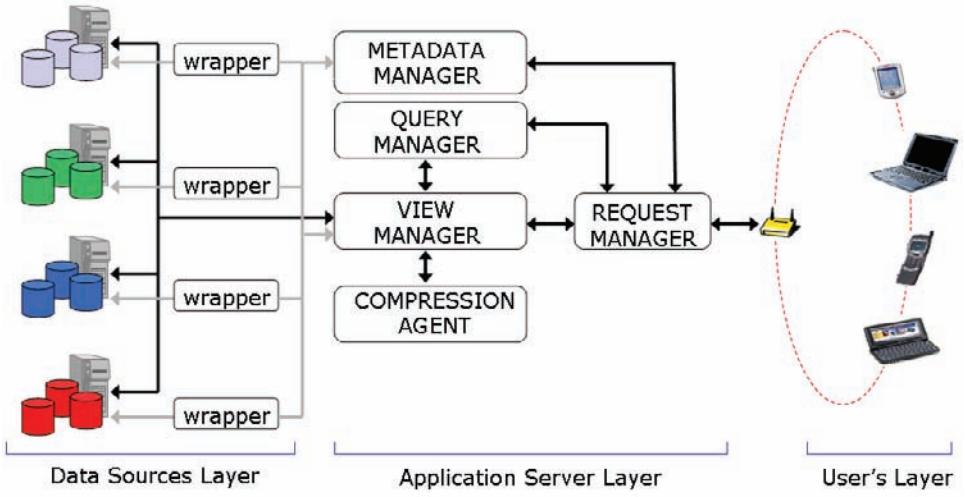
tion logic (see Figure 2). Specifically, the following software layers can be identified in the Hand-OLAP logical architecture:

- ***Data Sources Layer*:** it is the collection of (*i*) OLAP servers from which the desired information can be retrieved, and (*ii*) wrappers that extract meta-information about the available data cubes as well as the actual data;
- ***Application Server Layer*:** it is the layer that (*i*) elaborates OLAP-users' requests, (*ii*) interacts with OLAP servers, (*iii*) computes the compressed representation of the extracted OLAP view, and (*iv*) sends it to the handheld device;
- ***User's Layer*:** it includes the client-side tool that allows a handheld device to acquire and elaborate the desired information, by enabling useful functionalities such as connectivity services, metadata querying and browsing, range-query managing (e.g., editing, executing, browsing, refreshing etc).

Application Server Layer, which is the most interesting component of Hand-OLAP with respect to the Data Engineering point of view, consists of three components which cooperate to fulfill OLAP-users' requests:

- ***Request Manager*:** it is the component that receives the request of OLAP users, and translates it either into a request to the *Metadata Manager* for retrieving meta-information about the content of the target data cube, or into a request to the *View Manager* for retrieving a compressed representation of the two-dimensional OLAP view defined by OLAP users;
- ***Metadata Manager*:** it is the component that extracts meta-information about the OLAP server it is connected to, and returns them in a XML format;
- ***View Manager*:** it is the component that (*i*) extracts from the selected data cube the two-dimensional view defined by OLAP users, (*ii*) uses the *Compression Agent* for summarizing it, and (*iii*) returns the compressed representation to the handheld device;
- ***Compression Agent*:** it is the component that receives a two-dimensional view from the *View Manager*, and returns its compressed representation to it – in particular, the *View Manager* sends the extracted

Figure 2. Hand-OLAP logical architecture



two-dimensional view to the *Compression Agent* together with the value of the desired *compression ratio*, which depends on both the amount of storage space available at the handheld device and the size of the view;

- ***Query Manager*:** it is the component that is in charge of supporting range-query evaluation on the compressed two-dimensional view, and visualizing the results on the handheld device according to a partitioned hierarchical representation.

User's Layer consists of the Hand-OLAP client-side tool (see Figure 3), which OLAP users with instruments for:

- Scanning the wired network and selecting an OLAP server to extract information from;
- Asking for XML metadata about the information contained in the accessible data sources, and browsing them;
- Defining a two-dimensional view over the selected data cube, containing *interesting* information;
- Downloading the compressed representation of the selected view, after negotiating the compression ratio;
- Refreshing an already downloaded compressed view;
- Executing range queries on the compressed representation, without accessing to the original data.

5. CONCLUSION

Starting from similarities and differences between wired and wireless database and data warehouse access/query solutions, in this article we have proposed a critical discussion on several aspects of mobile databases and data warehouses, along with a survey on state-of-the-art data-intensive mobile applications and systems. This survey has been completed with the description of the Hand-OLAP system, a relevant instance of mobile OLAP systems, which can in turn be reasonably considered as a significant case of mobile data warehouse systems.

As highlighted throughout the paper, access and query functionalities represent critical bottlenecks for data-intensive mobile applications and systems, since these functionalities are in turn exploited by intelligent information processing techniques in order to provide support for even

Figure 3. Two screenshots of the Hand-OLAP client-side tool

complex knowledge extraction activities against mobile databases and data warehouses. From this evidence, it is a matter to note that the issue of efficiently accessing and querying mobile data will rapidly conquest the research scene during next years. In this respect, topics presented and discussed in this article can be reasonable considered as a significant contribution to this line of research, as well as a starting point for further research in this field.

6. FUTURE TRENDS

A lot of research needs to be developed in the context of mobile data management research, and particularly for what regards to data access and query aspects. In fact, specific characteristics of mobile data management make data processing technology developed in the context of wired environments inappropriate in order to efficiently support access and query functionalities over mobile databases and data warehouses. Among the widely-recognized directions of research in this filed, here we want to highlight the following ones: (*i*) devising efficient *indexing solutions* for mobile data able to deal with disconnections that can happen in the underlying WLAN; (*ii*) devising efficient *data reliability protocols* able to provide data consistency across mobile databases and data warehouses; (*iii*) devising innovative *query paradigms* that embed probabilistic schemes able to deal with incomplete/imprecise information; (*iv*) providing support for *privacy-preserving access schemes* over mobile databases and data warehouses; (*v*) designing *novel architectures* for the integration of mobile databases and data warehouses and traditional DBMSs.

REFERENCES

Acharya, S., Alonso, R., Franklin, M., & Zdonik, S. (1995). Broadcast Disks: Data Management for Asymmetric Communication Environments. *Proc. of the 1995 ACM Int. Conf. on Management of Data,* (pp. 199-210).

Acharya, S., Gibbons, P. B., Poosala, V., & Ramaswamy, S. (1999b). Join Synopses for Approximate Query Answering. *Proc of the 1999 ACM Int. Conf. on Management of Data,* (pp. 275-286).

Barbarà, D. (1999). Mobile Computing and Databases – A Survey. *IEEE Transactions on Knowledge and Data Engineering,* IEEE, 11(1), 108-117.

Bharadvaj, H., Joshi, A., & Auephanwiriyakyl, S. (1998). An Active Transcoding Proxy to Support Mobile Web Access. *Proc. of the 17th IEEE Symp. on Reliable Distributed Systems,* (pp. 118-123).

Bobineau, C., Bouganim, L., Pucheral, P., & Valduriez, P. (2000). PicoDBMS: Scaling Down Database Techniques for the Smartcard. *Proc. of the 26th Int. Conf. on Very Large Data Bases,* (pp. 11-20).

Buccafurri, F., Furfaro, F., Saccà, D., & Sirangelo, C. (2003). A Quad-Tree Based Multiresolution Approach for Two-Dimensional Summary Data. *Proceedings of the 15th IEEE Int. Conf. on Scientific and Statistical Database Management,* (pp. 127-140).

Buyukkokten, O., Garcia-Molina, H., & Paepcke, A. (2001). Seeing the Whole in Parts: text Summarization for Web Browsing on Handled Devices. *Proc. of the 10th Int. World Wide Web Conf.,* (pp. 652-662).

Chaudhuri, S., Das, G., Datar, M., Motwani, R., & Rastogi, R. (2001). Overcoming Limitations of Sampling for Aggregation Queries. *Proc. of the 17th IEEE Int. Conf. on Data Engineering,* (pp. 534-542).

Cuzzocrea, A. (2005a). Overcoming Limitations of Approximate Query Answering in OLAP. *Proc. of the 9th IEEE Int. Conf. on Database Engineering and Applications,* (pp. 200-209).

Cuzzocrea, A. (2005b). Providing Probabilistically-Bounded Approximate Answers to Non-Holistic Aggregate Range Queries in OLAP. *Proc. of the 8th ACM Int. Work. on Data Warehousing and OLAP*, (pp. 97-106).

Cuzzocrea, A. (2006). Improving Range-Sum Query Evaluation on Data Cubes via Polynomial Approximation. *Data & Knowledge Engineering*, Elsevier Science, 56(2), 85-121.

Cuzzocrea, A., Furfaro, F., & Saccà, D. (2003). Hand-OLAP: A System for Delivering OLAP Services on Handheld Devices. *Proc. of the 6th IEEE Int. Conf. on Autonomous Decentralized Systems*, (pp. 80-87).

Cuzzocrea, A., & Wang, W. (2007). Approximate Range-Sum Query Answering on Data Cubes with Probabilistic Guarantees. *Journal of Intelligent Information Systems*, Springer Science, Springer Science, 28(2), 161-197.

Franz, M., & Kistler, T.(1997). Slim Binaries. *Communications of the ACM*, ACM, 40(12), 87-94.

Gao, X., & Hurson, A.R. (2005). Location Dependent Query Proxy. *Proc. of the 20th ACM Int. Symp. on Applied Computing*, (pp. 1120-1124).

Garofalakis, M.N., & Kumar, A. (2004). Deterministic Wavelet Thresholding for Maximum-Error Metrics. *Proc. of the 23rd ACM Int. Symp. on Principles of Database Systems*, (pp. 166-176).

Gibbons, P.B., & Matias, Y. (1998). New Sampling-Based Summary Statistics for Improving Approximate Query Answers. *Proc. of the 1998 ACM Int. Conf. on Management of Data*, (pp.331-342).

Goodman, D., Borras, J., Mandayam, N., & Yates, R. (1997). INFOSTATIONS: A New System Model for Data and Messaging Services. *Proc. of the IEEE Vehicular Technology Int. Conf.*, 2, 969-973.

Gray, J., Bosworth, A., Layman, A., & Pirahesh, H. (1997). Data Cube: A Relational Aggregation Operator Generalizing Group-By, Cross-Tab and Sub-Totals. *Proc. of the 12nd IEEE Int. Conf. on Data Engineering*, (pp. 152-159).

Gunopulos, D., Kollios, G., Tsotras, V.J., & Domeniconi, C. (2000). Approximating Multi-Dimensional Aggregate Range Queries over Real Attributes. *Proc. of the 2000 ACM Int. Conf. on Management of Data*, (pp. 463-474).

Ho, C.-T., Agrawal, R., Megiddo, N., & Srikant, R. (1997). Range Queries in OLAP Data Cubes. *Proc. of the 1997 ACM Int. Conf. on Management of Data*, (pp. 73-88).

Imielinski, T., Viswanathan, S., & Badrinath, B.R. (1997). Data on Air: Organization and Access. *IEEE Transactions on Knowledge and Data Engineering*, IEEE, 9(3), 352-372.

Ioannidis, Y. E., & Poosala, V. (1999). Histogram-based Approximation of Set-Valued Query Answers. *Proc. of the 25th Int. Conf. on Very Large Data Bases*, (pp. 174-185).

Joshi, A. (2000). On Proxy Agents, Mobility and Web Access. *ACM/Baltzer Journal on Mobile Networks and Applications*, Springer Science, 5(4), 233-241.

Lenzerini, M. (2002). Data Integration: A Theoretical Perspective. *Proc. of the 21th ACM Symp. on Principles of Database Systems*, (pp. 233-246).

Liu, B., Lee, W.-C., & Lee, D.L., (2005). Distributed Caching of Multi-Dimensional Data in Mobile Environments. *Proc. of the 6th ACM Int. Conf. on Mobile Data Management*, (pp. 229-233).

Maniatis, A., Vassiliadis, P., Skiadopoulos, S., Vassiliou, Y., Mavrogonatos, G., & Michalarias, I. (2005). A Presentation Model & Non-Traditional Visualization for OLAP. *International Journal of Data Warehousing and Mining*, 1(1), 1-36. Idea Group.

Mokbel, M., & Aref, W., (2005). GPAC: Generic and Progressive Processing of Mobile Queries over Mobile Data. *Proc. of the 6th ACM Int. Conf. on Mobile Data Management,* (pp. 155-163).

NIST Wireless Communication Technologies Group (2005). *Wireless Ad Hoc Networks Bibliography,* http://w3.antd.nist.gov/wctg/manet/manet_bibliog.html

Oezsu, M. T., & Valduriez, P. (1999). *Principles of Distributed Database Systems*, Prentice Hall.

Poosala, V., & Ganti, V. (1999). Fast Approximate Answers to Aggregate Queries on a Data Cube. *Proc. of the 11st IEEE Int. Conf. on Statistical and Scientific Database Management,* (pp. 24-33).

Poosala, V., & Ioannidis, Y.E. (1997). Selectivity Estimation without the Attribute Value Independence Assumption. *Proc. of the 23rd Int. Conf. on Very Large Databases,* (pp. 486-495).

Perich, F., Joshi, A., Finin, T., & Yesha, Y. (2004). On Data Management in Pervasive Computing Environments. *IEEE Transactions on Knowledge and Data Engineering,* IEEE, 16(5), 621-634.

Rodriguez-Martinez, M., & Rossopoulos, N. (2000). MOCHA: A Self-Extensible Database Middleware System for Distributed Data Sources. *Proc. of the 2000 ACM Int. Conf. on Management of Data,* (pp. 213-224).

Sampaio, M. C., Dias, P. M., & Baptista, C. S. (2003). Incremental Updates on Mobile Data Warehousing Using Optimized Hierarchical Views and New Aggregation Operators. *Proc. of 17th Int. Conf. on Advanced Information Networking and Applications,* (pp. 78-83).

Sharaf, M. A., & Chrysanthis, P. K. (2002). Semantic-based Delivery of OLAP Summary Tables in Wireless Environments. *Proc. of the 2002 ACM Int. Conf. on Information and Knowledge Management,* (pp. 84-92).

Stanoi, I., Agrawal, D., El Abbadi, A., Phatak, S. H., & Badrinath, B. R. (1999). Data Warehousing Alternatives for Mobile Environments. *Proc. of the ACM Int. Work. on Data Engineering for Wireless and Mobile Access,* (pp. 110-115).

Tait, C., Lei, H., Acharya, S., & Chang, H. (1995). Intelligent File Hoarding for Mobile Computers. *Proc. of the 1st ACM Int. Conf. on Mobile Computing and Networking,* (pp. 119-125).

Vitter, J.S., Wang, M., & Iyer, B. (1998). Data Cube Approximation and Histograms via Wavelets. *Proc. of the 7th ACM Int. Conf. on Information and Knowledge Management,* (pp. 96-104).

KEY TERMS

Data Compression: A collection of techniques aiming at reducing the size of massive data repositories (such as databases and data warehouses) and structures (such as data cubes) in order to mitigate access, process and query costs.

Data Reliability: A collection of techniques aiming at ensuring the consistency of data in distributed settings, such as mobile environments. In this specific case, data inconsistence is due to fault or unavailability of wireless protocols and devices.

Location-Aware Services: Services whose providing depends on local characteristics such as geographical position and time.

Mobile Applications and Systems: Applications and systems incorporating in their scope the wireless environment, in combination or not with the traditional wired environment.

Network Protocol: A set of rules establishing how data must be handled and transmitted throughout the network.

On-Line Analytical Processing (OLAP): A methodology for representing, managing and querying massive DW data according to multi-dimensional and multi-resolution abstractions of them.

Wireless Middleware: A collection of software components running in a wireless environment and realizing a distributed system among wireless devices. It implements a specific task or service, such as query services.

Chapter XCII
Full-Text Manipulation in Databases

László Kovács
University of Miskolc, Hungary

Domonkos Tikk
Budapest University of Technology and Economics, Hungary

INTRODUCTION

The textual data format is one of the most important data types in database management. Databases support a wide range of special textual types that can be used to store string data. In the case of textual data, information retrieval mostly concerns the selection and the ranking of documents. In the traditional database management systems (DBMS), text manipulation is related to the usual string manipulation facilities, i.e. the exact matching of substrings. The main disadvantages of the traditional string-level operations are the high cost as they work without task-oriented index structures and the restriction to the syntax level.

The required appropriate full-text management operations belong to text mining, an interdisciplinary field of natural language processing and data mining. As the traditional DBMS engine is inefficient for these operations, database management systems are usually extended with a special full-text search (FTS) engine module. Full-text search engines provide a set of full text manipulation primitives that are based on the semantic aspects of the words and sentences. In

the recent years, the area of semantic query on full-text is treated as a special area of universal ontology. The main goal of ontology is to define a common set of concepts and relationships for knowledge representation.

The market of FTS engines is very promising because the amount of textual information stored in databases rises steadily. According to the study of Meryll Lynch (Blumberg & Arte, 2003), 85% of business information are text documents – e-mails, business and research reports, memos, presentations, advertisements, news, etc. – and their proportion still increases. In 2006, there were more than 20 billion documents available on the Internet (Chang, 2006). The estimated size of the pool reaches 550 billion documents when the documents of the hidden (or deep) web are also considered.

In a broader sense, the area of full-text manipulation is not restricted only to the management of database content. The query interface may be extended with a natural language interface for databases (NLIDB). The NLIDB means that a user can use natural languages to create query expressions, and also the answer can be presented in the same languages. The processing of the incoming queries includes a full-text analysis both at syntactic and semantic levels.

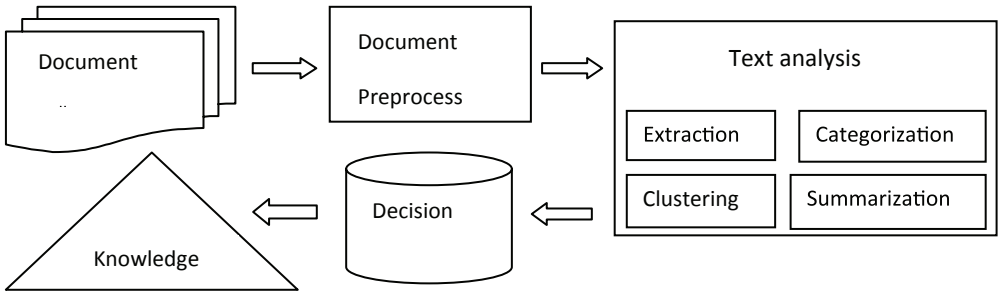
BACKGROUND

The subfield of document management that aims at processing, searching, and analyzing text documents is *text mining.* Text mining is an application oriented interdisciplinary field of machine learning which exploits tools and resources from computational linguistics, natural language processing, information retrieval, and data mining. The general application schema of text mining is depicted in Figure 1 (Fan, Wallace, Rich & Zhang, 2006). For giving a brief summary of text mining, four main areas are presented here: information extraction, text categorization/classification, document clustering, and summarization.

The goal of information extraction (IE) is to collect the text fragments (facts, places, people, etc.) from documents relevant to the given application. The IE module includes among others the following subtasks: named entity recognition (Sibanda & Uzuner, 2006); co-reference resolution (Ponzetto & Strube, 2006); identification of roles and their relations (Ruppenhofer et al, 2006).

Text categorization (TC) techniques aim at sorting documents into a given category system (see Sebastiani, 2002 for a good survey). Typical application examples of TC include among many others: document filtering (Lewis, 1995); patent

Figure 1. The text mining module



document routing (Larkey, 1999); assisted categorization (Tikk et al, 2007), automatic metadata generation (Liddy et al, 2002),

Document clustering (DC) methods group elements of a document collection based on their similarity. Documents are usually clustered based on their content. DC is applied for e.g.: clustering the results of (internet) search for helping users in locating information (Zamir et al, 1997), improving the speed of vector space based information retrieval (Manning et al, 2007) ; providing a navigation tool when browsing a document collection (Käki, 2005).

Text summarization aims at the automatic generation of short and comprehensible summaries of documents. The typical application areas of summarization span from the internet search to arbitrary document management system (Ganapathiraju, 2002).

The core problem in semantic level full-text management is the efficient implementation of concept-based operators. The area of concept management belongs to the umbrella-term ontology. The term ontology can be given as explicit "specification of conceptualization" (Gruber,1993). The ontology is used to define the concepts, relationships and other distinctions that are relevant for modeling a domain where the specification takes the form of the definitions of representational vocabulary which provides meaning and constraints on the usage. The key element of the ontology is a methodology that determines what the primitives of the conceptualization are and how to determine these primitives from the domain.

In the literature, there exists a large variety of description languages for ontology. One of the first languages for ontology is RDF (Lassila, 1999). RDF describes the concepts in a neutral, machine-readable format. The basic language elements of RDF are resources, literals and statements. There are two types of resources: entity resources and properties. A statement is a triplet (*p*,*s*,*o*), where *p* is a property, *s* is a resource and *o* is either a literal or a resource. One of the most widely used formalism is the OWL language that is a part of the W3C semantic standards. The OWL includes several parts and one of the variants is based on the descriptive logic (DL) in order to provide a formal way of constraining and reasoning (Borgida, 1994). This representation form supports a precise inference framework for reasoning purposes. Hayes (1979) has proven first the equivalence of the framework representation form and the first order logic formalism. Based on the research works of Brachman(1984), the reasoning in the frame and network representations can be accomplished without the full power of the first order logic.

FULL-TEXT OPERATORS

Having a set of full-text items, two groups of full-text operations can be defined on this set. The first cluster is related to the matching based on character level, where no meaning is associated to the strings. The second group relates to the semantic level where the goal is to mine the meaning of the full text.

Full-Text Character-Level Operators

Based on the literature (Maier, 2001; Curtmola, 2005), an effective FTS engine should support several querying functionalities. The simplest one is the string-based querying, which retrieves texts that exactly match the query string. In order to refine the matching, there are operators which consider only some specific parts of the words. In some cases, the position of the keywords within the document is also an important factor. In the case of approximate matching the best fitting dictionary elements are selected for a query string. The distance is based usually on the edit (Levenshtein) distance. The calculation of the distance uses a dynamic programming approach with an O(nm) cost factor. Due to the heterogeneity of the source pool, obviously dif-

ferent document formats have to be supported. The minimal usage of other resources provides an independent, flexible solution. From the aspect of software development, the open, standardized interface is a good investment. To provide a manageable, easy to understand response, the efficient ranking of the elements of the result set is crucial (Chakrabarti, 2006). The products and test systems currently available only partially meet the above requirements. The most widely used character-level operators in full-text search are the followings:

- exact word matching,
- approximate matching (fuzzy matching),
- sequence matching,
- distance based matching (related to the position within the text),
- frequency weighted matching.

Full Text Semantic Operators

An FTS engine should also support grammar (and therefore language) specific operators (e.g. stemming). The highest level of text search operates with semantic-based matching (thesaurus-based neighborhood, generalization of a word, specialization, synonyms). The grammatical analysis of full text belongs to the area of *computational linguistics*. There are different approaches to describe and to model the grammar of a natural language. The generative grammar (Chomsky, 1965) aims at defining and modeling linguistic objects and processes them with mathematical methods. Categorical grammar is a term used for a family of formalism motivated by the principle of compositionality, i.e. the meaning of a complex expression is determined by the meanings of its constituent expressions and the rules used to combine them. Construction grammar (Fillmore, 1980) covers models that are based on the idea that the primary unit of a grammar is the grammatical construction rather the atomic syntactic unit. Dependency grammar is a class of syntactic theories where a structure is defined by the relation between a word and its dependents. This grammar is well suited to languages with free word order. One of the most widely used grammars in computational linguistics is the tree adjoining grammar (TAG) (Joshi ; Schabes, 1992) uses a formalism similar to the context-free grammars but the elementary unit of rewriting is the tree rather than the symbols.

The most basic form of ontology-driven operators in DBMS is the application of thesaurus. The thesaurus implementation in (Oracle, 2007) supports the following relationships:

- specialization,
- generalization,
- association,
- synonyms,
- translation,
- stem-based matching.

The related SQL operators for the new matching mechanism are based on the ontology-level distance interpretation. The core of the ontology is a taxonomy containing the specialization relationship of concepts. The ONT_RELATED operator can be used to calculate distance value related to the underlying ontology model. The sample query below shows the syntax of the semantic matching operator:

SELECT * FROM employees WHERE ONT_RELATED (job,'IS_A','clerk','Ontology') = 1

FTS engines are structurally similar to database systems: they store data and metadata; their purpose is to provide an efficient information retrieval As the processing of the full-text querying requires several distinct steps, the FTS engines typically have modular structure (Microsoft, 2007; Oracle, 2007).

The *loader* module loads the input documents into a common staging area, and converts text into a common representation form. The input documents are stored in the *datastore unit*. The

sectioner unit has to discover the larger internal logical structure of the documents. The *word-breaker* parses the text into smaller syntactical units like paragraphs, sentences and words. For reducing the length and complexity of the text, several preprocessing steps are executed. First, a *filter module* is applied that discard irrelevant words (stop-words, noise words). Next, the *stemmer unit* generates the stem form for every word. In the background, the *language lexicon* supports the language-specific steps. This lexicon contains the grammar of the supported languages and the list of stop-words. The thesaurus is a special lexicon, which stores the terms organized in a graph based on their semantic relationship. The *indexer unit* manages the different document-word indices that enable the efficient access to posting list of words. On the front-end side, the *query preprocessor* transforms the user's query into an internal format. This format is processed by the *query matcher*, resulting in a set of matching documents. In order to provide a more accurate response, the *query refinement engine* performs the processing of the relevance feedback. The set of matching documents is pipelined to the *ranking module*. The *exporter module* generates the final format of the ranked document set.

FUTURE TRENDS

In our view, there are three main areas where the role of FTS engine should be improved in the future: web search engines, ontology-based information retrieval, and management of XML documents. A very important application area of full-text search is the web. A special feature of web search is that the users post mostly simple queries. Only 10% of queries use some complex full-text operators such as Boolean operators, stemming or fuzzy matching. Eastman (2003) investigated why complex full-text operators are not popular, and concluded that the application of those does not improve significantly the search results. Several research projects focus on the efficient implementation of full-text operators with improved indexing mechanism (Silvestri, 2004).

The efficiency of information retrieval can be improved with the extension of additional semantic information. The ALVIS project (Luu, 2006) aims at building a distributed, peer-to-peer semantic search engine. The peer-to-peer network is a self-organizing system for decentralized data management in distributed environments. During a query operation, a peer broadcasts search requests in the network. A peer may be assigned to a subset of data items. The key element in the cost reduction is the application of a special index type at the nodes.

The main standard for the query of XML documents is nowadays the XQuery language. This standard is very flexible for selecting structured data elements, but it has no special features for the unstructured part. In (Botev, 2004; Curtmola, 2005), an extension of XQuery with full-text functionality is proposed. The extended query language is called TeXQuery and GalaTex. The language contains a rich set of composite full-text primitives such as phrase matching, proximity distance, stemming and thesauri. The combination of structure- and content-based queries is investigated deeply from a theoretical viewpoint in (Amer, 2004).

Nowadays the automatic processing of information can be strengthened by involving ontology. The ontology can be considered as a tool to describe conceptualization of the problem domain. In addition to the formal description of the concepts, the ontology contains the relationships between the concepts, and the logical rules to perform consistency checking and knowledge inference. The involving of ontology into DB querying started in the late 1990's. The QUEST project developed a test system where the users could use an ontology-based category system to enter the query statement. In a more recent project (Bernstein, 2005), an NL interface is implemented

where the interpretation of the commands are performed with the help of ontology descriptions. The project applies simplified English, the ACE (Attempto Controlled English) language. Another ontology driven question answering system is the AQUA (Vargas-Vera, 2004) project where the QLL logical language is applied as the intermediate language.

CONCLUSION

The information is stored on the web and on computers mostly in full-text format. The current databases are able to store and manage huge document collection. Full-text data sources require specific search operations. Database management systems usually contain a separate full-text search engine to perform full-text search primitives. In general, the current FTS engines support the following functionalities: stemming, synonym and thesaurus based matching, fuzzy matching and Boolean operators. It has been shown that the average user requires additional help to exploit the benefits of these extra operators. Current research focuses on solving the problem of covering new document formats, adapting the query to the user's behavior, and providing an efficient FTS engine implementation.

REFERENCES

Amer Yahia, S., Lakshmanan, L., & Pandit, S. (2004). FlexPath: Flexible Structure and Full-Text Queryng for XML. In *Proc. of ACM SIGMOD* (pp. 83–94), Paris, France.

Blumberg, R., & Arte, S. (2003). The problem with unstructured data. *DM Review* (February).

Borgida, A. (1994). On the relationship between description logic and predicate logic queries. *Proc of Conference on Information and Knowledge Management*, Maryland, USA.

Botev, C., Amer-Yaiha, S., & Shanmugasundaram, J. (2004). A TexQuery-based XML full-text search engine. In *Proc. of ACM SIGMOD* (pp. 943–944), Paris, France.

Chakrabarti, K., Ganti, V., Han, J., & Xin, D. (2006). Ranking Objects by Exploiting Relationships: Computing Top-K-over Aggregation. In *Proc. of ACM SIGMOD* (pp. 371–382), Chicago, IL, USA.

Chang, K., & Cho, J. (2006). Accessing the Web: From Search to Integration. In *Proc. of ACM SIGMOD* (pp. 804–805) , Chicago, IL, USA.

Chomsky, N. (1965). Aspects of the Theory of Syntax. Cambridge, MA: MIT Press.

Codd, E. F (1985). Is Your DBMS Really Rational, (Codd's 12 rules). *Computerworld Magazine.*

Curtmola, E., Amer-Yaiha, S., Brown, P., & Fernandez, M. (2005). GalaTex: A Conformant Implementation of the Xquery Full-Text Language. *Proc. of WWW 2005.* (pp. 1024–1025), Chiba, Japan.

Eastman, C., & Jansen, B (2003). Coverage, Relevance, and Ranking: The Impact of Query Operators on Web Search Engine Results. *ACM Transactions on Information Systems, 21*(4), 383–411.

Fan, W., Wallace, L., Rich, S., & Zhang, Z. (2006). Tapping the power of text mining. *Communications of the ACM, 49*(9), 76–82.

Fillmore, C. (1968). The Case for Case, In Bach and Harms (Ed.), *Universals in Linguistic Theory.* New York: Holt, Rinehart, and Winston, (pp. 1-88).

Gruber, T. (1993). Towards principles for the design of ontologies used for knowledge sharing. *Proc of International Workshop on Formal Ontology*, Padova Italy.

Ganapathiraju, M. K. (2002). *Relevance of cluster size in MMR based summarizer.* Technical

Report 11-742, Language Technologies Institute, Carnegie Mellon University, Pittsburgh.

Joshi, A. K., & Schabes, Y. (1997). Tree-Adjoining Grammars. In G. Rozenberg & A. Salomaa (Eds.), *Handbook of Formal Languages, 3,* 69–124. Springer, Berlin, New York, 1997.

Käki, M. (2005). Findex: search result categories help users when document ranking fails. In *CHI -05: Proc. of the SIGCHI conference on Human factors in computing systems* (pp. 131–140), Portland, OR, USA.

Lassila, O. (1999). Resource Description Framework. *Proc in XML'99 Conference*, Philadelphia.

Larkey, L. S. (1999). A patent search and classification system. In *Proc. of DL-99, 4th ACM Conference on Digital Libraries* (pp. 179–187), Berkeley, CA, USA.

Lewis, D. D. (1995). The TREC-4 filtering track: description and analysis. In *Proc. of TREC-4, 4th Text Retrieval Conference,* (pp. 165–180), Gaithersburg, MD, USA.

Liddy, E. D., Sutton, S., Allen, E., Harwell, S., Corieri, S., Yilmazel, O., Ozgencil, N. E., Diekema, A., McCracken, N., & Silverstein, J. (2002). Automatic metadata generation and evaluation. In *Proc. of ACM SIGIR* (pp. 401–402), Tampere, Finland.

Luu, T., Klemm, F., Podnar, I., Rajman, M., & Aberer, K. (2006). ALVIS Peers: A Scalable Full-text Peer-to-Peer Retrieval Engine. *Proc. of ACM P2PIR'06* (pp. 41–48), Arlington, VA, USA.

Maier, A., & Simmen, D. (2001). *DB2 Optimization in Support of Full Text Search.* Bulletin of IEEE on Data Engineering.

Manning, Ch. D., Raghavan, P., & Schütze, H. (2007). *Introduction to Information Retrieval.* Cambridge University Press.

Microsoft (2007). SQL Server Full Text Search Engine, http://technet.microsoft.com/en-us/library/ms345119.aspx

Oracle Text (2007). *Oracle Text Product Description,* homepage: http://www.oracle.com/technology/products/text/index.html

Ponzetto, S. P., & Strube, M. (2006). Exploiting semantic role labeling, WordNet and Wikipedia for coreference resolution. In *Proc. of HLT-NAACL, Human Language Technology Conf. of the NAACL* (pp. 192–199), New York, USA.

Ruppenhofer, J., Ellsworth, M., Petruck, M. R. L., Johnson, Ch. R., & Scheffczyk, J. (2006). *FrameNet II: Extended Theory and Practice.* International Computer Science Institute, Berkeley, USA.

Sebastiani, F. (2002). Machine learning in automated text categorization. *ACM Computing Surveys, 34*(1), 1–47.

Sibanda, T., & Uzuner, Ö. (2006). Role of local context in automatic deidentification of ungrammatical, fragmented text. In *Proc. of HLT-NAACL, Human Language Technology Conf. of the NAACL* (pp. 65–73), New York, USA.

Silvestri, F., Orlando, S., & Perego, R. (2004). WINGS: A Parallel Indexer for Web Contents, *Lecture Notes in Computer Science, 3036,* 263-270.

Tikk, D., Biró, Gy., & Törcsvári, A. (2007). A hierarchical online classifier for patent categorization. In H. A. do Prado & E. Ferneda (Eds.), *Emerging Technologies of Text Mining: Techniques and Applications.* Idea Group Inc. (in press).

Zamir, O., Etzioni, O., Madani, O., & Karp, R. M. (1997). Fast and intuitive clustering of web documents. In *Proc. of SIGKDD-97, 3rd Int. Conf. on Knowledge Discovery and Data Mining* (pp. 287–290), Newport Beach, USA.

Zobel, J., & Moffat, A. (2006). Inverted Files for Text Search Engines. *ACM Computing Surveys*, *38*(2), Article 6.

KEY TERMS

Computational Linguistics: Application of computer for modeling and investigation of natural languages.

Full-Text: Text data without any predefined structure. It contains usually sentences given in some natural language.

Full-Text Search (FTS) Engine: A module within a database management system that supports efficient search in free texts. The main operations supported by the FTS engine are the exact matching, position-based matching, similarity-based matching, grammar-based matching and semantic-based matching.

Fuzzy Matching: A special type of word matching where the similarity of two terms are calculated as the cost of the transformation from one into the other. The most widely used cost calculation method is the edit distance method.

Indexer: It builds one or more indices for the speed up information retrieval from free text. These indices usually contain the following information: terms (words), occurrence of the terms, format attributes.

Ontology: Explicit specification of conceptualization. The ontology is used to define the concepts, relationships and other distinctions that are relevant for modeling a domain.

Semantic operators: operators that are based on the semantic content of the text. Specialization and synonyms are base examples of semantic operators.

Stemmer: It is a language-dependent module that determines the stem form of a given word. The stem form is usually identical to the morphological root. It requires a language dictionary.

Thesaurus: A special repository of terms, which contains not only the words themselves but the similarity, the generalization and specialization relationships. It describes the context of a word but it does not give an explicit definition for the word.

Chapter XCIII
Bind but Dynamic Technique:
The Ultimate Protection Against SQL Injections

Ahmad Hammoud
Lebanese American University, Lebanon

Ramzi A. Haraty
Lebanese American University, Lebanon

INTRODUCTION

Most Web developers underestimate the risk and the level of damage that might be caused when Web applications are vulnerable to SQL (structured query language) injections. Unfortunately, Web applications with such vulnerability constitute a large part of today's Web application landscape. This article aims at highlighting the risk of SQL injection attacks and provides an efficient solution.

BACKGROUND

Attackers usually make use of SQL injection attacks in order to compromise both the confidentiality and integrity of RDBMS- (relational database management system) powered Web applications. In some cases, even their availability is compromised (Cerrudo, 2002).

In his "Introduction to SQL Injection Attacks for Oracle Developers," Stephen Kost (2004) says,

application audits have found many web applications vulnerable to SQL injection even though well established coding standards were in place during development of many of these applications. Function-based SQL injection attacks are of most concern since these attacks do not require knowledge of the application and can be easily automated.

Fortunately, developers can use simple and easy-to-implement techniques to defend against SQL injection attacks. There is no need for a special tool or to introduce dedicated hardware; simple coding practices can do the job.

MAIN THRUST: SQL INJECTIONS

To tackle the problem immediately, an example will be given so that the concept of SQL injection is clear before solutions are explored. Consider the following code.

```
SQL = "Select * from UsersTbl WHERE Usr ="
SQL = SQL + SuppliedUsr
SQL = SQL + "And Pwd = "
SQL = SQL + SuppliedPwd
Execute SQL
```

The above code is a typical SQL statement that will be executed whenever a user is trying to log in. This is the code that exists behind the log-in button on the log-in page. Obviously, the SQL statement attempts to find a record in the table called UsersTbl so that the two fields *user* and *password* are equal to the ones supplied by the user. If the *Execute* statement returned rows, then this means the supplied user name and password are correct and the user will be allowed to proceed because he or she is authenticated. The following two scenarios may occur.

Normal User Scenario

Suppose the supplied user name and password is as follows.

- SuppliedUsr : 123
- SuppliedPwd : 456

Then, the SQL statement that will be executed is as follows:

```
Select * from UsersTbl WHERE Usr = 123 And Pwd = 456
```

If there is such a record in the UsersTbl table, then the EXECUTE statement returns a record and, as a result, the user will be able to proceed. If no such record exists, the EXECUTE statement returns zero records. Consequently, the user will be asked to try again.

Hacker Scenario

Suppose the supplied user name and passwords are as follows.

- SuppliedUsr : 123 --
- SuppliedPwd : whatever

Then, the SQL statement that will be executed follows.

```
Select * from UsersTbl WHERE Usr = 123 -- And Pwd = whatever
```

That way, a hacker can deceive the code and bypass the authentication because the above SQL statement will always return the record of the user 123. This is due to the fact that the two consecutive dashes are used in SQL to comment a line so the DBMS will ignore everything after them. Therefore, the password part of the WHERE clause is ignored and the hacker will be able to log in as if he or she is the user 123. Quite trivial yet a catastrophic trick!

Categories of SQL Injection Attacks

SQL Injection attacks against databases can be categorized as follows (Kost, 2004).

SQL Manipulation

This occurs when the SQL statement is modified through the use of SET operations or when the WHERE clause is exploited to let the SQL statement return a different set of rows. Modifying the WHERE clause of the user authentication statement is among the most well-known attacks. Using several tricks and techniques, hackers will deceive the code so that the WHERE clause will always result in true (Overstreet, 2003).

As an example, consider the following piece of code.

```
Declare @SQL nvarchar(100)
Declare @SingleQuote char(1)
```

```
Set @SingleQuote = ‘’’
Set @SQL = ‘Select * from Users Where User-
Name = ‘ + @SingleQuote + @UserName + @
SingleQuote + ‘ and Password = ‘ + @SingleQuote
+ @UserPassword + @SingleQuote
Execute (@SQL)
```

What will happen if the user's input is "admin" as the username and "secret" as the password? The SQL statement that will be executed is the following.

```
Select * from Users Where UserName = ‘admin’
and Password = ‘secret’
```

Consider what will happen when the user makes the following entry as both username and password.

```
Any’ or ‘1’ = ‘1
```

The SQL statement that will result is the following:

```
Select * from Users Where UserName = ‘ Any’ or
‘1’ = ‘1’ and Password = ‘ Any’ or ‘1’ = ‘1’.
```

When this SQL statement is executed, all the records in the table will be retrieved because the WHERE clause will always be true.

Code Injection

A classic example is to append an EXECUTE command to the vulnerable SQL statement. This type of attacks may work only when multiple SQL statements per DB request are supported. It is essential to go back to the previous example in order to check what might happen if a hacker is trying to perform code injection.

```
Declare @SQL nvarchar(100)
Declare @SingleQuote char(1)
Set @SingleQuote = ‘’’
Set @SQL = ‘Select * from Users Where User-
Name = ‘ + @SingleQuote + @UserName + @
SingleQuote + ‘ and Password = ‘ + @SingleQuote
+ @UserPassword + @SingleQuote
Execute (@SQL)
```

At this point, suppose the hacker made the following entries.

```
Username        : any
Password        : any’ ; sp_addlogin ‘USERme’,
‘MYpwd’ ; SELECT ‘any
```

What might the result be if the user supplied the above parameters? The following is the concatenated SQL statement.

```
Select * from Users Where UserName = ‘any’ and
Password = ‘any’; sp_addlogin ‘USERme’, ‘MYpwd’
; SELECT ‘any’
```

This is the SQL statement that will be executed by the EXECUTE command. Obviously, there are three SQL statements.

- The first SQL statement will return no records. The hacker is no more interested in it since he or she can do more by the SQL after the semicolon.
- The second will enable the hacker to penetrate the database because he or she did create an account for him or her. Using the same technique, the hacker will be able to add himself or herself to the system administrators group.
- The third SQL statement is used just to avoid errors. The single quote at the end of the original SQL statement needs to be dealt with. If ignored, an error may occur.

Function Call Injection

This occurs when user-defined functions or DB built-in functions are inserted into vulnerable SQL statements. These functions may manipulate the data in the database or request the operating system to call critical commands (Finnigan, 2002). To give an example, one of the SQL Server's commands will be used. It is called xp_cmdshell.

This command spawns a Windows command shell. In other words, it asks the operating system to execute one of its commands by passing in a string for execution.

The code snippet that follows shows an extreme-case scenario through which a hacker performs SQL injection to create a user and then adds it to the administrators group. When this happens, the hacker will be an administrator and will be able to do anything including formatting the hard disk, shutting down the system, and deleting the accounts of other administrators. The code is as follows.

```
exec master..xp_cmdshell 'net user MyUserName
MyPassword /add'
exec master..xp_cmdshell 'net localgroup admin-
istrators MyUserName /add'
```

The following example, taken from SQL Server 2005 documentation, focuses on the problem and highlights the risk under consideration. By using "net send," users are notified that the SQL Server is about to be shut down. By using "net pause," the server is paused. By using "net stop," the server is eventually shut down.

```
EXEC xp_cmdshell 'net send /domain:SQL_US-
ERS ''SQL Server will be shut down in 5 seconds.
Log off immediately.', no_output
WAITFOR DELAY '00:00:05'
EXEC xp_cmdshell 'net stop sqlserver', no_out-
put
```

Buffer Overflow

This type of attack is a subset of the previous, function call injection. In several commercial databases, several vulnerabilities that exist in a few built-in database functions may result in a buffer overflow.

Oracle Buffer Overflow

The standard Oracle database functions are vulnerable to known buffer overflows. The following list of functions is provided as an example.

- tz_offset
- to_timestamp_tz
- bfilename

Unfortunately, most Web servers and Web applications do not handle DB connection loss resulting from a buffer overflow. The process will usually keep hanging until the client connection is terminated. Thus, an effective denial-of-service attack will take place (Kost, 2004).

SQL Server Buffer Overflow

The main point here is that Transact-SQL is a rich programming environment. It interacts with a large number of APIs (application programming interfaces) and subcomponents, some of which will have buffer overflow problems (Anley, 2002).

One of the best defenses against such an action is to prevent distrusted users from submitting arbitrary SQL. Other defensive measures against Windows buffer overflows are as follows.

- Run SQL Server in the context of a low-privileged account. It should never run using the SYSTEM account.
- Ensure that this account cannot run the command processor (cmd.exe). This will greatly minimize the risk of arbitrary-command and reverse-shell exploits.
- Ensure that this account has minimal access to the file system. This will help in diminishing the risk of file-copy exploits.

It is important to bear in mind that the above points should never be relied upon as strategies for total protection. There are two scenarios that should be fully illustrated in order to understand why Web developers may write SQL-injection-vulnerable Web sites. These scenarios are as follows.

1. Bind-variable SQL scenario
2. Dynamic SQL scenario

In what follows, a sample SQL Server stored procedure that uses bind variables will be dem-

Box 1.

```
CREATE PROCEDURE [dbo].[DynamicSQL]
@Name1 nvarchar(40), @Name2 nvarchar(40), @Name3 nvarchar(40)
AS
BEGIN
DECLARE @SQL nvarchar(2000)
DECLARE @WHR nvarchar(2000)
SELECT @SQL = 'SELECT * FROM PersonTbl'
SELECT @WHR = ' WHERE '
      IF @Name1 <> ''
            BEGIN
                  SELECT @SQL = @SQL + @WHR + 'FirstName LIKE ''' + @Name1 + ''''
                  SELECT @WHR = ' AND '
            END
      IF @Name2 <> ''
            BEGIN
                  SELECT @SQL = @SQL + @WHR + 'FatherName LIKE ''' + @Name2 + ''''
                  SELECT @WHR = ' AND '
            END
      IF @Name3 <> ''
            BEGIN
                  SELECT @SQL = @SQL + @WHR + 'FamilyName LIKE ''' + @Name3 + ''''
                  SELECT @WHR = ' AND '
            END
      EXECUTE (@SQL)
END
```

onstrated. Then, another procedure that makes use of dynamic SQL will be presented. Next, both approaches will be compared. Finally, a new approach of dealing with the problem will be introduced so that the developer will be able to use bind variables but still be able to build SQL statements dynamically.

Bind-Variable SQL Scenario

Before posting a sample code, it is fundamental to show what Stephen Kost (2004) said regarding the concept of bind variables:

The most powerful protection against SQL injection attacks is the use of bind variables. Using bind variables will also improve application performance. Application coding standards should require the use of bind variables in all SQL statements. No SQL statement should be created by concatenating together strings and passed parameters.

The following code represents a stored procedure that uses bind variables.

```
CREATE PROCEDURE dbo.BindVariables
(@Name1 nvarchar(40), @Name2 nvarchar(40),
@Name3 nvarchar(40))
AS
BEGIN
SELECT * FROM PersonTbl
WHERE PersonTbl.FirstName = @Name1
AND PersonTbl.FatherName = @Name2
AND PersonTbl.FamilyName = @Name3
END
```

It is obvious that the developer who wrote this procedure decided not to build the SQL in a dynamic way. Instead, it is an already defined statement that expects parameters all of which should be supplied. It is not created by concatenating together strings and passed parameters. Parameter binding is a simple solution that lets you put placeholders in the SQL statement and binds them to a particular parameter (Jepson, 2002).

The main disadvantage here is that the user has to send all three parameters. But what will the user do if he or she knows only the first name of the person? The user will not be able to use this procedure because of the need to supply all of the parameters. Therefore, the developer has to go for the other option: to build the SQL statement dynamically.

Dynamic SQL Scenario

Web applications are vulnerable to SQL injection only when strings input by end users are not validated in a proper way and are passed to be concatenated to a dynamic SQL statement (Kost, 2004).

The code shown in Box 1 represents an SQL Server stored procedure that dynamically creates the SQL statement then executes it. As you can see, the resulting SQL is not known at design time. It will be only known at the run time. This is why it is called dynamic.

As clarified in the code above, this stored procedure is dynamically building the WHERE clause of the SQL statement. After concatenating strings together, the produced SQL statement will be executed. This is very dangerous since the user input may lead to catastrophic results if not validated. The main drawback of the above approach is the fact that the SQL statement that will be executed is the string that results from concatenating several substrings entered by the user.

Input Validation

If the bind-variables approach is not used, all parameters passed to a dynamic SQL statement have to be validated. If a user's input strings are passed to a dynamic SQL statement before being validated, the Web application is then vulnerable to SQL injections (Kiely, 2002). Every string that is passed as a parameter should be subject to validation. If a bind variable is not in use, special DB characters have to be escaped (Litwin, 2002).

All single quotes, for example, have to be removed. Other characters of critical nature may also have to be escaped. Every DBMS has a list of characters that need to be escaped. If SQL Server is used, then semicolons (;) have to be fetched and replaced since this character is used to separate two consecutive SQL commands. Restricting the maximum length of user's input strings is an inevitable way to validate user input. For example, if there is a field that is used to enter the first name, no more than 20 characters should be allowed. If there is a field that is used for entering a password, its maximum length should not allow the hacker to enter the following as a password.

```
1' ; drop table employees ; select '
```

The Web developer should review all the controls inside all of his or her pages to make sure that no control will accept too many characters. Strictly speaking, input validation is a good practice but it does not provide a complete solution. A user should be able to enter any character in the password field. Furthermore, hackers may encode their input so that detecting special characters is much trickier. The following section illustrates this idea.

Encoding Injected Statements

There are many ways for encoding SQL queries. Consider the following example.

```
declare @Encoded varchar(4000)
set @Encoded = 0x73656c65637420404076657
273696f6e execute(@Encoded)
```

This will run "select @@version," as does the following.

```
declare @AnotherEx nvarchar(4000)
Set @AnotherEx = 0x730065006c0065006300
740020004000400076006500720073006900 6f
006e00
execute(@AnotherEx)
```

A parameter may contain multiple SQL statements, although single quotes or semicolons are not used.

The BBD Technique

At this point, BBD (bind but dynamic) comes into action. It reveals a new approach that enables the developer to dynamically produce bind-variable SQL statements. This is to say that the developer can still create a dynamic SQL statement, but through using bind variables only. The BBD concept focuses on using a predefined SQL statement that cannot be misused by the user. It is not an SQL statement defined at the time of execution.

The WHERE clause is always the same, but the input of the user will control the way it behaves. This control is achieved using the CASE command. The used SQL statement should be saved as a stored procedure since it is far more difficult to break with SQL injection. Stored procedures require a specified number of parameters to be supplied in specified places, formats, types, and sizes. When satisfied, these many prerequisites will ensure better security. Improved performance is another advantage of using stored procedures.

Proposed BBD Scenario

Consider the following stored procedure that offers the same functionality.

```
CREATE PROCEDURE [dbo].[BBD]
@Name1 nvarchar(40), @Name2 nvarchar(40),
@Name3 nvarchar(40)
AS
BEGIN
SELECT * FROM PersonTbl
WHERE
CASE WHEN @Name1 IS NULL THEN First-
Name ELSE @Name1 END = FirstName
AND
CASE WHEN @Name2 IS NULL THEN Father-
Name ELSE @Name2 END = FatherName
AND
CASE WHEN @Name3 IS NULL THEN Fami-
lyName ELSE @Name3 END = FamilyName
END
```

Before explaining the idea behind this, there is a need to explain the CASE function syntax: The simple CASE function, as revealed in the SQL Server books online, compares an expression to a set of simple expressions and returns the result of the comparison. The following example displays the list price based on the price range.

```
SELECT ProductNumber, 'Price Range' =
CASE
WHEN ListPrice < 50 THEN 'Under $50'
WHEN ListPrice >= 50 and ListPrice < 250 THEN
'Under $250'
ELSE 'Over $250'
END
FROM Product
```

Now, it is time to go back to the BBD procedure to see the effect of the CASE function. The WHERE clause can include several CASE functions each of the following form.

```
CASE WHEN CONDITION THEN ValueToRe-
turn ELSE AnotherValueToReturn END
```

To simplify the idea, one should look at each CASE clause as if it represents a column in a table. Hence, everything between CASE and END should be considered as a column. Looking at it this way, the above BBD procedure can be rewritten as follows:

```
CREATE PROCEDURE [dbo].[BBD]
@Name1 nvarchar(40), @Name2 nvarchar(40),
@Name3 nvarchar(40)
AS
BEGIN
SELECT * FROM PersonTbl WHERE
Field1 = FirstName AND Field2 = FatherName
AND Field3 = FamilyName
END
```

Field1 in the syntax above is the replacement of the following line.

```
CASE WHEN @Name1 IS NULL THEN First-
Name ELSE @Name1 END
```

*Table 1. BBD run time***

	Exp. 1	Exp. 2	Exp. 3	Exp. 4	Exp. 5	Exp. 6
DBMS	SQL Server	SQL Server	SQL Server	Oracle 10g	MS Access	MS Access
Total number of records	700,000	3,000,000	5,600,000	200,000	400,000	811,000
Number of returned records	134	550	1,050	200,000	128	256
Field data type	nchar (100)	nchar (100)	Int (numeric)	Datetime	Text (50)	Text (50)
Run time (dynamic approach)	1 second	2 seconds	25 seconds	3 seconds	2 seconds	32 seconds
Run time (BBD approach)	1 second	2 seconds	25 seconds	4 seconds	2 seconds	32 seconds

** *All the experiments reflect that the BBD approach will never slow the SELECT statement. Consequently, the BBD approach will have no effect on the speed of the execution if the database is of reasonable size. All the queries were run against databases of 1 or 2GB in size at most.*

Note that the effect of the CASE function is the same for all the records of the target table. Either the CASE function will restrict the data to be retrieved or it will do nothing as if it did not exist at all.

What will happen if the user provides a value for @Name1? If a value is provided, the CASE END clause will return this provided value for all the records. That is why it restricts the records to be returned. Consider the following.

```
WHERE
CASE WHEN @Name1 IS NULL THEN First-
Name ELSE @Name1 END = FirstName
```

The previous line can be rewritten as follows: WHERE @Name1 = FirstName. Because the value is provided, the first part of the CASE will return false. Hence, the other part will be returned. This part is @Name1. Clearly, this is what the developer wants. The developer simply developed the SQL in a dynamic way but without concatenating strings. This is the secret behind the concept of BBD. It is dynamic because several parts of the WHERE clause can dynamically be either ignored or considered depending on the parameter supplied. Binding comes onto play because the WHERE clause is predefined and not subjected to risky user's input strings.

Consider the case where the user does not provide a value for @Name1. If a value is not provided, the CASE END clause will return true for all the records. This is to say that this part of the WHERE clause is ignored because it is true for every record in the table. Consider the following.

```
WHERE
CASE WHEN @Name1 IS NULL THEN First-
Name ELSE @Name1 END = FirstName
```

The previous line can be rewritten as follows: WHERE FirstName = FirstName. Because the value is not provided, the first part of the CASE is true. Hence, the FirstName will be returned. Since FirstName = FirstName is always true, this part of the WHERE clause will have no effect. There is one exception to this: The expression NULL = NULL will not return true. Therefore, NULL values require special care. Two scenarios apply here to overcome the NULL problem.

- Disallow the table column to accept NULL values. This way, no records stored in the table will contain a NULL value in the specified column.
- Define a new function that converts a NULL value to an empty string. It should be a very simple one that checks the parameter value. If it is NULL, an empty string should be returned. If the value sent to the function is not NULL, return it as is. The following shows such a function.

```
CREATE FUNCTION Nul2Str(@Val nvar-
char(100))
RETURNS nvarchar(100)
AS
```

```
BEGIN
    If @Val IS NULL SELECT @Val = ‘’
    RETURN @Val
END
```

This function should then be used before and after the operand. Before using Nul2Str, the following line will still suffer from NULL values.

```
WHERE
CASE WHEN @Name1 IS NULL THEN First-
Name ELSE @Name1 END = FirstName
```

After defining the function, it can be used as follows.

```
WHERE
Nul2Str (CASE WHEN @Name1 IS NULL
THEN FirstName
ELSE @Name1 END) = Nul2Str (FirstName)
```

BBD Experiments

Several queries were executed in order to test how efficient this approach is and to what extent the CASE clause will affect the execution time. The results are shown in Table 1.

BBD Experiments Summary

All experiments were performed under the same conditions and in the same environment. This includes the operating system and the hardware configuration. No optimizations have been made. Default settings were used.

The query optimizer, after analyzing the submitted query, discovers that the CASE command is dealing with fixed values. Therefore, there is no need to check the value of the parameter for each record. Perhaps, the optimizer is designed in a smart way so that it realizes that the fixed-value parameter should be taken out of the WHERE statement.

The syntax of the query used in the second experiment is as follows.

```
DECLARE @Serial int

SELECT @Serial = 0

SELECT * FROM Clients WHERE Serial <=
CASE WHEN @Serial = 0 THEN Serial ELSE
@Serial END
```

The above SQL statement will produce the same output as the following.

```
SELECT * FROM Clients
```

Because @Serial = 0, the second part of the CASE clause will be ignored. Therefore, the following WHERE clause is the one that will result.

```
WHERE Serial <= Serial
```

This query will return all records since the WHERE clause returns true for all the records.

For those who are interested in the syntax used in Microsoft Access, this is the syntax that was used in the fourth experiment.

```
SELECT * FROM PersonTbl
WHERE IIF ([UserInput] IS NULL, PersonTbl.
PrsName, [UserInput]) = PersonTbl.PrsName
```

FUTURE TRENDS

In the future, several experiments need to be conducted. These experiments will add more value to the findings presented here. They could cover aspects such as the following.

1. Experiments focusing on SQL injections using databases of greater sizes.
   - 10GB
   - 100GB

   It is extremely important to prove whether or not the BBD approach is still effective when the database is of huge size. Future experiments will be decisive in this regard.
2. Experiments focusing on SQL injections

using other DBMSs

- Sybase
- DB2

Many experiments have been conducted using SQL Server 2005, Oracle 10g, and Microsoft Access 2003. In the future, more experiments are expected to be conducted using other commercial DBMSs.

CONCLUSION

When it is time to build SQL statements, four choices are available for Web developers.

1. Build the SQL statement in a dynamic way by concatenating strings according to the user's input. This approach provides maximum flexibility; however, it is dangerous since hackers can take advantage of it and insert malicious strings. The dynamic approach is also slower than the bind-variable approach.
2. Build the SQL statement using the bind-variable approach and save it to the database in the form of a stored procedure. This way, SQL injections will not be an option to hackers. Although this approach is safe, it lacks flexibility because it does not allow the developer to dynamically change the WHERE clause as it is the case with dynamic SQL. When there is a need to control the WHERE clause dynamically, this approach is not an option.
3. Build the SQL statement in a dynamic way and enforce input validation. For instance, the developer may reject any character other than [a...z], [A...Z], or [0...9]. The single quotes and the semicolons are among the critical characters that have to be rejected. Although it is not enough to rely on it solely, a solid input validation policy should be applied.
4. Build the SQL statement using the BBD approach. When a developer applies BBD, he or she will build the SQL dynamically (advantage of Choice 1), or benefit from the safety of the bind-variable approach (advantage of Choice 2).

REFERENCES

Anley, C. (2002). Advanced SQL injection. *NGSSoftware*. Retrieved August 2, 2006, from http://www.ngssoftware.com

Cerrudo, C. (2002). Manipulating Microsoft SQL Server using SQL injection (Tech. Rep.). *Application Security, Inc.* Retrieved July 29, 2006, from http://www.appsecinc.com/presentations/Manipulating_SQL_Server_Using_SQL_Injection.pdf

Finnigan, P. (2002). SQL injection and Oracle, part one (Tech. Rep.). *Security Focus*. Retrieved August 30, 2006, from http://www.securityfocus.com/infocus/1644

Jepson, B. (2002). *Beware SQL injection in Web applications.* Retrieved September 12, 2006, from http://www.oreillynet.com/mac/blog/2002/06/beware_sql_injection_in_web_ap.html

Kiely, D. (2002). *Guarding against SQL injection attacks.* Retrieved September 5, 2006, from http://www.itworld.com/nl/windows_sec/03252002/

Kost, S. (2004). An introduction to SQL injection attacks for Oracle developers. *Help Net Security.* Retrieved August 1, 2006, from http://www.net-security.org/article.php?id=633

Litwin, P. (2002). *Guard against SQL injection attacks.* Retrieved July 23, 2006, from http://www.aspnetpro.com/opinion/2002/08/asp200208pl_o/asp200208pl_o.asp

Overstreet, R. (2003). *Protecting yourself from SQL injection attacks.* Retrieved June 16, 2006, from http://www.4guysfromrolla.com/webtech/061902-1.shtml

KEY TERMS

Bind-Variable SQL: SQL statement that expects parameters, all of which should be supplied. It is not created by concatenating together strings and passed parameters.

Code Injection: This occurs when attackers insert database commands or new fragments of SQL statements into the original one.

DB Built-In Functions: Intrinsic functions that are merged into an RDBMS as a part of it, not added by the user.

Dynamic SQL: SQL statements that are created, prepared, and executed while a program is executing. It is, therefore, possible with dynamic SQL to change the SQL statement during program execution and have many variations of an SQL statement at run time.

Encoding Injected Statements: A way to encode user input so that malicious characters are accepted. Hackers do so to deceive the code checking for invalid input by supplying characters in an unexpected format.

Function Call Injection: This type of SQL injection occurs when database functions are inserted into vulnerable SQL statements.

SQL Injections: SQL injection is the name for a general class of attacks that can allow nefarious users to retrieve data, alter server settings, or even take over your server if you are not careful. SQL injection is not an SQL server problem, but a problem with improperly written applications.

Compilation of References

Aas, K. (2001). *Microarray Data Mining: A Survey.* NR Note, SAMBA, Norwegian Computing Center.

Abadi, D. et al. (2005). REED: Robust, Efficient Filtering and Event Detection in Sensor Networks. *In Proceedings of VLDB 2005,* Trodheim, Norway.

Abel, D. J., Taylor, K., Ackland, R., & Hungerford, S. (1998). An exploration of GIS architectures for Internet environments. *Computers, Environment, and Urban Systems, 22*(1), 7-23.

Abelló, A., Samos, J., & Saltor, F. (2006). YAM^2: A multidimensional conceptual model extending UML. *Information Systems, 32*(6), 541–567.

Aberer, K. (2001). P-grid: A self-organizing access structure for P2P information systems. *Proceedings of the 6th International Conference on Cooperative Information Systems* (pp. 179-194).

Abiteboul, S, & Hull, R. (1987). IFO: A Formal Semantic Database Model. *ACM Trans. Database Syst., 12*(4), 525-565.

Abiteboul, S., Agrawal, R., Bernstein, P., Carey, M., Ceri, S., Croft, B., et al. (2005). The Lowell database research self-assessment. *Communications of the ACM, 48*(5), 111-118.

Abiteboul, S., Bonifati, A., Cobéna, G., Manolescu, I., & Milo, T. (2003). *Dynamic XML documents with distribution and replication.* Proceedings of SIGMOD 2003, San Diego, California.

Abiteboul, S., Buneman, P., & Suciu, D. (1999). *Data on the Web: From relations to semistructured data and XML.* Morgan Kaufmann Publishers.

Abiteboul, S., Cluet, S., Christophides, V., Milo, T., Moerkotte, G., & Simeon, J. (1997). Querying documents in object databases. *International Journal on Digital Libraries, 1*(1), 5-19.

Abiteboul, S., Hull, R., & Vianu, V. (1994). *Foundations of databases.* Addison-Wesley.

Abowd, G., & Mynatt, E. (2000). Charting past, present, and future research in ubiquitous computing. *ACM Transactions on Computer-Human Interaction, 7*(1), 29-58.

Abrial, J. R. (1974). Data semantics. In J. W. Klimbie, & K. L. Koffeman (Eds.), Data Base Management. Amsterdam: North Holland Publishing Company, (pp. 1-60).

Abu Bakar, A., et al. (2003). Rough discretisation approach in probability analysis for neural classifier. *ITSIM.*

Acharya, S., Alonso, R., Franklin, M., & Zdonik, S. (1995). Broadcast Disks: Data Management for Asymmetric Communication Environments. *Proc. of the 1995 ACM Int. Conf. on Management of Data,* (pp. 199-210).

Acharya, S., Gibbons, P. B., & Poosala, V. (1999). AQUA: A fast decision support system using approximate query answers. *Proceedings of the 25th International Conference on Very Large Data Bases* (pp. 754-757).

Acharya, S., Gibbons, P. B., Poosala, V., & Ramaswamy, S. (1999). Join synopses for approximate query answering. *Proceedings of the 1999 ACM International Conference on Management of Data* (pp. 275-286).

Acharya, S., Gibbons, P. B., Poosala, V., & Ramaswamy, S. (1999b). Join Synopses for Approximate Query Answering. *Proc of the 1999 ACM Int. Conf. on Management of Data,* (pp. 275-286).

ACM-SIGKDD Explorations, 5(2), 1-5.

ACORD (2007). ACORD Corporation. www.acord.org.

Active Internet users by country, July 2004. (2004, August 25). *ClickZ*. Retrieved July 10, 2008, from http://www.clickz.com/stats/sectors/geographics/article.php/3397231

Adam, N. R., & Wortmann, J. C. (1989). Security-control methods for statistical databases: A comparative study. *ACM Computing Surveys, 21*(4), 515-556.

Addey, D., Ellis, J., Suh, P., & Thiemecke, D. (2002). *Content management systems*. Birmingham, UK: glasshaus.

Adobe Flash, Retrieved October 8, 2007, from *http://www.adobe.com/products/flash/*

Aebi, D. (1997). Data engineering: A case study. In C. J. Risjbergen (Ed.), *Proceedings in advances in databases and information systems*. Berlin, Germany: Springer Verlag.

Aebi, D., & Largo, R. (1994). Methods and tools for data value re-engineering. In *Lecture notes in computer science: Vol. 819. International Conference on Applications of Databases* (pp. 1-9). Berlin, Germany: Springer-Verlag.

Aerts, K., Maesen, K., & van Rompaey, A. (2006). *A practical example of semantic interoperability of large-scale topographic databases using semantic Web technologies*. Proceedings of the AGILE'06: 9th Conference on Geographic Information Science, Visegrd, Hungary, 35–42.

Agarwal, P. K., & Procopiuc C. M. (2002). Advances in indexing for mobile objects. *IEEE Data Engineering Bulletin, 25*(2), 25-34.

Agarwal, P. K., Arge, L., & Erickson, J. (2000). Indexing moving points. *Proc. of the 19th ACM Symposium on Principles of Database Systems (PODS'2000),* 175-186. Madison, WI.

Agarwal, R., Sinha, A. P., & Tanniru, M. (1996a). Cognitive fit in requirements modeling: A study of object and process methodologies. Journal of Management Information Systems, 13(2) (Fall), 137-162.

Agarwal, R., Sinha, A. P., & Tanniru, M. (1996b). The role of prior experience and task characteristics in object-oriented modeling: An empirical study. International Journal of Human-Computer Studies, 45, 639-667.

Agarwal, S., Keller, A. M., Wiederhold, G., & Saraswat, K. (1995). *Flexible relation: An approach for integrating data from multiple, possibly inconsistent databases.* ICDE.

Aggarwal, C. C., & Yu, P. S. (2001). Outlier detection in high dimensional data. *Proceedings of the 2001 ACM SIGMOD International Conference on Management of Data (SIGMOD'01)* (pp. 37-46).

Aggarwal, C. C., Han, J., Wang, J., & Yu, P. S. (2003). *A framework for clustering evolving data streams.* Paper presented at the 29th international conference on Very Large Data Bases, Berlin, Germany.

Aggarwal, C. C., Wolf, J. L., Yu, P. S., Procopiuc, C., & Park, J. S. (1999). *Fast algorithms for projected clustering.* Paper presented at SIGMOD '99: the 1999 ACM SIGMOD international conference on Management of Data, New York, NY, USA.

Aggarwal, C., Han, J., Wang, J., & Yu, P. (2003). A framework for clustering evolving data streams. *Proceedings of the 29th International Conference on Very Large Databases* (pp. 81-92).

Aggarwal, G., Bawa, M., Ganesan, P., Garcia-Molina, H., Kenthapadi, K., Mishra, N., et al. (2004). Vision paper: Enabling privacy for the paranoids. *Proceedings of the 30th International Conference on Very Large Data Bases (VLDB'04)* (pp. 708-719).

Aggrawal, R., Gehrke, J., Gunopulos, D., & Raghavan, P. (1998). Automatic subspace clustering of high dimensional data for data mining applications. *Proceedings of the 1998 ACM SIGMOD International Conference on Management of Data (SIGMOD'98)* (pp. 94-105).

Agrawal R. & Srikant R. (1994). Fast Algorithms for Mining Association Rules in Large Databases. Proceedings of the 20th International Conference on Very Large Data Bases, Santiago, Chile, 478-499.

Agrawal, R., & Srikant, R. (1994, September). *Fast Algorithms for Mining Association Rules in Large Databases.* Paper presented at the 20th International Conference on Very Large Data Bases (VLDB), Santiago, Chile.

Agrawal, R., & Srikant, R. (1995). *Mining sequential patterns.* Proceedings of the 11th International Conference on Data Engineering (ICDE 95). Taipei, Taiwan, 3–14.

Agrawal, R., & Srikant, R. (2000). *Privacy preserving data mining.* Proceedings of ACM SIGMOD Conference on Management of Data (SIGMOD'00), Dallas, TX.

Agrawal, R., Asonov, D., Kantarcioglu, M., & Li, Y. (2006). *Sovereign joins.* Proceedings of the 22nd International Conference on Data Engineering (ICDE'06).

Agrawal, R., Evfimievski, A., & Srikant, R. (2003). Information sharing across private databases. *Proceedings of ACM SIGMOD International Conference on Management of Data (SIGMOD'03)* (pp. 86-97).

Agrawal, R., Faloutsos, C., & Swami, A. (1993). Efficient similarity search in sequence databases. *Proceedings of the 4th Conference on Foundations of Data Organization and Algorithms* (pp. 69-84).

Agrawal, R., Gehrke, J., Gunopulos, D., & Raghavan, P. (1998). *Automatic subspace clustering of high dimensional data for data mining applications.* Paper presented at SIGMOD '98: the 1998 ACM SIGMOD international conference on Management of Data, New York, NY, USA.

Agrawal, R., Gupta, A., & Sarawagi, S. (1997). Modeling multidimensional databases. *13th International Conference on Data Engineering (ICDE'97)* (pp. 232-243).

Agrawal, R., Kiernan, J., Srikant, R., & Xu, Y. (2002). *Hippocratic databases.* Proceedings of the 28th International Conference on Very Large Data Bases (VLDB'02), Hong Kong, China.

Agrawal, R., Lin, K.-I., Sawhney, H. S., & Swim, K. (1995). Fast similarity search in the presence of noise, scaling, and translation in time-series databases. *Proceedings of VLDB*, 490-501. Zurich, Switzerland.

Agrawal, R., Srikant, R., & Thomas, D. (2005). Privacy preserving OLAP. *Proceedings of ACM SIGMOD International Conference on Management of Data (SIGMOD'05)* (pp. 251-262).

Agrawal, S., Chaudhuri, S., & Narasayya, V. (2001). Materialized view and index selection tool for Microsoft SQL Server 2000. *ACM SIGMOD International Conference on Management of Data (SIGMOD 2001)* (p. 608).

Agrawal, S., Chaudhuri, S., Kollàr, L., Marathe, A. P., Narasayya, V. R., & Syamala, M. (2004). Database tuning advisor for Microsoft SQL Server 2005. *30th International Conference on Very Large Data Bases (VLDB 2004)* (pp. 1110-1121).

Agrawal, S., Chaudhuri, S., Kollar, L., Marathe, A., Narasayya, V., & Syamala, M. (2004). Database tuning advisor for Microsoft SQL server 2005. *In Proceedings of the 30th Very Large Database Conference* (VLDB), 1110-1121. Toronto, Canada.

Ahmed, T., & Miquel, M. (2005). *Multidimensional structures dedicated to continuous spatio-temporal phenomena.* Proceedings of the 22nd British National Conference on Databases, 29–40.

Aiken, P. (1996). *Data reverse engineering: Slaying the legacy dragon.* McGraw-Hill.

Aiken, P., & Muntz, A. (1993). A framework for reverse engineering DoD legacy information systems. *WCRE.*

Aksnes, D., & Taxt, R. (2004). Peer review and bibliometrics indicators: A comparative study at a Norwegian university. *Research Evaluation, 13*(1), 33-41.

Akyildiz, I. F., Su, W., Sankarasubramaniam, Y., & Cayirci, E. (2002). Wireless Sensor Networks: A Survey. *Computer Networks, 38*, 393-422.

Akyildiz, I. F., Su, W., Sankarasubramaniam, Y., & Cayirci, E. (2002). A Survey on Sensor Networks. *IEEE Communications Magazine, 40*(8), 102-114.

Alam, M., & Wasan, S. K. (2006). Migration from relational database to object oriented database. *Journal of Computer Science, 2*(10), 781-784.

Alasoud, A., Haarslev, V., & Shiri, N. (2005). A hybrid approach for ontology integration. *Proceedings of the 31st VLDB Conference.* Trondheim, Norway.

Albrecht, J. (1999). Geospatial information standards: A comparative study of approaches in the standardization of geospatial information. *Computers & Geosciences, 25*(1), 9–24.

Al-Jumaily, H. T., Pablo C., Cuadra D., & Martínez P. (2006). Using UML's sequence diagrams as termination analyzer for triggers-based executing. *In the 23rd British National Conference on Databases,* 18-20. Northern Ireland, Queen's University Belfast.

Allen, J. F. (1983). Maintaining knowledge about temporal intervals. *Communications of the ACM, 26*(11), 832-843.

Alon, N., Gibbons, P. B., Matias, Y., & Szegedy, M. (1999). Tracking join and self-join sizes in limited storage.

Proceedings of the 18th ACM International Symposium on Principles of Database Systems (pp. 10-20).

Alon, N., Matias, Y., & Szegedy, M. (1996). The space complexity of approximating the frequency moments. *Proceedings of the 28th ACM International Symposium on Theory of Computing* (pp. 20-29).

Alonso, G., Casati, F., Kuno, H., & Machiraju, V. (2004). *Web Services - Concepts, Architectures and Applications.* Springer-Verlag.

Alonso, G., Casati, F., Kuno, H., & Machiraju, V. (2004). *Web Services - Concepts, Architectures and Applications.* Springer-Verlag.

Alonso, L. and Schott, R. (1993). On the tree inclusion problem. In *Proceedings of Mathematical Foundations of Computer Scienc*e, 211-221.

Alonso-Jiménez, J. A., Borrego-Díaz, J., Chávez-González A., Gutiérrez-Naranjo M. A., & Navarro-Marín, J. D. (2003). Towards a practical argumentative reasoning in qualitative spatial databases. *Proceedings of the 16th International Conference on Industrial & Engineering Applications of Artificial Intelligence & Expert Systems (IEA/ AIE 2003), Lecture Notes in Computer Science, 2850* (pp. 789-798).

Alpaydin, E. (2004). *Introduction to Machine Learning*: The MIT Press.

Alsabti, K., Ranka, S., & Singh, V. (1998). *An Efficient K-Means Clustering Algorithm*. Paper presented at the First Workshop on High Performance Data Mining, Orlando, Florida.

Alvarez, M., Pan, A., Raposo, J., & Vina, J. (2004). Client-side deep web data extraction. *IEEE Conference on e-Commerce Technology for Dynamic E-Business (CEC-East'04),* 158-161.

Alwan, A. A., Ibrahim, H., & Udzir, N. I. (2007). Local integrity checking using local information in a distributed database. *Proceedings of the 1st Aalborg University IEEE Student Paper Contest 2007 (AISPC'07).* Aalborg, Denmark.

Alwan, A.A., Ibrahim, H., & Udzir, N.I. (2007). *Local integrity checking using local information in a distributed database.* Proceedings of the 1st Aalborg University IEEE Student Paper Contest 2007 (AISPC'07), Denmark.

Ambite, J. L., Arens, Y., Ashish, N., Knoblock, C. A., & collaborators (1997). The SIMS manual 2.0. Technical Report, *University of Southern, California.* December 22. Retrieved August 01, 2007, from http://www.isi.edu/sims/papers/sims-manual.ps.

Ambler, S. W. (2003). *Agile database techniques.* Wiley.

Ambler, S. W., & Sadalage, P. J. (2006), *Refactoring databases: Evolutionary database design*. Addison Wesley Professional.

Amer Yahia, S., Lakshmanan, L., & Pandit, S. (2004). FlexPath: Flexible Structure and Full-Text Queryng for XML. In *Proc. of ACM SIGMOD* (pp. 83–94), Paris, France.

Amer-Yahia S. (2003). *Storage Techniques and Mapping Schemas for XML*. Technical Report TD-5P4L7B, AT&T Labs-Research.

Amer-Yahia, S., Du, F., & Freire, J. (2004). A Comprehensive Solution to the XML-to-Relational Mapping Problem. In *WIDM'04: Proceedings of the 6th Annual ACM International Workshop on Web Information and Data Management*, (pp. 31-38), New York, NY, USA: ACM Press.

Amza, C., Cox, A. L., & Zwaenepoel, W. (2005). A Comparative Evaluation of Transparent Scaling Techniques for Dynamic Content Servers. *International Conference on Data Engineering,* (pp. 230-241).

An, Y., Borgida, A., & Mylopoulos, J. (2005). Constructing semantic mappings between XML data and ontologies. *ISWC'05.*

Andersson, M. (1994). Extracting an entity relationship schema from a relational database through reverse engineering. *Proceedings of the 13th International Conference on ER Approach.*

Andre-Joesson, H. and Badal, D. (1997). Using signature files for querying time-series data. In *Proc. 1st European Symp. on Principles of Data Mining and Knowledge Discovery.*

Andros, D., Cherrington, J. O., & Denna, E. L. (1992). Reengineer your accounting the IBM way. The Financial Executive, July/August, (pp. 28-31).

Angiulli, F., & Pizzuti, C. (2002). Fast outlier detection in high dimensional spaces. *Proceedings of the 6th European*

Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD'02) (pp. 15-26).

Ankerst, M., Breunig, M. M., Kriegel, H.-P., & Sander, J. (1999). *OPTICS: ordering points to identify the clustering structure.* Paper presented at SIGMOD '99: the 1999 ACM SIGMOD international conference on Management of Data, New York, NY, USA.

Ankolekar, A., Burstein, M., Hobbs, J., Lassila, O., Martin, D., McDermott, D., McIlraith, S., Narayanan, S., Paolucci, M., Payne, T., & Sycara, K. (2002). DAML-S: Web service description for the Semantic Web. In *Proceedings of the First International Semantic Web Conference – ISWC 2002* (LNCS 2342). Heidelberg: Springer-Verlag.

Anley, C. (2002). Advanced SQL injection. *NGSSoftware.* Retrieved August 2, 2006, from http://www.ngssoftware.com

Antoniou, G., & van Harmelen, F. (2004 April). *A semantic Web primer.* The MIT Press.

Antova, L., Koch, C., & Olteanu, D. (2007). 10^10^6 Worlds and Beyond: Efficient Representation and Processing of Incomplete Information. *Proceedings of the 23rd International Conference on Data Engineering ICDE'07,* Istanbul, Turkey.

Aouiche, K., Darmont, J., Boussaïd, O., & Bentayeb, F. (2005). Automatic selection of bitmap join indexes in data warehouses. In *Lecture notes in computer science: Vol. 3589. 7th International Conference on Data Warehousing and Knowledge Discovery (DaWaK 2005)* (pp. 64-73). Berlin, Germany: Springer.

Aouiche, K., Jouve, P. E., & Darmont, J. (2006). Clustering-based materialized view selection in data warehouses. In *Lecture notes in computer science: Vol. 4152. 10th East-European Conference on Advances in Databases and Information Systems (ADBIS 2006)* (pp. 81-95). Berlin, Germany: Springer.

APICS. (1998). Defining enterprise resource planning. http://www.apics.org/OtherServices/articles/defining.htm.

Apinar, I. B., Sheth, A., Ramakrishnan, C., Usery, E. L., Azami, M., & Kwan, M.-P. (2005). Geospatial ontology development and semantic analysis. In J. P. Wilson & S. Fotheringham (Eds.), *Handbook of geographic information science.* Blackwell Publishing.

Apolloni, B., Zamponi, G., & Zanaboni, A. M. (2000). An integrated symbolic/connectionist architecture for parsing Italian sentences containing pp-attachment ambiguities. *Applied Artificial Intelligence, 14*(3), 271-308.

Apt, K. R., & van Emden, M. H. (1982). Contributions to the Theory of Logic Programming. *Journal of the ACM, 29*(3), 841-862.

Apte, A. (2002). *Java Connector Architecture: Building Enterprise Adaptors.* SAMS Publishing.

Aragão, M., & Fernandes A. (2004). Seamlessly supporting combined knowledge discovery and query answering: A case study. In E. Suzuki & S. Arikava (Eds.), *Proceedings of the 7th International Conference "Discovery Science.* Lecture Notes in Computer Science, *3245*, 403-411. Berlin: Springer.

Arenas, M., Bertossi, L., & Chomicki, J. (1999). Consistent query answers in inconsistent databases. *Proceedings of PODS 1999* (pp. 68-79).

Arévalo, J. L., Polo, A., & Fernández, J. M. (2006). Representing Versions in XML documents using versionstamp. In *ER (Workshops) 2006, Lecture Notes in Computer Science, 4231,* 257-267.

Arge, L., Procopiuc, O., Ramaswamy, S., Suel, T., & Vitter, J. S. (1998). Scalable Sweeping-Based Spatial Join. *In Proceedings of 24th International Conference on Very Large Data Bases (VLDB 1998),* New York City, New York, USA, 24-27 August, 1998, (pp. 570-581). San Francisco: Morgan Kaufmann.

Ariel, O., & Avron, A. (1999). A model-theoretic approach for recovering consistent data from inconsistent knowledge bases. *Journal of Automated Reasoning, 22*(2), 263- 309.

Armendáriz-Íñigo, J. E., Juárez, J. R., González de Mendívil, J. R., Decker, H., & Muñoz-Escoí, F. D. (2007). k-Bound GSI: A Flexible Database Replication Protocol. *ACM Annual Symposium on Applied Computing,* (pp. 556-560).

Armitage, J., & Kellner, M.A. (1994). *Conceptual schema for process definitions and models*. Proceedings of the Third International Conference on Software Process, Reston, Virginia, 153–165.

Arms, W. Y., Blanchi, C., & Overly, E. A. (1997). An Architecture for Information in Digital Libraries. *D-Lib Magazine,* Feb 1997.

Arnold, S. E. (2003). Content management's new realities. *Online, 27*(1), 36-40.

Arnold-Moore, T. (1997). Automatic Generation of Amendment Legislation. *In Sixth International Conference on Artificial Intelligence and Law, ICAIL'97* (Melbourne, Victoria, Australia, 1997), (pp. 56-62).

Arnon, A., Efrat, A., Indyk, P., and Samet, H. (1999). Efficient regular data structures and algorithms for location and proximity problems. *Proceedings of the 40th Annual Symposium on Foundations of Computer Science*. New York (Oct 17-19), 160-170.

Aronov, B., Bronnimann, H., Chang, A. Y., & Chiang, Y.-J. (2003). Cost-driven octree construction schemes: An experimental study. *Proceedings of the Nineteenth Annual Symposium on Computational Geometry* (pp. 227-236).

Arruda, L., Baptista, C., & Lima, C. (2002). MEDIWEB: A mediator-based environment for data integration on the Web. *Databases and Information Systems Integration*. ICEIS, 34-41.

Artsy, Y., & Finkel, R. (1989). Designing a process migration facility. Computer, *22*(9), 47–56.

ASC (2007). The Accredited Standards Committee (ASC) X12. www.x12.org

Ashman, H. (2000). Electronic document addressing: dealing with change. *ACM Computing Surveys, 32*(3), 201-212.

Aslinger, A., & Son, S.H. (2005). *Efficient replication control in distributed real-time databases.* Proceedings of the 3rd ACS/IEEE International Conference on Computer System and Applications.

Asprey, L., & Middleton, M. (2003). *Integrative document and content management: strategies for exploiting enterprise knowledge.* Hershey, PA: Idea Group.

Assfalg, J., Kriegel, H., Kroger, P., Kunath, P., Pryakhin, A., & Renz, M. (2006). Threshold similarity queries in large time series databases. *Proceedings of IEEE ICDE*, 149. Atlanta, Georgia, USA.

ASSO, Project home page. http://www.info.fundp.ac.be/asso/. Last access on June 2008.

Association of GIS Laboratories in Europe (AGILE). (2004). Tutorial & workshop interoperability for geo-information. *7th AGILE Conference on Geographic Information Science.* Retrieved from http://agile2004.iacm.forth.gr/

Atallah, M., Elmagarmid, A., Ibrahim, M., Bertino, E., & Verykios, V. (1999). Disclosure limitation of sensitive rules. *Proceedings of the 1999 Workshop on Knowledge and Data Engineering Exchange* (pp. 45-52).

Atkinson, S., Bailes, P. A., Chapman, M., Chilvers, M., & Peake, I. (1998). A re-engineering evaluation of software refinery: Architecture, process and technology.

Au, W., Chan, K. C., Wong, A., & Wang, Y. (2005). Attribute of grouping, selection, and classification of gene expression data. *IEEE/ACM Transactions on Computational Biology and Bioinformatics, 2*(2), 83-101.

Aufrecht, S. E., & Bun, L. S. (1995). Reform with Chinese Characteristics: The Context of Chinese Civil Service Reform. *Public Administration Review, 55*, 175-183.

Aumueller, D., Do, H., Massmann, S., & Rahm, E. (2005). Schema and ontology matching with COMA++. *Proceedings of SIGMOD.*

Avan Alstyne, M., Brynjolfsson, E., & Madnick, S. E. (1994). *Why not One Big Database? Principles for Data Ownership.* Sloan School of Management, Technical Report WP 3695-94. Cambridge, MA: Massachusetts Institute of Technology.

Avnur, R., & Hellerstein, J. (2000). Eddies: Continuously adaptive query processing. *Proceedings of the International Conference on Management of Data (SIGMOD), 2000,* (pp. 261-272).

Baader, F., & Nutt W. (2002). Basic Description Logics. In F. Baader, D. Calvanese, D. L. McGuinness, D. Nardi, & P. F. Patel-Schneider (Eds.), *The Description Logic Handbook* (pp. 47-100). Cambridge University Press.

Baader, F., Calvanese, D., McGuinness D., Nardi, D., & Patel-Schneider, P. (2003). *The description logic handbook*. Cambridge, UK: Cambridge University Press.

Baader, F., Calvanese, D., McGuinness, D. L., Nardi, D., & Patel-Schneider, P. F. (2003). *The description logic handbook: Theory, implementation, and applications.* Cambridge University Press.

Baader, F., Calvanese, D., McGuinness, D., Nardi, D., & Patel-Schneider, P. F., (eds.) (2002). The description logic handbook: Theory, implementation and applications. Cambridge: University Press.

Babcock, B., Babu, S., Datar, M., Motawani, R., & Widom, J. (2002). Models and issues in data stream systems. *Proceedings of the 21st ACM International Symposium on Principles of Database Systems* (pp. 1-16).

Babcock, B., Chaudhuri, S., & Das, G. (2003). Dynamic sample selection for approximate query answers. *Proceedings of the 2003 ACM International Conference on Management of Data* (pp. 539-550).

Babu, S., & Widom, J. (2001). Continuous queries over data streams. *SIGMOD Record, 30*(3), 109-120.

Baclawski, K., Kokar, M., Waldinger, R., & Kogut, P. (2002). Consistency checking of semantic web ontologies. *Proceedings of the First International Semantic Web Conference 2002 (ISWC'02), Lecture Notes in Computer Science, 2342* (pp. 454-459).

Badia, A. (2006). Relational, object-oriented and object-relational data models. In: Rivero, L. C., Doorn, J. H., & Ferragine, V. E. (Eds.). *Encyclopedia of database technologies and applications* (pp. 530-535). Idea Group Reference.

Baeza Yates, R. (1997). Searching: An algorithm tour. In A. Kent & Williams (Eds.), *Encyclopedia of computer science* (Vol. 37, pp. 331-359). Marcel Dekker Inc.

Bagai, R., & Sunderraman, R. (1995). A paraconsistent relational data model. *International Journal of Computer Mathematics, 55*(3).

Bagai, R., & Sunderraman, R. (1996). Bottom-up computation of the fitting model for general deductive databases. *Journal of Intelligent Information Systems, 6*(1), 59–75.

Bagui, S., & Earp, R. (2003). *Database Design Using Entity-Relationship Diagrams.* Auerbach Publications, Boca Raton, Florida: CRC Press.

Bailey, J., Georgeff, M. P., Kemp, D. B., Kinny, D., & Ramamohanarao, K. (1995). Active Databases and Agent Systems - A Comparison. *Rules in Database Systems,* (pp. 342-356).

Balaton, Z., Kacsuk, P., & Podhorszki, N. (2001). *Use cases and the proposed grid monitoring architecture* (Tech. Rep. No. LDS-1/2001). Hungary: Hungarian Academy of Sciences, Computer and Automation Research Institute.

Balmin, A., & Papakonstantinou, Y. (2005). Storing and Querying XML Data Using Denormalized Relational Databases. *The VLDB Journal, 14*(1), 30-49.

Banaei-Kashani, F., & Shahabi, C. (2003a). Criticality-based analysis and design of unstructured peer-to-peer networks as complex systems. *Proceedings of the Third International Workshop on Global and Peer-to-Peer Computing (GP2PC) in conjunction with CCGrid, 2003,* 351-359.

Banaei-Kashani, F., & Shahabi, C. (2003b). Searchable querical data networks. In *Lecture notes in computer science, 2944,* 17-32. Berlin, Germany: Springer-Verlag.

Banerjee, J., Kim, H.-J., et al. (1986). Schema Evolution in Object-Oriented Persistent Databases. *XP/7.52 Workshop on Database Theory*, University of Texas at Austin, TX, USA.

Banerjee, J., Kim, W., Kim, H.-J., & Korth, H.F. (1987). Semantics and implementation of schema evolution in object-oriented databases. *Proceedings of the 1987 ACM SIGMOD International Conference on Management of data,* 311-322. San Francisco, CA.

Banerjee, J., Kim, W., Kim, H.J., & Korth, H.F. (1987). *Semantics and implementation of schema evolution in object-oriented databases.* Proceedings of the SIGMOD'87, San Francisco, California, 311–322.

Baral, C., Kraus, S., & Minker, J. (1991). Combining multiple knowledge bases. *IEEE-TKDE, 3*(2), 208-220.

Baralis, E., & Widom, J. (2000, September). An algebraic approach to static analysis of active database rules. *ACM Transactions on Database Systems, 25*(3), 269-332.

Baralis, E., Paraboschi, S., & Teniente, E. (1997). Materialized views selection in a multidimensional database. *23rd International Conference on Very Large Data Bases (VLDB 1997)* (pp. 156-165).

Barbarà, D. (1999). Mobile Computing and Databases – A Survey. *IEEE Transactions on Knowledge and Data Engineering,* IEEE, 11(1), 108-117.

Barbara, D., & Garcia-Molina, H. (1992). *The demarcation protocol: A technique for maintaining linear arithmetic constraints in distributed database systems.* Proceedings of the Conference on Extending Database Technology, LNCS 580, Austria, 373–388.

Barbosa, L., & Freire, J. (2004). Siphoning hidden-web data through keyword-based interfaces. *19th Brazilian Symposium on Data Base (SBBD'04),* 309-321.

Barbosa, L., & Freire, J. (2005). Searching for hidden-web databases. *8th International Workshop on the Web and Databases (WebDB'05),* 1-6.

Barghouti, N., & Kaiser, G. (1991). Concurrency control in advanced database applications. *ACM Computing Surveys, 23*(3), 269-317.

Baril, X., & Bellahsene, Z. (2003). Selection of materialized views: A cost-based approach. In *Lecture notes in computer science: Vol. 2681. 15th International Conference on Advanced Information Systems Engineering (CAiSE 2003)* (pp. 665-680). Berlin, Germany: Springer.

Barnett, V., & Lewis, T. (1994). *Outliers in statistical data* (3rd ed.). New York: John Wiley.

Baru, C. (1998). Features and requirements for an XML view definition language: Lessons from XML information mediation. In *W3CWorkshop on Query Language (QL´98).* Boston.

Bar-Yam, Y. (1997). *Dynamics of complex systems.* New York, NY: Westview Press.

Barzilay, R., & McKeown, K. (2005). Sentence fusion for multidocument news summarization. *Computational Linguistics, 31*(3), 297-328.

Basch, J., Guibas, L. J., & Hershberger, J. (1997). Data structures for mobile data. *Proc. of the ACMSIAM Symposium on Discrete Algorithms (SODA'1997),* 747-756.

Basic, (1996). BasicScript 2.25 Language Reference, Summit Software. http://www.cardiff.com/CSdownload/misc/t1057.pdf.

Basile, C., Calanna, S., Nitto, E., Fuggetta, A., & Gemo, M. (1996). *Mechanisms and policies for federated PSEEs: Basic concepts and open issues.* Proceedings of the 5th European Workshop on Software Process Technology, LNCS, 1149, 86–91.

Basu, S., Bilenko, M., & Mooney, R. (2003). *Comparing and unifying search-based and similarity-based approaches to semi-supervised clustering.* Paper presented at ICML'03: the twentieth International Conference on Machine Learning, 2003.

Basu, S.; Bilenko, M., & Mooney, R. J. (2004). *A probabilistic framework for semi-supervised clustering.* Paper presented at KDD '04: the tenth ACM SIGKDD international conference on Knowledge discovery and data mining, ACM, 2004, 59-68.

Batini, C., Ceri, S., & Navathe, S. B. (1992). Conceptual Database Design: An Entity Approach. Redwood City, CA: Benjamin Cummings.

Bauer, A., Hümmer, W., & Lehner, W. (2000). *An alternative relational OLAP modeling approach.* Proceedings of the 2nd International Conference on Data Warehousing and Knowledge Discovery, 189–198.

Baxevanis, A. D., & Ouellette, F. (2005). *Bioinformatics: A practical guide to the analysis of genes and proteins.* New York: Wiley-Interscience.

Baxter, I., & Mehlich, M. (2000). Reverse engineering is reverse forward engineering. *Science of Computer Programming, 36,* 131-147.

Bayardo Jr., R. J., Bohrer, W., Brice, R., Cichocki, A., Fowler, J., Helal, A., Kashyap, V., Ksiezyk, T., Martin, G., Nodine, M., Rashid, M., Rusinkiewicz, M., Shea, R., Unnikrishnan, C., Unruh, A., & Woelk, D. (1997). InfoSleuth: Agent-based semantic integration of information in open and dynamic environments. *Proceedings ACM SIGMOD International Conference on Management of Data,* 195-206.

Bayardo, R. J., & Agrawal, R. (2005). Data privacy through optimal *k*-anonymization. *Proceedings of the 21st International Conference on Data Engineering (ICDE'05)* (pp. 217-228).

BEA (2007). *BEA Systems, Inc.* www.bea.com.

Bebel, B., Eder, J., Koncilia, C., Morzy, T., & Wrembel, R. (2004). *Creation and management of versions in multiversion data warehouse.* Proceedings of the 19th ACM Symposium on Applied Computing (SAC 04), Nicosia, Cyprus, 717–723.

Bechhofer, S., van Harmelen, F., Hendler, J., Horrocks, I., McGuinness, D. L., Patel-Schneider, P. F., & Stein, L.A., (eds) (2004). *OWL Web ontology language reference – W3C recommendation 10 February 2004.* W3C (http://www.w3.org/TR/owl-ref/).

Beck, T., Clarke, G., Groff, A., Keefer, P., & Walsh, P. (2001). New Tools in Comparative Political Economy:

the Database of Political Institutions. *The World Bank Economic Review, 15*(1), 165-176.

Becker, S. A. (2003). *Effective databases for text & document management.* Hershey PA: IRM Press.

Beckett, D. (Ed.). (2004). RDF/XML syntax specification. *W3C recommendation.* Retrieved from http://www.w3.org/TR/2004/REC-rdf-syntax-grammar-20040210

Beckett, D., (ed.) (2004). *RDF/XML syntax specification (Revised) – W3C recommendation 10 February 2004.* W3C (http://www.w3.org/TR/rdf-syntax-grammar/).

Beckmann, N., Kriegel, H. P., Schneider, R., & Seeger, B. (1990). The R*-tree: An efficient and robust access method for points and rectangles. *Proceedings of SIGMOD Conference* (pp. 322-331).

Bédard, Y., Rivest, S., & Proulx, M. (2007). Spatial on-line analytical processing (SOLAP): Concepts, architectures and solutions from a geomatics engineering perspective. In R. Wrembel, & C. Koncilia (Eds.), *Data warehouses and OLAP: Concepts, architectures and solutions* (pp, 298–319). Hershey, PA: Idea Group Publishing.

Behm, A., Geppert, A., & Diettrich, K. R. (1997). *On the migration of relational schemas and data to object-oriented database systems.* Proceedings of Re-Technologies in Information Systems, Klagenfurt, Austria.

Beil, F., Ester, M., & Xu, X. (2002). *Frequent term-based text clustering.* Paper presented at the eighth ACM SIGKDD international conference on Knowledge Discovery and Data Mining.

Bellahsène, Z. (1996). *View mechanism for schema evolution*. Proceedings of the BNCOD-14. LNCS, Edinburgh, UK, 1094, 18–35.

Bellahsene, Z. (2002). Schema evolution in data warehouses. *Knowledge and Information Systems, 4*(3), 283–304.

Bellatreche, L., & Boukhalfa, K. (2005). An Evolutionary Approach to Schema Partitioning Selection in a Data Warehouse. *7th International Conference on Data Warehousing and Knowledge Discovery,* (pp. 115-125).

Bellatreche, L., Boukhalfa, K., & Abdalla, H. I. (2006). SAGA: A Combination of Genetic and Simulated Annealing Algorithms for Physical Data Warehouse Design. In 23rd British National Conference on Databases (BNCOD'06), (pp. 212-219).

Bellatreche, L., Giacometti, A., Marcel, P., Mouloudi, H., & Laurent, D. (2005). *A personalization framework for OLAP queries.* Proceedings of the 8th ACM International Workshop on Data Warehousing and OLAP (DOLAP 05), Bremen, Germany, 9–18.

Bellatreche, L., Karlapalem, K., & Schneider, M. (2000). On efficient storage space distribution among materialized views and indices in data warehousing environments. *9th International Conference on Information and Knowledge Management (CIKM 2000)* (pp. 397-404).

Bellatreche, L., Karlapalem, K., & Simonet, A. (2000). Algorithms and Support for Horizontal Class Partitioning in Object-Oriented Databases. *Distributed and Parallel Databases, 8*(2), 155-179.

Bellatreche, L., Karlapalem, K., Mukesh, K. M., & Mohania, S. M. (2000). What Can Partitioning Do for Your Data Warehouses and Data Marts? *Proceedings on International Database Engineering and Applications Symposium* (IDEAS'2000), (pp. 437-446).

Belussi, A., & Faloutsos, C. (1998). Self-Spacial Join Selectivity Estimation Using Fractal Concepts. *ACM Transactions on Information Systems, 16*(2), 161-201.

Benatallah, B. (1996). Un *Compromis: Modification et Versionnement du Schéma. 12 èmes Journées Bases de Données Avancées,* 27-30 Août 1996, France INRIA.

Bench-Capon, T. (2001). The role of ontologies in the verification & validation of knowledge-based systems. *International Journal of Artificial Intelligence, 16,* 377-390.

Beneventano, D., Bergamaschi, S., Guerra, F., & Vincini, M. (2001). The MOMIS approach to information integration. Proceedings of the 3rd International Conference on Enterprise Information Systems, Setubal, Portugal, 91.

Benítez-Guerrero, E., & Hernández-López, A.R. (2006). *The mine SP operator for mining sequential patterns in inductive databases.* Proceedings of the MICAI 2006: Advances in Artificial Intelligence, 5th Mexican International Conference on Artificial Intelligence, 4293, 684–694.

Benítez-Guerrero, E., Collet, C., & Adiba, M. (2003). The WHES approach to data warehouse evolution. *Digital Journal e-Gnosis,* 1665–5745. Retrieved from http://www.e-gnosis.udg.mx

Benitez-Guerrero, E.I., Collet, C., & Adiba, M. (2004). The WHES approach to data warehouse evolution. *Electronic Journal e-Gnosis, 2*.

Benjelloun, O., Sarma, A. D., Halevy, A., Theobald, M., & Widom, J. (2008). Databases with Uncertainty and Lineage. *VLDB Journal, 17*(2), 243-264.

Ben-Shaul, I.Z., & Kaiser, G.E. (1995). A paradigm for decentralized process modeling. Kluwer Academic Publisher.

Ben-Shaul, I.Z., & Kaiser, G.E. (1998). Federation of process-centered environments: The Oz experience. *Automated Software Engineering, 5*(1).

Bentley, J. L. (1975). Multidimensional binary search trees used for associative searching. *Communications of the ACM, 18*(9), 509-517.

Berchtold, S., Keim, D. A., & Kriegel, H.-P. (1996). The X-tree: An index structure for high-dimensional data. *Proceedings of 22nd VLDM Conference* (pp. 28-39).

Berenson, H., Bernstein, P. A., Gray, J., Melton, J., O'Neil, E. J., & O'Neil, P. E. (1995). A Critique of ANSI SQL Isolation Levels. *ACM SIGMOD Conference,* (pp. 1-10).

Bergman, M. (2001). The deep Web: surfacing hidden value. *Journal of Electronic Publishing, 7*(1).

Berkhin, P. (2002). *Survey of Clustering Data Mining Techniques.* Accrue Software, San Jose, CA, USA. URL: http://citeseer.ist.psu.edu/berkhin02survey.html.

Bernabé-Gisbert, J. M., Salinas-Monteagudo, R., Irún-Briz, L., & Muñoz-Escoí, F. D. (2006). Managing Multiple Isolation Levels in Middleware Database Replication Protocols. *Lecture Notes in Computer Science, 4330,* 710-723.

Bernard, L., Craglia, M., Gould, M., & Kuhn, W. (2005). *Towards an SDI research agenda*. Proceedings of the EC & GIS Workshop, Sardinia.

Bernard, L., Einspanier, U., Haubrock, S., Hübner, S., Kuhn, W., Lessing, R., et al. (2003). Ontologies for intelligent search and semantic translation in spatial data infrastructures. *Photogram-metrie, Fernerkundung, Geoinformation 2003* (pp. 451-462).

Berners-Lee, T., Hall, W., Hendler, J. A., O'Hara, K., Shadbolt, N., & Weitzner, D. J. (2006). A framework for Web science. *Foundations and Trends in Web Science, 1,* 1-130.

Berners-Lee, T., Hendler, J. A., & Lassila, O. (2001). The Semantic Web. *Scientific American, 284*(5), 34-43.

Berners-Lee, T., Hendler, J., & Lassila, O. (2001). The semantic Web. *Scientific American*.

Bernoit, D. G. (2005). Automatic diagnosis of performance problem in database management systems. *In Proceedings of the Second International Conference on Autonomic Computing (ICAC'05),* 326-327.

Bernstein, A., Provost, F., & Hill, S. (2005). Towards Intelligent Assistance for the Data Mining Process:An Ontology-based Approach for Cost/Sensitive Classification. *In IEEE Transactions on Knowledge and Data Engineering, 17*(4), 503-518.

Bernstein, P. A., & Goodman, N. (1981). Concurrency Control for Distributed Database Systems. *ACM Computing Surveys, 13*(2), 185-221.

Bernstein, P. A., Giunchiglia, F., Kementsietsidis, A., Mylopulos, J., Serafini, L., & Zaihrayen, I. (2002). Data management for peer-to-peer computing: A vision. *WebDB* (pp. 89-94).

Bernstein, P.A., & Blaustein, B.T. (1981). *A simplification algorithm for integrity assertions and concrete views.* Proceedings of the 5th International Computer Software and Applications Conference, 90–99.

Berson, A., & Smith, S. J. (1997). *Data warehousing, data mining, and OLAP.* New York: McGraw-Hill.

Bertino, E., Catania, B., & Wang, W.Q. (2004). *XJoin index: Indexing XML data for efficient handling of branching path expressions.* Proceedings of the 20th International Conference on Data Engineering (ICDE 04), Boston, Massachusetts, 828.

Bertino, E., Catania, B., & Zarri, G. P. (2001). *Intelligent database systems*. London: Addison-Wesley and ACM Press.

Bertossi, L. & Bravo, L. (2004). Query answering in peer-to-peer data exchange systems. *EDBT Workshop* (pp. 476-485).

Bertossi, L. E., & Schind, C. (2004). Database repairs and analytic tableaux. *Annals of Mathematics and Artificial Intelligence, 40*(1/2), 5-35.

Betini, C., Jajodia, S., & Wang, S. (1988). Time granularities in databases, data mining and temporal reasoning. Springer Verlag.

Bettini, C., Dyreson, C. E., Evans, W. S., Snodgrass, R. T., & Wang, X. S. (1998). A Glossary of Time Granularity Concepts. *Lecture Notes in Computer Science, 1399, 406-413.* Springer-Verlag Publishing.

Beyer, K. S., Chamberlin, D. D., Colby, L. S., Özcan, F., Pirahesh, H., & Xu, Y. (2005). Extending XQuery for analytics. *ACM SIGMOD International Conference on Management of Data (SIGMOD 2005)* (pp. 503-514).

Beyer, K.S., Chamberlin, D.D., Colby, L.S., Özcan, F., Pirahesh, H., & Xu, Y. (2005). *Extending XQuery for analytics.* Proceedings of the 2005 ACM SIGMOD International Conference on Management of Data (SIGMOD 05), Baltimore, Maryland, 503–514.

Bharadvaj, H., Joshi, A., & Auephanwiriyakyl, S. (1998). An Active Transcoding Proxy to Support Mobile Web Access. *Proc. of the 17th IEEE Symp. on Reliable Distributed Systems,* (pp. 118-123).

Bianchi, A., Caivano, D., & Visaggio, G. (2000). Method and process for iterative reengineering of data in a legacy system. *WCRE*. Washington, DC.

Bielawski, L., & Boyle, J. (1997). *Electronic document management systems: a user centered approach for creating, distributing and managing online publications.* Upper Saddle River, NJ: Prentice-Hall.

Billard, L., & Diday, E. (2007). *Symbolic Data Analysis: Conceptual Statistics and Data Mining .Wiley Series in Computational Statistics.* Chichester, West Sussex, England: John Wiley & Sons Ltd.

Bimonte, S., Tchounikine, A., & Miquel, M. (2005). *Towards a spatial multidimensional model.* Proceedings of the 8th ACM International Workshop on Data Warehousing and OLAP, 39–46.

Bimonte, S., Tchounikine, A., & Miquel, M. (2005). Towards a spatial multidimensional model. *Proceedings of the 8th International Workshop on Data Warehousing and OLAP* (pp. 39-46).

Bin, J., & Itzhak, O. (2006). Spatial topology and its structural analysis based on the concept of simplicial complex. *9th AGILE Conference on Geographic Information Science*, Visegrád, Hungary (pp. 204-212).

Biosensornet: *Autonomic Biosensor Networks for Pervasive Healthcare*,http://www.doc.ic.ac.uk/ mss/Biosensornet.htm 2007

Birman, K. P., & Joseph, T. A. (1987). Reliable Communication in the Presence of Failures. *ACM Transactions on Computer Systems, 5*(1), 47-76.

Biron, P. V., & Malhotra, A. (2004). *XML Schema Part 2: Datatypes (Second Edition).* W3C Recommendation. www.w3.org/TR/xmlschema-2/.

Biron, P. V., & Malhotra, A. (eds.) (2001). *XML schema part 2: Datatypes – W3C recommendation 02 May 2001.* W3C (http://www.w3.org/TR/xmlschema-2/).

Bisbal, J., Lawless, D., Wu, B., & Grimson, J. (1999). Legacy information systems: Issues and directions. *IEEE Software, 16*(5), 103-111.

Bishr, Y. A. (1998). Overcoming the semantic and other barriers to GIS interoperability. *International Journal of Geographic Information Science, 12*(4), 299-314.

Bishr, Y. A., Pundt, H., Kuhn, W., & Radwan, M. (1999). Probing the concept of information communities: A first step toward semantic interoperability. In M. Goodchild, M. Egenhofer, R. Fegeas, & C. Kottman (Eds.), *Interoperating geographic information systems* (pp. 55-69). Kluwer Academic.

Bizer, C. (2003). *D2R MAP: A database to RDF mapping language.* Poster at the 12th World Wide Web Conference, Budapest, Hungary.

Blaha, M. R., & Premerlani, W. J. (1995). Observed idiosyncrasies of relational database designs. *Proceedings of the 2nd IEEE Working Conference on Reverse Engineering.*

Blanchard, J. et al. (2005). Assessing Rule Interestingness with a Probabilistic Measure of Deviation from Equilibrium. *Proceedings of the 11th International Symposium on Applied Stochastic Models and Data Analysis*, France, (pp. 191-200).

Blaschka, M., Sapia, C., & Höfling, G. (1999). *On schema evolution in multidimensional databases.* Proceedings of the DaWak'99, Florence, Italy, 153–164.

Blaustein, B.T. (1981). Enforcing database assertions: Techniques and applications [doctoral thesis]. Harvard University.

Bliss, A. (2001). Rhetorical structures for multilingual and multicultural students. In C. G. Panetta (Ed.), *Contrastive rhetoric revisited and redefined* (pp. 15-30). Mahwah, NJ: Lawrence Erlbaum Associates.

Bliujute, R., Saltenis, S., Slivinskas, G., & Jensen, C. (1998). *Systematic change management in dimensional data warehousing.* Proceedings of the 3rd International Baltic Workshop on Databases and Information Systems, Riga, Latvia, 27–41.

Bloch, I., Hunter, A., Appriou, A., Ayoun, A., Benferhat, S., Besnard, P., Cholvy, L., Cooke, R., Cuppens, F., Dubois, D., Fargier, H., Grabisch, M., Kruse, R., Lang, J., Moral, S., Prade, H., Saffiotti, A., Smets, P., & Sossai, C. (2001). Fusion: General concepts and characteristics, *International Journal of Intelligent Systems, 16*(10), 1107-1134.

Blum, A., Dwork, C., McSherry, F., & Nissim, K. (2005). *Practical privacy: The SuLQ framework.* Proceedings of ACM Symposium on Principles of Database Systems (PODS'05), Baltimore.

Blumberg, R., & Arte, S. (2003). The problem with unstructured data. *DM Review* (February).

Boag, S., Chamberlin, D., Fernandez, M., Florescu, D., Robie, J., & Siméon, J. (2006). XQuery 1.0: An XML query language [W3C working draft]. Retrieved September 20, 2006, from http://www.w3.org/TR/xquery/

Bobineau, C., Bouganim, L., Pucheral, P., & Valduriez, P. (2000). PicoDBMS: Scaling Down Database Techniques for the Smartcard. *Proc. of the 26th Int. Conf. on Very Large Data Bases,* (pp. 11-20).

Bock, H., & Diday, E. (2000). *Analysis of Symbolic Data.* Studies in Classification, Data Analysis and Knowledge Organization. Heidelberg, Germany: Springer Verlag-Berlin.

Bodart, F., Patel, A., Sim, M., & Weber, R. (2001). Should optional properties be used in conceptual modeling? A theory and three empirical tests. Information Systems Research, 12(4), 384-405.

Body, M., Miquel, M., Bédard, Y., & Tchounikine, A. (2002). *A multidimensional and multiversion structure for OLAP applications.* Proceedings of the 5th ACM International Workshop on Data Warehousing and OLAP (DOLAP 02), McLean, Virginia, 1–6.

Body, M., Miquel, M., Bédard, Y., & Tchounikine, A. (2003). *Handling evolutions in multidimensional structures.* Proceedings of the 19th International Conference on Data Engineering (ICDE 03), Bangalore, India, 581–591.

Böehnlein, M., & Ulbrich-vom Ende, A. (1999). Deriving initial data warehouses structures from the conceptual data models of the underlying operational information systems. *Proceedings of the 2nd ACM International Workshop on Data Warehousing and OLAP,* 15-21.

Böehnlein, M., & Ulbrich-vom Ende, A. (2000). Business process oriented development of data warehouse structures. *Proceedings of Data Warehousing,* 3-21.

Bogorny, V. et al. (2006). Towards elimination of well known geographic domain patterns in spatial association rule mining. *In Third IEEE International Conference on Intelligent Systems,* (pp. 532-537).

Bohlen, M. H., Snodgrass, R. T., & Soo, M. D. (1996). Coalescing in temporal databases. *In Proceedings of International Conference on Very Large Databases.*

Böhm, C. (2000). A cost model for query processing in high dimensional data spaces. *ACM Transactions on Database Systems, 25*(2), 129-178.

Bohm, C., & Jacopini, G. (1966). Flow diagrams, Turing machines, and languages with only two formation rules. *CACM, 9*(5), 266.

Böhm, C., & Krebs, F. (2002). High Performance Data Mining Using the Nearest Neighbor Join. *In Proceedings of the 2002 IEEE International Conference on Data Mining (ICDM 2002),* Maebashi City, Japan, 9-12 December, 2002, (pp. 43-50), IEEE Computer Society Press.

Boiko, B. (2002). *Content management bible.* New York: Hungry Minds.

Boley, H., Tabet, S., & Wagner, G. (2001). Design rationale of RuleML: A markup language for Semantic Web rules. In *Proceedings of SWWS'01, The First Semantic Web Working Symposium.* Stanford: Stanford University.

Bolshakova, N., & Azuaje, F. (2006). Estimating the number of clusters in DNA microarray data. *Methods of Information in Medicine, 45*(2), 153-157.

Bonifati, A., & Cuzzocrea, A. (2006). Storing and retrieving XPath fragments in structured P2P networks. *Data & Knowledge Engineering,* 59(2), 247-269.

Bonifati, A., Cattaneo, F., Ceri, S., Fuggetta, A., & Paraboschi, S. (2001). Designing data marts for data warehouses. *ACM Transactions on Software Engineering and Methodology, 10*(4), 452-483.

Bonifati, A., Cuzzocrea, A., & Zinno, B. (2006). *Fragmenting XML documents via structure constraints.* Proceedings of the 10th East European Conference on Advances in Databases and Information Systems (ADBIS 06), Thessalonica, Greece, 17–29.

Bonifati, A., Cuzzocrea, A., Matrangolo, U., & Jain, M. (2004). XPath lookup queries in P2P networks. *Proceedings of the 6th ACM International Workshop on Web Information and Data Management, in conjunction with the 13th ACM International Conference on Information and Knowledge Management* (pp. 48-55).

Bonnet, P., Gehrke, J. E., & Seshadri, P. (2001). Towards Sensor Database Systems. *Proceedings of the Second International Conference on Mobile Data Management,* Hong Kong, January 2001.

Booch, G., Rumbaugh, J., & Jacobson, I. (1998). *The unified modeling language user guide*. Addison-Wesley Longman.

Borges, K., Davis, C., & Laender, A. (2001). OMT-G: An object-oriented data model for geographic applications. *Geoinformatica, 5,* 221–260.

Borgida, A. (1994). On the relationship between description logic and predicate logic queries. *Proc of Conference on Information and Knowledge Management,* Maryland, USA.

Borgida, A., & Williamson, K. E. (1985). Accommodating Exceptions in Databases, and Refining the Schema by Learning from them. *11th International Conference on Very Large Data Bases VLDB'85,* Stockholm, Sweden, Morgan Kaufmann

Borland, J. & Kanellos, M. (2004, July 28). South Korea leads the way. *News.Com.* Retrieved November 2, 2005, from http://news.com.com/South+Korea+leads+the+way/2009-1034_3-5261393.html

Borstlap, G. (2006). *Understanding the technical barriers of retargeting ISAM to RDBMS.* Retrieved from *http://www.anubex.com/anugenio!technicalbarriers1.asp*

Bosc, P., Dubois, D., & Prade, H. (1994). Fuzzy functional dependencies - An overview and a critical discussion. *Proc. of 3rd IEEE Internat. Conf. on Fuzzy Systems,* 325-330. Orlando.

Boskovitz, A., Goré, R., & Hegland, M. (2003). A logical formalisation of the Fellegi-Holt method of data cleaning. *International Conference on Intelligent Data Analysis (IDA 2003), Lecture Notes in Computer Science, 2810* (pp. 554-565).

Botea, V., Mallett, D., Nascimento, M. A., & Sander, J. (2007). PIST: An efficient and practical indexing technique for historical spatio-temporal point data. *Geoinformatica*, onlinefirst.

Botev, C., Amer-Yaiha, S., & Shanmugasundaram, J. (2004). A TexQuery-based XML full-text search engine. In *Proc. of ACM SIGMOD* (pp. 943–944), Paris, France.

Botta, M., Boulicaut, F.C., Masson, C., & Meo, R. (2002). *A comparison between query languages for the extraction of association rules.* Proceedings of the 4th International Conference on Data Warehousing and Knowledge Discovery (DaWaK'2000), 1–10.

Boucelma, O., & Colonna, F. M. (2005). Mediation for online geoservices. In *Lecture notes in computer science: Vol. 3428. Web and wireless geographical information systems, interoperability and security in W2GIS* (pp. 81-93). Berlin, Germany: Springer.

Boudjeloud, L., & Poulet, F. (2005). Visual interactive evolutionary algorithm for high dimensional data clustering and outlier detection. *Proceedings of the 9th Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining (PAKDD'05)* (pp. 426-431).

Boufares, F., & Bennaceur, H. (2004). Consistency problems in ER-schemas for database systems. *Information Sciences, 163,* 263-274.

Boufares, F., & Kraïem, N. (2001). A new tool to analyze ER-schemas. *Proceedings of the Second Asia-Pacific Conference on Quality Software* (pp. 302-307).

Boulcema, O., Essid, E., & Lacroix, Z. (2002). *A WFS-based mediation system for GIS interoperability.* Proceedings of the 10th ACM International Symposium on Advances in Geographic Information Systems, 23–28.

Boulicaut, J.F., Klemettinen, M., & Mannila, H. (1999). *Modeling KDD processes within the inductive database framework.* Proceedings of the First International Confer-

ence on Data Warehousing and Knowledge Discovery (DaWaK'99), 293–302).

Boulos, J., & Karakashian, S. (2006). *A new design for a native XML storage and indexing manager.* Proceedings of the 10th International Conference on Extending Database Technology (EDBT 06), Munich, Germany, 3896, 755–772.

Bounif, H., & Spaccapietra, S. et al. (2006). Requirements Ontology and Multi-representation Strategy for Database Schema Evolution., Seoul, Korea, 2006 *VLDB Workshop on Ontologies-based techniques for DataBases and Information Systems.*

Bouquet, P., Giunchiglia F, van Harmelen F., Serafini, L., & Stuckenschmidt, H. (2003). C-OWL: Contextualizing ontologies. *Proceedings of the 2nd International Semantic Web Conference 2003 (ISWC'03), Lecture Notes in Computer Science , 2870* (pp. 164-179).

Bouzeghoub, M., Fabret, F., & Matulovic-Broqué, M. (1999). *Modeling the data warehouse refreshment process as a workflow application.* Proceedings of the International Workshop on Design and Management of Data Warehouses (DMDW 99), Heidelberg, Germany, 19, 6.

Boyd, M., Kittivoravitkul, S., Lazanitis, C., McBrien, P.J., & Rizopoulos, N. (2004). *AutoMed: A BAV data integration system for heterogeneous data sources.* Proceedings of the CAiSE04, 3084, 82–97.

Bozkaya, T., Yazdani, N., & Ozsoyoglu, M. (1997). Matching and indexing sequences of different lengths. *Proceedings of CIKM*, 128-135. Las Vegas, NV, USA.

BPEL (2007). OASIS Web Services Business Process Execution Language. www.oasis-open.org/committees/tc_home.php?wg_abbrev=wsbpel

Bravo, L., & Bertossi, L. (2006). Semantically correct query answers in the presence of null values. *Proceedings of EDBT WS on Inconsistency and Incompleteness in Databases (IIDB)* (pp. 336-357).

Bray, T., Paoli, J., Sperberg-McQueen, C. M., Maler, E., & Yergeau, F. (eds.) (2004). *Extensible markup language (XML) 1.0 (Third Edition) – W3C recommendation 04 February 2004.* W3C (http://www.w3.org/TR/REC-xml/).

Bray, T., Paoli, J., Sperberg-McQueen, C. M., Maler, E., & Yergeau, F. (2006). *Extensible Markup Language (XML) 1.0 (Fourth Edition).* W3C Recommendation. http://www.w3.org/TR/REC-xml/.

Bray, T., Paoli, J., Sperberg-McQueen, C.M., Maler, E., & Yergeau, F. (2004). Extensible Markup Language (XML) 1.0 (third edition). World Wide Web Consortium.

Bremer, J.-M., & Gertz, M. (2003). On distributing XML repositories. *Proceedings of the 2003 ACM International Workshop on Web and Databases, in conjunction with the 2003 ACM International Conference on Management of Data* (pp. 73-78).

Breton, R. (1998). Replication Strategies for High Availability and Disaster Recovery. *IEEE Bulletin of the Technical Committee on Data Engineering, 21*(4), 38-43.

Breuning, M., Kriegel, H.-P., Ng, R., & Sander, J. (2000). LOF: Identifying density-based local outliers. *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data (SIGMOD'00)* (pp. 93-104).

Brewka, G., & Eiter, T. (1999). Preferred answer sets for extended logic programs. *AI, 109*(1-2), 297-356.

Brewka, G., Niemela, I., & Truszczynski, M. (2003). Answer set optimization. *IJCAI* (pp. 867-872).

Brezany, P. et al. (2004). GridMiner: A Framework for Knowledge Discovery on the Grid - from a Vision to Design and Implementation. *Cracow Grid Workshop,* Cracow, Poland: Springer.

Brickley, D., & Guha, R. V. (eds.) (2004). *RDF vocabulary description language 1.0: RDF schema – W3C recommendation 10 February 2004.* W3C (http://www.w3.org/TR/rdf-schema/).

Bridewell, W. et al. (2006). An Interactive environment for the Modeling on discovery of scientific knowledge. *International Journal of Human-Computer Studies, 64,* 1009-1014.

Brilingaite, A., Jensen, C. S., & Zokaite, N. (2004). Enabling routes as context in mobile services. *Proceedings of 12th ACM Symposium on Advances in Geographic Information Systems* (pp. 127-136).

Brin, S. et al. (1997). Beyond Market Baskets: Generalizing Association Rules to Correlations, *Proceedings of the 1997 ACM SIGMOD International Conference on Management of Data*, USA, (pp. 265-276).

Brinkhoff, T., Kriegel, H. P., & Seeger, B. (1993). Efficient Processing of Spatial Joins Using R-Trees. *In Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data (SIGMOD 1993),* Washington D.C., Washington, USA, 26-28 May, 1993, (pp. 237-246), ACM Press.

Brinkhoff, T., Kriegel, H. P., Schneider, R., & Seeger, B. (1994). Multi-Step Processing of Spatial Joins. *In Proceedings of the 1994 ACM SIGMOD International Conference on Management of Data (SIGMOD 1994),* Minneapolis, Minnesota, USA, 24-27 May, 1994, (pp. 197-208), ACM Press.

Brisson, L., & Collard, M. (2007). *An Ontology Driven Data Mining Process.* Research report of University of Nice, France. Retrieved from http://www.i3s.unice.fr/~mcollard/KEOPS.pdf on July 2007.

Brodie, M. L., & Stonebraker, M. (1995). *Migrating legacy systems: Gateways, interfaces, and the incremental approach.* Morgan Kaufmann.

Brodie, M., & Stonebraker, M. (1995). *Migrating legacy systems.* Morgan Kaufmann.

Brooks, C., Cooke, J., & Vassileva, J. (2003). *In Proceedings of the 3rd. IEEE International Conference on Advanced Learning Technologies (ICALT'03)* (pp. 296-297). Athens, Greece: IEEE Computer Society.

Brown, K. P., Carey, M. J., & Livny, M. (1993). Managing memory to meet multiclass workload response time goals. In Proceedings of 19th VLDB, 328-341. Dublin, Ireland.

Brown, K. P., Carey, M. J., & Livny, M. (1996) Goal-oriented buffer management revisited. *In Proceedings. Of the ACM SIGMOD International Conference on Management of Data,* 353- 364.

Brun, M., Sima, C., Hua, J., Lowey, J., Carroll, B., Suh, E., & Dougherty, E. R. (2007). *Model-based evaluation of clustering validation measures, 40*, 807-824. Pattern Recognition, Elsevier Science Inc.

Bruno, N., Koudas, N., & Srivastava, D. (2002). Holistic twig hoins: Optimal XML pattern matching. In *Proc. SIGMOD Int. Conf. on Management of Data,* 310-321. Madison, Wisconsin.

Bry, F. (1997). Query answering in information system with functional des. *IICIS* (pp. 113-130).

Buccafurri, F., Furfaro, F., Saccà, D., & Sirangelo, C. (2003). A Quad-Tree Based Multiresolution Approach for Two-Dimensional Summary Data. *Proceedings of the 15th IEEE Int. Conf. on Scientific and Statistical Database Management,* (pp. 127-140).

Buccella, A., Cechich, A., & Brisaboa, N. R. (2003). An ontology approach to data integration. *Journal of Computer Science and Technology, 3*(2), 62-68.

Buccella, A., Cechich, A., & Brisaboa, N. R. (2004). A federated layer to integrate heterogeneous knowledge. *In VODCA'04 First International Workshop on Views on Designing Complex Architectures.* Bertinoro, Italy. Electronic Notes in Theoretical Computer Science, Elsevier Science B.V, 101-118.

Burger, Albert., Kumar, Vijay & Hines, Mary Lou. (1997). Performance of Multiversion and Distributed Two - Phase Locking Concurrency Control Mechanisms in Distributed Databases. International Journal of Information Sciences, 1-2, 129-152.

Burns, E. (2007, July 5). Urban Indian web users. *ClickZ.* Retrieved April 3, 2008, from http://www.clickz.com/showPage.html?page=3626341

Busse, S., Kutsche, R. D., Leser, U., & Weber, H. (1999). Federated information systems: Concepts, terminology and architectures. *Technical Report. Nr. 99-9*, TU Berlin.

Bussler, C. (2003). *B2B Integration.* Springer-Verlag.

Butz, A., Baus, J., & Kruger, A. (2000). *Different views on location awareness.* Accessed November 10, 2006 at http://w5.cs.uni-sb.de/~butz/publications/papers/awareness.pdf

Buyukkokten, O., Garcia-Molina, H., & Paepcke, A. (2001). Seeing the Whole in Parts: text Summarization for Web Browsing on Handled Devices. *Proc. of the 10th Int. World Wide Web Conf.,* (pp. 652-662).

Cabibbo, L., & Torlone, R. (2000). The design and development of a logical system for OLAP. *Proceedings of the 2nd International Conference on Data Warehousing and Knowledge Discovery,* 1-10.

Cai, H. M. (1990). The standpoint of the primitive buddhism (原始佛教的缘起观). *Dharmaghosa (The Voice of Dharma), 1990*(05), 25-27.

Cai, J., Tan, K.-L., & Ooi, B. C. (1997). On incremental cache coherency schemes in mobile computing environments. In A. Gray, & P.-Å Larson (Eds.), *Proceedings of the Thirteenth International Conference on Data Engineering*, April 7-11, 1997 Birmingham U.K, S. 114–123. IEEE Computer Society, Los Altos, CA, USA

Cai, M., & Frank, M. (2004). RDFPeers: A scalable distributed RDF repository based on a structured peer-to-peer network. *Proceedings of the 13th International World Wide Web Conference* (pp. 650-657).

Cai, Y. D., Clutter, D., Pape, G., Han, J., Welge, M., & Auvil, L. (2004). MAIDS: Mining alarming incidents from data streams. *Proceedings of the 2004 ACM International Conference on Management of Data* (pp. 919-920).

Calero, C., Ruiz, F., Baroni, A., Brito e Abreu, F., & Piattini, M. (2006). An ontological approach to describe the SQL:2003 object-relational features. *Computer Standards & Interfaces, 28*(6), 695-713

Calì, A., Calvanese, D., De Giacomo, G., & Lenzerini, M. (2002). Data integration under integrity constraints. *CAiSE* (pp. 262-279).

Calì, A., Calvanese, D., De Giacomo, G., Lenzerini, M., Naggar, P., & Vernacotola, F. (2003). *IBIS: Semantic data integration at work*. Proceedings of the CAiSE 2003, 79–94.

Calí, A., Calvanese, D., Giacomo, D., & Lenzerini, M. (2002). On the expressive power of data integration systems. *In Proceedings of the 21st Int. Conf. on Conceptual Modeling, ER, 2503 of Lecture Notes in Computer Science*. Springer.

Callaghan, K., & Schnell, F. (2001). Assessing the Democratic Debate: How the News Media Frame Elite Policy Discourse. *Political Communication, 18*(2), 183-213.

Callan, J. (2000). Distributed information retrieval. In *Advances in information retrieval* (pp. 127-150). Kluwer Academic Publishers.

Callan, J., & Connell, M. (2001). Query-based sampling of text databases. *ACM Transactions on Information Systems (TOIS), 19*(2), 97-130.

Calvanese, C., Lenzerini, M., & Nardi, D. (1998). Description Logics for Conceptual Data Modelling. In J. Chomicki & G. Saake (Eds.), *Logics for Databases and Information Systems* (pp. 229-263). Kluwer.

Calvanese, D., & Giacomo, G. D. (2005). Data integration: A logic-based perspective. *AI Magazine, 26*(1), 59-70.

Calvanese, D., De Giacomo, G., Lembo, D., Lenzerini, M., & Rosati, R. (2004). Inconsistency tolerance in P2P data integration: An epistemic logic approach. *DBPL* (pp. 692-697).

Calvanese, D., De Giacomo, G., Lenzerini, M., & Rosati, R. (2004). Logical foundations of peer-to-peer data integration. *PODS* (pp. 241-251).

Calvanese, D., Giacomo, G. D., & Lenzerini, M. (2001). A framework for ontology integration. *In SWWS*, 303-316.

Camolesi, L. (2004). Survivability and Applicability in Database Constraints: Temporal Boundary to Data Integrity Scenarios. *Proceedings of V IEEE International Conference on Information Technology: Coding and Computing – ITCC, 1*, 518-522. IEEE Computer Society Press.

Campbell, Charles P. (1998). Rhetorical ethos: A bridge between high-context and low-context cultures? In S. Niemeier, C. P. Campbell, & R. Dirven (Eds.), *The Cultural Context in Business Communication* (pp. 31-47). Philadelphia, PA: John Benjamin.

Canada gets with IT. (2004, September 29). *Insurance-Canada.ca*. Retrieved February 5, 2005, from http://www.insurance-canada.ca/ebusiness/canada/eMarketer-Canada-IT-409.php

Cannataro, M. et al. (2007). Using ontologies for preprocessing and mining spectra data on the Grid. *Future Generation Computer Systems, 23*(1), 55-60.

Cantwell, J. (1998). Resolving conflicting information. *Journal of Logic, Language and Information, 7*(2), 191-220.

Caragea, D., Silvescu, A., & Honavar, V. (2004). A framework for learning from distributed data using sufficient statistics and its application to learning decision trees. *International Journal of Hybrid Intelligent Systems, 1*, 80–89.

Caragea, D., Zhang, J., Bao, J., Pathak, J., & Honavar, V. (2005). *Algorithms and software for collaborative discovery from autonomous, semantically heterogeneous, distributed information sources*. Proceedings of the Conference on Algorithmic Learning Theory, LNCS, 3734, 13–44.

Cardoso, J. (2007). The Semantic Web vision : Where are we ? *IEEE Intelligent Systems, 22*(5), 84-88.

Carnap, R. (1950). *Logical foundations of probability*, University of Chicago Press.

Caroprese & Zumpano. (2006). A framework for merging, repairing and querying inconsistent databases. *ADBIS* (pp. 383-398).

Carpineto C. & Romano G. (2004). Concept Data Analysis: Theory and Applications. John Wiley & Sons, England.

Carter, D. E., & Baker, B. S. (1992). *CE: Concurrent Engineering.* Boston, MA: Addison-Wesley Publishing.

Cary, M. (2001). Towards optimal ε-approximate nearest neighbor algorithms. *Journal of Algorithms, 41*(2), 417-428.

Casanova, M. A., & Amaral De Sa, J. E. (1984). Mapping uninterpreted schemes into entity-relationship diagrams: Two applications to conceptual schema design. *IBM J. Res. & Develop., 28*(1).

Casella, G., & Berger, R.L. (2001). *Statistical inference.* Belmont, CA: Duxbury Press.

Castano, S., Ferrara, A., & Montanelli, S. (2005, February). *Ontology-based interoperability services for semantic collaboration in open networked systems.* Proceedings of the 1st International Conference on Interoperability of Enterprise Software and Applications (INTEROP-ESA 2005), Geneva, Switzerland.

Castano, S., Fugini, M., Martella, G., & Samarati, P. (1995). *Database security.* Addison Wesley.

Catania, B., Maddalena, A., & Vakali, A. (2006). XML document indexes: A classification. *IEEE Internet Computing Journal, 9*(5), 64–71.

Catell, R. G., Barry, D. K., Berler, M., & Eastman, J. (2000). *The object data standard: ODMG 3.0.* Morgan Kaufmann Publishers.

Cattell, R. G. G. et al. (1997). *Object Database Standard: Odmg 2.0*, R. G. G. Cattell, D. K. Barry, D. Bartels, M. Berler, J. Eastman, S. Gamerman, D. Jordan, A. Springer, H. Strickland & D. Wade (Eds.). Morgan Kaufmann Series in Data Management Systems.

Caverlee, J., & Liu, L. (2005). QA-Pagelet: data preparation techniques for large-scale data analysis of the deep Web. *IEEE Transactions on Knowledge and Data Engineering, 17*(9), 1247-1262.

Cellary, W., & Jomier, G. (1992). *Building an object-oriented database system. The story of O_2.*, chapter Consistency of Versions in Object-Oriented Databases. Number 19 in The Morjgan Kaufmann Series in Data Management Systems. Morgan Kaufmann (pp. 447-462).

Celle, A., & Bertossi, L. (1994). Consistent data retrieval. *Information Systems, 19*(4), 33-54.

Cellucci, C. (1998). The scope of logic: Deduction, abduction, analogy. *Theoria,* 217-242.

Cellucci, C. (2000). The growth of mathematical knowledge: An open world view. *The growth of mathematical knowledge,* 153-176.Kluwer.

Ceri, S., & Fraternali, P. (1997). Designing database applications with objects and rules: The IDEA Methodology. Addsion-Wesley. ISBN: 0201403692

Ceri, S., & Pelagatti, G. (1984). *Distributed databases: principles & systems.* McGraw-Hill International Editions – Computer Science Series.

Ceri, S., Fraternali, P., & Paraboschi, S. (2000). XML: Current Developments and Future Challenges for the Database Community. *Proceedings of the 7th International Conference on Extending Database Technology (EDBT),* Springer, LNCS 1777, Konstanz.

Ceri, S., Negri, M., & Pelagatti, G. (1982). Horizontal data partitioning in database design. *Proceedings of the ACM SIGMOD International Conference on Management of Data,* (pp. 128-136).

Cerrudo, C. (2002). Manipulating Microsoft SQL Server using SQL injection (Tech. Rep.). *Application Security, Inc.* Retrieved July 29, 2006, from http://www.appsecinc.com/presentations/Manipulating_SQL_Server_Using_SQL_Injection.pdf

Chakrabarti, K., Ganti, V., Han, J., & Xin, D. (2006). Ranking Objects by Exploiting Relationships: Computing Top-K-over Aggregation. In *Proc. of ACM SIGMOD* (pp. 371–382), Chicago, IL, USA.

Chakrabarti, K., Garofalakis, M., Rastogi, R., & Shim, K. (2000). Approximate query processing using wavelets. *Proceedings of the 26th International Conference on Very Large Data Bases* (pp. 111-122).

Chakrabarti, K., Keogh, E., Mehrotra, S., & Pazzani, M. (2002). Locally adaptive dimensionality reduction for indexing large time series databases. *ACM TODS, 27*(2), 188-228.

Chamberlin, D. D., Clark, J., Florescu, D., & Stefanescu, M. (2000). XQuery 1.0: An XML query language. http://www.w3.org/TR/query-datamodel/.

Chamberlin, D. D., Robie, J., & Florescu, D. (2000). Quilt: An XML query language for heterogeneous data sources. *WebDB 2000.*

Chan, E. P. F. (1993). A Possible World Semantics for Disjunctive Databases, IEEE Trans. Knowl. *Data Eng., 5*(2), 282-292.

Chan, E. P. F. (2003). Buffer Queries. *IEEE Transactions Knowledge Data Engineering, 15*(4), 895-910.

Chan, G. K. Y., Li Q., & Feng, L. (1999). Design and selection of materialized views in a data warehousing environment: A case study. *2nd ACM international workshop on Data warehousing and OLAP (DOLAP 1999)* (pp. 42-47).

Chan, K., & Fu, A. W. (1999). Efficient time series matching by wavelets. *Proceedings of IEEE ICDE,* 126-133. Sydney, Australia.

Chandra, A. K., & Merlin, P. M. (1977). Optimal implementation of conjunctive queries in relational databases. *In Proceedings of the ninth annual ACM symposium on Theory of computing*, Boulder, Colorado, United States, (pp. 77–90), New York, NY, USA.

Chandrasekaran, S., & Franklin, M. J. (2002). Streaming queries over streaming data. *Proceedings of VLDB,* 203-214. Hong Kong, China.

Chang, J.-W., Kim, Y.-K., Kim, S.-M., & Kim, Y.-C. (2006). *New query processing algorithms for range and k-NN search in spatial network databases.* Proceedings of the ER Workshops, 130–139.

Chang, J-W., Um, J-H., & Lee, W-C. (2006). A new trajectory indexing scheme for moving objects on road networks. *Prof of. the 23rd British National Conference on Databases (BNCOD'* 2006), 291-294.

Chang, K., & Cho, J. (2006). Accessing the Web: From Search to Integration. In *Proc. of ACM SIGMOD* (pp. 804–805) , Chicago, IL, USA.

Chang, K., He, B., Li, C., Patel, M., & Zhang, Z. (2004). Structured databases on the Web: observations and implications. *SIGMOD Rec., 33*(3), 61-70.

Chang, S. K., Deufemia, V., & Polese, G. (2005). Normalizing multimedia databases. *Encyclopedia of Database Technologies and Applications,* 408-412.

Chang, S. K., Deufemia, V., Polese, G., & Vacca, M. (2007). A Logic Framework to Support Database Refactoring. In *Proc. of 18th International Conference on Database and Expert Systems Applications, LNCS, 4653,* 509-51.

Chang, W.W. and Schek, H.J. (1989). A signature access method for the STARBURST database system. In *Proc. 19th VLDB Conf.,* 145-153.

Chaowei, P. Y., Wong, D., Ruixin, Y., Menas, K., & Qi, L. (2005). Performance-improving techniques in Web-based GIS. *International Journal of Geographical Information Science, 19*(3), 319-342.

Chaudhri, A. B., Rashid, A., & Zicari, R. (Eds.). (2003). *XML data management: Native XML and XML-enabled database systems.* Addison-Wesley.

Chaudhri, V. K., Farquhar, A., Fikes, R., Karp, P. D., & Rice, J. P. (1998). OKBC: A programmatic foundation for knowledge base interoperability. In *Proceedings of the 1998 National Conference on Artificial Intelligence – AAAI/98.* Cambridge, MA: MIT Press/AAAI Press.

Chaudhuri, S., & Narasayya, V. R. (1997). An efficient cost-driven index selection tool for Microsoft SQL server. *23rd International Conference on Very Large Data Bases (VLDB 1994)* (pp. 146-155).

Chaudhuri, S., Das, G., Datar, M., Motwani, R., & Rastogi, R. (2001). Overcoming limitations of sampling for aggregation queries. *Proceedings of the 17th IEEE International Conference on Data Engineering* (pp. 534-542).

Chaudhuri, S., Das, G., Datar, M., Motwani, R., & Rastogi, R. (2001). Overcoming Limitations of Sampling for Aggregation Queries. *Proc. of the 17th IEEE Int. Conf. on Data Engineering,* (pp. 534-542).

Chaudhuri, S., Datar, M., & Narasayya, V. (2004). Index selection for databases: A hardness study and a principled heuristic solution. *IEEE Transactions on Knowledge and Data Engineering, 16*(11), 1313-1323.

Chaudhuri, S., Motwani, R., & Narasayya, V. (1998). Random sampling for histogram construction: How much is enough? *Proceedings of the 1998 ACM International Conference on Management of Data* (pp. 436-447).

Chaudhuri, S., Narasayya, V., & Sarawagi, S. (2004). Extracting predicates from mining models for efficient query evaluation. *ACM Transactions on Database Systems, 29*(3), 508–544.

Chavent, M. (1997). Analyse des Données symboliques. *Une méthode divisive de classification.* Thèse de doctorat in Sciences. I'Université Paris IX-Dauphine.

Chavez, E., & Figueroa, K. (2004). Faster proximity searching in metric data. In *Lecture notes in computer science: Vol. 2972. Proceedings of MICAI 2004.* México: Springer.

Chavez, E., Navarro, G., Baeza-Yates, R., & Marroquin, J. (2001). Searching in metric spaces. *ACM Computing Surveys, 33*(3), 273-321.

Chawathe, S. S. (1999). Comparing hierarchical data in external memory. *Proceedings of the International Conference on Very Large Data Bases* (pp. 90-101).

Chawathe, S., Garcia-Molina, H., Hammer, J., Ireland, K., Papakonstantinou, Y., Ulman, J., et al. (1994). The TSIMMIS project: Integration of heterogeneous information systems. *Proceedings of ISPJ Conference* (pp. 7-18).

Chawathe, S., Rajaraman, A., Garcia-Molina, H., & Widom, J. (1996). Change detection in hierarchically structured information. *SIGMOD Record (ACM Special Interest Group on Management of Data), 25*(2), 493-504.

Chawla, S., Dwork, C., McSherry, F., Smith, A., & Wee, H. (2005). Towards privacy in public databases. *Proceedings of 2nd Theory of Cryptography Conference* (pp. 363-385).

Chekuri, C., & Rajaraman, A. (1997). Conjunctive Query Containment Revisited. In F. N. Afrati & P. G. Kolaitis, (Eds.), *Proceedings of the 6th International Conference on Database Theory - ICDT '97,* Delphi, Greece, January 8-10, 1997, volume 1186 of Lecture Notes in Computer Science, (pp. 56–70), Heidelberg. Springer-Verlag.

Chen, G., Kerre, E. E., & Vandenbulcke, J. (1996). Normalization based on fuzzy functional dependency in a fuzzy relational data model. *Information Systems, 21*(3), 299-310.

Chen, J. L., Mcleod, D. et al. (1995). Domain-Knowledge-Guided Schema Evolution for Accounting Database Systems. *Expert Systems with Applications, 9*(4), 491-501.

Chen, L., & Ng, R. (2004). On the marriage of edit distance and Lp norms. *Proceedings of VLDB,* 792-803. Toronto, Ontario, Canada.

Chen, L., Özsu, M. T., & Oria, V. (2005). Robust and fast similarity search for moving object trajectories. *Proceedings of ACM SIGMOD,* 491-502. Baltimore, Maryland, USA.

Chen, P. (1976). The entity-relationship model – Toward a unified view of data. *ACM Transactions on Database Systems, 1*(1).

Chen, P. (1976). The entity-relationship model- Toward a unified view of data. *ACM Transaction on Database Systems, 1*(1), 9-36.

Chen, P., Gibson, R., & Geiselhart, K. (2006). *Electronic Democracy? The Impact of New Communications Technologies on Australian Democracy.* Democratic Audit of Australia, Australia National University.

Chen, P., Thalheim, B., & Wong, L. Y. (1999). Future Directions of Conceptual Modeling. *LNCS*, 1565, 287.

Chen, Q., Chen, L., Lian, X., Liu, Y., & Yu, J. X. (2007). Indexable PLA for efficient similarity search. *Proceedings of VLDB*, 453-446. Vienna, Austria.

Chen, S. (2006). *Twig²Stack*: Bottom-up processing of generalized-tree-pattern queries over XML documents. In *Proc. VLDB*, 283-323. Seoul, Korea.

Chen, T., Lu, J., & Ling, T. W. (2005). On boosting holism in XML twig pattern matching. In *Proc. SIGMOD,* 2005, 455-466.

Chen, Y. (2002). Signature files and signature trees. *Information Processing Letters,* 82(4):213-221, 2002.

Chen, Y. (2004). Building Signature Trees into OODBs. *Journal of Information Science and Engineering,* 20: 275-304.

Chen, Y. (2005). On the signature trees and balanced signature trees. *Processings of 21th Conf. on Data Engineering,* Tokyo, Japan, April 2005, pp. 742-753.

Chen, Y. Y., Suel, T., & Markowetz, A. (2006). Efficient Query Processing in Geographic Web Search Engines.

In Proceedings of the 2006 ACM SIGMOD International Conference on Management of Data (SIGMOD 2006), Chicago, Illinois, USA, June 27-29, 2006, (pp. 277-288), ACM Press.

Chen, Y., & Patel, J. M. (2007). Efficient Evaluation of All-Nearest-Neighbor Queries. *In Proceedings of the 23rd IEEE International Conference on Data Engineering (ICDE 2007),* Istanbul, Turkey, 15-20 April, 2007, (pp. 1056-1065). IEEE Computer Society Press.

Chen, Y., Davidson, S., Hara, C., & Zheng, Y. (2003). RRXS: Redundancy Reducing XML Storage in Relations. In *VLDB'03: Proceedings of the 29th international Conference on Very Large Data Bases, 29,* 189–200, Berlin, Germany. VLDB Endowment.

Chen, Z. (1996). Generating suggestions through document structure mapping. *Decision Support Systems, 16*(4), 297-314.

Cheng, Y., & Church, G. M. (2000). *Biclustering of Expression Data.* Paper presented at the eighth International Conference on Intelligent Systems for Molecular Biology.

Cherrington, J. O., Denna, E. L., & Andros, D. P. (1996). Developing an event-based system: The case of IBM's national employee disbursement system. Journal of Information Systems, 10(1), 51-69.

Chien, S.Y., Tsotras, V. J., Zaniolo, C., & Zhang, D. (2006). Supporting complex queries on multiversion XML documents. *ACM Transactions on Internet Technology, 6*(1), 53-84.

Chien, S-Y., Vagena, Z., Zhang, D., Tsotras, V.J., & Zaniolo, C. (2002). *Efficient structural joins on indexed XML documents.* Proceedings of the 28th International Conference on Very Large Data Bases (VLDB 02), Hong Kong, China, 263–274.

Cho, H. J., & Chung, C. W. (2005). An efficient and scalable approach to CNN queries in a road network. *Proceedings of 31st International Conference on Very Large Databases* (pp. 865-876).

Cho, H.-J., & Chung, C.-W. (2005). *An efficient and scalable approach to CNN queries in a road network.* Proceedings of the VLDB 2005, 865–876.

Choenni, S., Blanken, H. M., & Chang, T. (1993a). Index selection in relational databases. *5th International Conference on Computing and Information (ICCI 1993)* (pp. 491-496).

Choenni, S., Blanken, H. M., & Chang, T. (1993b). On the selection of secondary indices in relational databases. *Data Knowledge Engineering, 11*(3), 207-238.

Choi, B., Mahoui, M., & Wood, D. (2003). On the optimality of holistic algorithms for twig queries. In *Proc. DEXA,* 235-244.

Choi, F. (2000). Advances in domain independent linear text segmentation. In *NAACL'00.*

Choi, M.S., et al. (2000). *Two-step backup mechanism for real-time main memory database recovery.* Proceedings of the 7th International Conference on Real-time Computing Systems and Applications, 453–457.

Choi, N., Song, I. Y., & Han, H. (2006). A survey on ontology mapping. *ACM SIGMOD Record, 35*(3), 34-41.

Cholvy, L., & Moral, S. (2001). Merging databases: Problems and examples. *International Journal of Intelligent Systems, Special Issue on Data and Knowledge Fusion, 16*(10).

Chomsky, N. (1965). Aspects of the Theory of Syntax. Cambridge, MA: MIT Press.

Chon, H. D., Agrawal, D., & Amr, El Abbadi (2002). Data management for moving objects. *IEEE Data Engineering Bulletin, 25*(2), 41-47.

Christensen, E., Curbera, F., Meredith, G., & Weerawarana, S. (2001). *Web services description language (WSDL) 1.1 – W3C Note 15 March 2001.* W3C (http://www.w3.org/TR/wsdl).

Christiansen, H., & Martinenghi, D. (2005). *Incremental integrity checking: Limitations and possibilities.* Proceedings of the 12th International Conference on Logic for Programming and Artificial Intelligence and Reasoning, Jamaica, 712–727.

Christiansen, H., & Martinenghi, D. (2006). On simplification of database integrity constraints. *Fundamenta Informaticae, 71*(4), 371-417.

Christodoulakis, D., Soupos, P. et al. (1989). Adaptive DB schema evolution via constrained relationships. *EEE International Workshop onTools for Artificial Intelligence*, 1989. Architectures, Languages and Algorithms, Fairfax, VA , USA, IEEE.

Christodoulakis, S. and Faloutsos, C. (1984). "Design consideration for a message file server," *IEEE Trans. Software Engineering*, 10(2) (1984) 201-210.

Christodoulakis, S., Theodoridou, M., Ho, F., Papa, M. and Pathria, A. (1986). Multimedia document presentation, information extraction and document formation in MINOS - A model and a system. *ACM Trans. Office Inform. Systems*, 4(4), 345-386.

Chu D. et al. (2006). *Approximate Data Collection in Sensor Networks using Probabilistic Models*, ICDE.

Chung, C., Gertz, M., & Sundaresan, N. (2002). Reverse engineering for Web data: From visual to semantic structures. *Proceedings of the 18th International Conference on Data Engineering (ICDE'02).*

Chung, C., Min, J., & Shim, K. (2002, June). APEX: An adaptive path index for XML data. *ACM SIGMOD.*

Ciaccia, P. and Zezula, P. (1996). Declustering of key-based partitioned signature files. *ACM Trans. Database Systems*, 21(3), 295-338.

Ciaccia, P., Patella, M., & Zezula, P. (1997). M-tree: An efficient access method for similarity search in metric spaces. *VLDB Journal*, 426-435.

Civilis, A., Jensen, C. S., & Pakalnis, S. (2005). Techniques for efficient road-network-based tracking of moving objects. *IEEE Transactions on Knowledge and Data Engineering, 17*(5), 698-712.

Clark, J., & DeRose, S. (1999). XML path language (XPath), Version 1.0 [W3C working draft]. Retrieved September 20, 2006, from http://www.w3.org/TR/xpath/

Clarke, I., Sandberg, O., Wiley, B., & Hong, T. W. (2000). Freenet: A distributed anonymous information storage and retrieval system. *Proceedings of the ICSI Workshop on Design Issues in Anonymity and Unobservability* (pp. 311-320).

Clève, A. (2006). Co-transformations in database applications evolution. In R. Lämmel, J. Saraiva, & J. Visser (Eds.), *Lecture notes in computer science: Vol. 4143. Generative and transformational techniques in software engineering* (pp. 399-411). Springer-Verlag.

Cleve, A., Henrard, J., & Hainaut, J.-L. (2006). Data reverse engineering using system dependency graphs. *WCRE.*

Clifford J., & Tansel, A. U. (1985). On an algebra for historical relational databases: Two views. *In Proceedings of ACM SIGMOD International Conference on Management of Data*, 247-265.

Clifford, J., Croker, A., & Tuzhilin, A. (1993). On completeness of historical data models. *ACM Transactions on Database Systems, 19*(1), 64-116.

Cobena, G., Abiteboul, S., & M. A. (2002). Detecting Changes in XML Documents. *In Data Engineering 2002 (ICDE2002)*, (pp. 41-52).

Codd, E. (1970). A relational model for large shared data banks. *Communications of the ACM, 13*(6), 377-387.

Codd, E. (1993). *Providing OLAP (on-line analytical processing) to users-analysts: An IT mandate* [white paper]. E.F. Codd and Associates.

Codd, E. F (1985). Is Your DBMS Really Rational, (Codd's 12 rules). *Computerworld Magazine.*

Codd, E. F. (1970). A relational model for large shared data banks. Comm. of the ACM, 13(6):377–387.

Cohen, A. M., & Hersh, W. R. (2005). A survey of current work in biomedical text mining. *Briefings in Bioinformatics, 6*(1), 57–71.

Cohn, A.G., & Hazarika, M. (2001). Qualitative spatial representation and reasoning: An overview. *Fundamenta Informaticae, 46*(1-2), 1–29.

Cohn, D., Caruana, R., & McCallum, A. (2003). *Semi-supervised clustering with user feedback.* Technical Report TR2003-1892, Cornell University, USA.

Colliat, G. (1996). OLAP, relational, and multidimensional database systems. *SIGMOD Record, 25*(3), 64-69.

Collins, H. (2003). *Enterprise knowledge portals.* NY: American Management Association.

Comer, D. (1979). The ubiquitous B-tree. *ACM Computing Surveys, 11*(2), 121-137.

Conradi, R., & Westfechtel, B. (1998). Version models for software configuration management. *ACM Computing Surveys, 30*(2), 232-282.

Conway, W. A. & Morrison, T. (1999). The color of money. *Global business basics*, Retrieved December 10, 1999 from, http://www.getcustom.com/omnibus/iw0897.html

Cooke, A. W., Gray, A., & Nutt, W. (2004). The relational grid monitoring architecture: Mediating information about the grid. *Journal of Grid Computing, 2*(4), 323-339.

Cooper, B. F., Sample, N., Franklin, M., Hialtason, A. B., & Shadmon, M. (2001, September). A fast index for semistructured data. In *Proc. VLDB*, 341-350.

Corral, A., Manolopoulos, Y., Theodoridis, Y., & Vassilakopoulos, M. (2000). Closest Pair Queries in Spatial Databases. *In Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data (SIGMOD 2000)*, Dallas, Texas, USA, 16-18 May, 2000, (pp. 189-200), ACM Press.

Corral, A., Manolopoulos, Y., Theodoridis, Y., & Vassilakopoulos, M. (2004) Multi-way Distance Join Queries in Spatial Databases. *GeoInformatica, 8*(4), 373-400.

Corral, A., Manolopoulos, Y., Theodoridis, Y., & Vassilakopoulos, M. (2006). Cost models for distance joins queries using R-trees. *Data and Knowledge Engineering, 57*(1), 1-36.

Cosmadakis, S., Kanellakis, P. C., & Spyratos, N. (1986). Partition semantics for relations. *Journal of Computer and System Sciences, 33*(2), 203-233.

Costa, G., Manco, G., Ortale, R., & Tagarelli, A. (2004). A tree-based approach to clustering XML documents by structure. *Proceedings of the European Conference on Principles and Practice of Knowledge Discovery in Databases* (pp. 137-148).

Costa, R. L. de C., Lifschitz, S., Noronha, M. F de., & Salles, M. A. V. (2005). Implementation of an agent architecture for automated index tuning. In Proceedings. *Of the 21st International Conference on Data Engineering Workshops (ICDEW)*, 1215-1222.

Cox, S., Daisay, P., Lake, R., Portele, C., & Whiteside, A. (Eds.). (2002). OpenGIS geography markup language (GML), Version 3.0. Open Geospatial Consortium. Retrieved from http://www.opengeospatial.org/standards/gml

Crainiceanu, A., Linga, P., Gehrke, J., & Shanmugasundaram, J. (2004). Querying peer-to-peer networks using p-trees. *Proceedings of the 2004 ACM International Workshop on Web and Databases, in conjunction with the 2004 ACM International Conference on Management of Data* (pp. 25-30).

Crescenzi, V., Mecca, G., & Merialdo, P. (2001). RoadRunner: towards automatic data extraction from large web sites. *27th International Conference on Very Large Data Bases (VLDB'01)*, 109-118.

Crespo, A., & Garcia-Molina, H. (2002). Routing indices for peer-to-peer systems. *Proceedings of the 22nd IEEE International Conference on Distributed Computing Systems* (pp. 23-34).

Crespo, A., & Garcia-Molina, H. (2003). *Semantic overlay networks for P2P systems* (Tech. Rep.). Stanford University, Computer Science Department.

Crick, F. H. C. (1958). On protein synthesis. *Symposium of the Society for Experimental Biology XII*, 139-163.

Crick, F. H. C. (1970). Central Dogma of Molecular Biology. *Nature, 227*, 561-563.

CROSI (2005). *Capturing Representing and Operationalising Semantic Integration*. University of Southampton and Hewlett Packard Laboratories

Cruz, I. F., Sunna, W., & Chaudhry, A. (2004). Semi-automatic ontology alignment for geospatial data integration. In M. J. Egenhofer, C. Freksa, & H. J. Miller (Eds.), *Lecture notes in computer science* (Vol. 3234, pp. 51-66). GIScience.

Cuadra, D., & Martínez, P. (2002). Preserving relationship cardinality constraints in relational schemata. *Database Integrity: Challenges and Solutions*, (Ed.). Idea Group Publishing.

Cubero, J. C., & Vila, M. A. (1994). A new definition of fuzzy functional dependency in fuzzy relational databases. *International Journal of Intelligent Systems, 9*(5), 441-448

Cui, Z., & O'Brien, P. (2000). Domain ontology management environment. In *Proceedings of the 33rd Hawaii International Conference on System Sciences.*

Curtmola, E., Amer-Yaiha, S., Brown, P., & Fernandez, M. (2005). GalaTex: A Conformant Implementation of the Xquery Full-Text Language. *Proc. of WWW 2005.* (pp. 1024–1025), Chiba, Japan.

Cuzzocrea, A. (2005). Towards a semantics-based framework for KD- and IR-style resource querying on XML-based P2P information systems. *Proceedings of the 2005 IEEE/WIC/ACM International Conference on Web Intelligence* (pp. 58-61).

Cuzzocrea, A. (2005a). Overcoming limitations of approximate query answering in OLAP. *Proceedings of the 9th IEEE International Conference on Database Engineering and Applications* (pp. 200-209).

Cuzzocrea, A. (2005a). Overcoming Limitations of Approximate Query Answering in OLAP. *Proc. of the 9th IEEE Int. Conf. on Database Engineering and Applications*, (pp. 200-209).

Cuzzocrea, A. (2005b). Providing probabilistically-bounded approximate answers to non-holistic aggregate range queries in OLAP. *Proceedings of the 8th ACM International Working Group on Data Warehousing and OLAP* (pp. 97-106).

Cuzzocrea, A. (2006). Improving range-sum query evaluation on data cubes via polynomial approximation. *Data & Knowledge Engineering, 56*(2), 85-121.

Cuzzocrea, A., & Wang, W. (2007). Approximate Range-Sum Query Answering on Data Cubes with Probabilistic Guarantees. *Journal of Intelligent Information Systems*, Springer Science, Springer Science, 28(2), 161-197.

Cuzzocrea, A., Furfaro, F., & Saccà, D. (2003). Hand-OLAP: A System for Delivering OLAP Services on Handheld Devices. *Proc. of the 6th IEEE Int. Conf. on Autonomous Decentralized Systems*, (pp. 80-87).

Cuzzocrea, A., Furfaro, F., Masciari, E., Saccà, D., & Sirangelo, C. (2004). Approximate query answering on sensor network data streams. In A. Stefanidis & S. Nittel (Eds.), *Sensor-based distribution geocomputing* (pp. 53-72). CRC Press.

Cuzzocrea, A., Wang, W., & Matrangolo, U. (2004). Answering approximate range aggregate queries on OLAP data cubes with probabilistic guarantees. In *Lecture notes in computer science: Vol. 3181. Proceedings of the 6th International Conference on Data Warehousing and Knowledge Discovery* (pp. 97-107). Springer-Verlag.

Czajkowski, S., Fitzgerald, K., & Foster, I. (2001). Grid information services for distributed resource sharing. *Proceedings of the 10th IEEE International Symposium on High-Performance Distributed Computing* (pp. 181-194).

Dageville, B., & Dias, K. (2006) Oracle's self-tuning architecture and solutions. *Bulletin of the IEEE Computer Society Technical Committee on Data Enginering, 29*(3).

Dageville, B., Das, D., & Dias, K. (2004). Automatic SQL tuning in oracle 10g. *In Proceedings of the 30th Very Large Database Conference (VLDB)*, 1098-1109. Toronto, Canada.

Dageville, B., Das, D., Dias, K., Yagoub, K., Zaït, M., & Ziauddin, M. (2004). Automatic SQL tuning in Oracle 10g. *30th International Conference on Very Large Data Bases (VLDB 2004)* (pp. 1098-1109).

Dalamagas, T., Cheng, T., Winkel, K. J., & Sellis, T. (2006). A methodology for clustering XML documents by structure. *Information Systems, 31*(3), 187-228.

Dar, S., Franklin, M. J., Jonsson, B., Srivastava, D., & Tan, M. (1996). Semantic Data Caching and Replacement. In T. M.Vijayaraman, A. P. Buchmann, C. Mohan, & N. L. Sarda (Eds.), *Proceedings of 22th International Conference on Very Large Data Bases (VLDB'96)*, (pp. 330–341), San Francisco, CA, USA. Morgan Kaufmann.

Darmont, J., Boussaïd, O., Ralaivao, J.C., & Aouiche, K. (2005). *An architecture framework for complex data warehouses.* Proceedings of the 7th International Conference on Enterprise Information Systems (ICEIS 05), Miami, Florida, 370–373.

Dasu, T., Johnson, T., Muthukrishnan, S., & Shkapenyuk, V. (2002). Mining database structure or how to build a data quality browser. *Proceedings of the 2002 ACM International Conference on Management of Data* (pp. 61-72).

Daswani, N., Garcia-Molina, H., & Yang, B. (2003). Open problems in data-sharing peer-to-peer systems. *Proceedings of the International Conference on Database Theory (ICDT), 2003,* 1-15.

Database Language SQL – Part 2: Foundation. (1999). *ISO/IEC, 9075*(2), *1999* standard.

Database Language SQL – Part 2: Foundation. (2003). *ISO/IEC, 9075*(2), *2003* standard.

Date, C. D., Darwen, H., & Lorentzos, N. (2003). *Temporal data and the relational data model.* Morgan Kaufmann, 2002.

Date, C. J. (2000). *An introduction to database systems* (7th ed.). Reading, MA: Addison Wesley.

Date, C. J. (2000). *What not How: The Business Rules Approach to Applications Development.* Boston, MA: Addison-Wesley Publishing.

Date, C.J. (2003). *An introduction to database systems.* Pearson Addison Wesley.

Date, J. (1995). *An Introduction to Database Systems.* Addison Wesley.

Daudjee, K., & Salem, K. (2006). Lazy Database Replication with Snapshot Isolation. *International Conference on Very Large Data Bases,* (pp. 715-726).

David, A. S. (2002). Detecting events with date and place information in unstructured text. *In Proceedings of the 2nd ACM+IEEE Joint Conference on Digital Libraries,* Portland, OR, (pp. 191-196).

David, J. S. (1995). An empirical analysis of REA accounting systems, productivity, and perceptions of competitive advantage. Unpublished doctoral dissertation, Michigan State University.

David, J. S., Dunn, C. L., & McCarthy, W. E. (1999). Enterprise resource planning systems research: The necessity of explicating and examining patters in symbolic form. Working paper, Arizona State University.

Davidson, S., Fan, W., & Hara, C. (2007). Propagating XML Constraints to Relations. *J. Comput. Syst. Sci., 73*(3), 316–361.

Davies, N., & Gellersen, H. (2002) Beyond prototypes: Challenges in Deploying Ubiquitous Systems. *IEEE Pervasive Computing, 1*(1), 29-35.

Davis, K. H., & Aiken, P. H. (2000). Data reverse engineering: A historical view. *Proceedings of the 7th Working Conference on Reverse Engineering (WCRE'00).*

Davis, K. H., & Arora, A. K. (1985). A methodology for translating a conventional file system into an entity-relationship model. *Proceedings of ERA.*

De Almeida, V. T., & Güting, R. H. (2005). Indexing the trajectories of moving objects in networks. GeoInformatica, *9*(1), 33-60.

De Almeida, V.T. (2006). *Moving objects in networks databases.* Proceedings of the EDBT Ph.D. WorkshopEDBT Workshops 2006, 75–85.

De Almeida, V.T., & Güting, R.H. (2005). Indexing the trajectories of moving objects in networks. *GeoInformatica, 9*(1), 33–60.

De Almeida, V.T., & Güting, R.H. (2006). *Using Dijkstra's algorithm to incrementally find the k-nearest neighbors in spatial network databases.* Proceedings of the SAC 2006, 58–62.

de Bruijn, J., Fensel, D., Keller, U., & Lara, R. (2005). Using the Web service modeling ontology to enable Semantic E-Business. *Communications of the ACM, 48*(12), 43-47.

de Carvalho Costa, R. L., Lifschitz, S., & Vaz Salles, M. A. (2003). Index Self-tuning with Agent-based Databases. *CLEI Electronic Journal, 6*(1).

De Raedt, L. (2002). A perspective on inductive databases. *SIGKDD Explorations Newsletter, 4*(2) 69–77.

Dean, R. (1966). *Elements of abstract algebra.* New York: Wiley.

Decker, H. (1987). Integrity enforcement on deductive databases. In L. Kerschberg (Ed.), *Expert database system* (pp. 381-395). Benjamin Cummings.

Decker, H. (1998). Some notes on knowledge assimilation in deductive databases. In Freitag et al. (Eds.), *Lecture notes in computer science: Vol. 1472. Transactions and change in logic databases* (pp. 249-286). Springer.

Decker, H. (2003). Historical and computational aspects of paraconsistency in view of the logic foundation of databases. In Thalheim et al. (Eds.), *Lecture notes in computer science: Vol. 2582. Semantics in databases* (pp. 63-81). Springer.

Decker, H., & Martinenghi, D. (2006). A relaxed approach to integrity and inconsistency in databases. *Proceedings of 13th LPAR* (LCNS 4246, pp. 287-301).

Decker, H., & Martinenghi, D. (2006). Integrity checking for uncertain data. *Proceedings of 2nd Twente Data Management Workshop on Uncertainty in Databases* (pp. 41-48).

Decker, H., Martinenghi, D., & Christiansen, H. (2006). Integrity checking and maintenance in relational and deductive databases and beyond. *In Intelligent Databases: Technologies and Applications,* 238-285. Idea Group.

Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K., & Harshman, R. (1999). Indexing by latent semantic analysis. *Journal of the American Society for Information Science, 41*(6), 391-407.

Dekkers, M., & Weibel, S. (2003, April). State of the Dublin core initiative. *D-Lib Magazine, 9*(4). (http://www.dlib.org/dlib/april03/weibel/04weibel.html).

Delgrande, J. P., Schaub, T., & Tompits, H. (2003). A framework for compiling preferences in logic programs. *TPLP, 3*(2), 129-187.

Dempster, A., Laird, N., & Rubin, D. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B, 39*(1), 1-38.

Denecker, M., & Kakas, A. C. (2002). Abduction in logic programming. *Computational Logic: Logic Programming and Beyond* (pp. 402-436).

Deng, K., Zhou, X., & Shen, H.T. (2007). *Multi-source skyline query processing in road networks*. Proceedings of the ICDE 2007.

Deng, K., Zhou, X., Shen, H.T., Xu, K., & Lin, X. (2006). *Surface k-NN query processing.* Proceedings of the ICDE 2006, 78.

Dennis, A., Wixom, B. H., & Tegarden, D. (2005). *Systems analysis and design with UML version 2.0* (2nd ed.). Wiley.

Deppisch, U. (1986). S-tree: A Dynamic Balanced Signature Index for Office Retrieval. *ACM SIGIR Conference*, 77-87.

Derniame, J.-C., & Gruhn, V. (1994). Development of process-centered IPSEs in the ALF project. *Journal of Systems Integration, 4*(2), 127–150.

DeRosa, M. (2004). *Data mining and data analysis for counterterrorism*. Center for Strategic and International Studies.

Deshpande A. et al. (2004). *Model-Driven Data Acquisition in Sensor Networks, VLDB.*

Deufemia, V., Polese, G., Tortora, G., & Vacca, M. (2007). Conceptual Foundations of Interrogative Agents. In *Proc. of Workshop from Objects to Agents* (*WOA'07*), (pp. 26-33).

Deutsch, A., Popa, L., & Tannen, V. (2006). Query reformulation with constraints. *SIGMOD Record, 35*(1), 65-73.

Deveaux, M. (2003). A Deliberative Approach to Conflicts of Culture. *Political Theory, 31*(6), 780-807.

Devogele, T., Parent, C., & Spaccapietra, S. (1998). On spatial database integration. *Intl. Journal of Geographical Information Science, 12*(4), 335–352.

Devogele, T., Parent, C., & Spaccapietra, S. (1998). On spatial database integration. *International Journal of Geographic Information Science, 12*(4), 335-352.

DeWitt, D. J., & Gray, J. (1992). Parallel Database Systems: The Future of High Performance Database Systems. *Communications of the ACM, 35*(6), 85-98.

Dey, D., Storey, V., & Barron, T., (1999). Improving Database Design through the Analysis of Relationships. *ACM Transactions on Database Systems, 24*(4), 453-486.

Dias, K., Ramacher, M., Shaft, U., Vftenkataramani, V., & Wood, G. (2005). Automatic performance diagnosis and tuning in oracle. *In Proceedings of CIDR.*

Diday, E. (2002). An introduction to Symbolic Data Analysis and the Sodas software. *The Electronic Journal of Symbolic Data Analysis.* Retrieved from http://www.jsda.unina2.it/volumes/Vol0/Edwin.PDF on December 2007.

Diday, E. (2003). Concepts and Galois Lattices in Symbolic Data Analysis. *Journées de l'Informatique Messine*. Knowledge Discovery and Discrete Mathematics Metz, France.

Diday, E. (2004). From Data Mining to Knowledge Mining: Symbolic Data Analysis and the Sodas Software. *Workshop on Applications of Symbolic Data Analysis.* January 2004. Lisboa Portugal.

Diday, E., & Billard, L. (2002). *Symbolic Data Analysis: Definitions and examples.* Retrieved from http://www.stat.uga.edu/faculty/LYNNE/tr_symbolic.pdf. on November 2007.

Diday, E., & Noirhomme-Fraiture, M. (Eds.) (2008). *Symbolic Data Analysis and the SODAS Software*. Chichester, West Sussex, England: Wiley-Interscience.

Ding, Z., & Güting, R. H. (2004). Modelling temporally variable transportation networks. *Proceedings of 9th International Conference on Database Systems for Advances Applications* (pp. 154-168).

Dinur, I., & Nissim, K. (2003). *Revealing information while preserving privacy.* Proceedings of ACM Symposium on Principles of Database Systems (PODS'03), San Diego, CA.

Do, H., & Rahm, E. (2002). *COMA: A system for flexible combination of schema matching approach*. Proceedings of the 28th VLDB Conference, Hong Kong, China.

Dobkin, D., Jones, A. K., & Lipton, R. J. (1979). Secure Databases: Protection Against User Influence. *ACM Transactions on Database Systems, 4*(1), 97-106.

Dobra, A., Garofalakis, M., Gehrke, J., & Rastogi, R. (2002). Processing complex aggregate queries over data streams. *Proceedings of the 2002 ACM International Conference on Management of Data* (pp. 61-72).

Domingos, P. (1997). Knowledge acquisition from examples via multiple models. Proceedings of the Fourteenth International Conference on Machine Learning, Nashville, Tennessee, 98–106.

Domingos, P., & Hulten, G. (2000). Mining high-speed data streams. *Proceedings of the 2000 ACM International Conference on Knowledge Discovery from Data* (pp. 31-42).

Domingues, E., Lloret, J., & Zapata, M. A. (2003). Architecture for Managing Database Evolution. *Lecture Notes in Computer Science, 2784*, 63-74. Springer-Verlag Publishing.

Doorn, J. H., & Rivero, L. C. (2002). *Database Integrity: Challenges and Solutions.* Hershey, PA: Idea Group Publishing.

Dorigo, M., Birattari, M., & Stutzle, T. (2006). *An Introduction to Ant Colony Optimization.* Technical Report 2006-010, IRIDIA, Bruxelles, Belgium.

Dou, D, & LePendu, P. (2006). Ontology-based integration for relational databases. *Proc. Of 2006 ACM Symposium on Applied computing,* (pp. 461-466).

Doucet, A., & Myka, H. (2002). Naive clustering of a large XML document collection. *Proceedings of the Annual ERCIM Workshop of the Initiative for the Evaluation of XML Retrieval (INEX)* (pp. 81-88).

Driskill, L. (1996). Collaborating across national and cultural borders. In D. C. Andrews (Ed.), *International dimensions of technical communication* (pp. 21-44). Arlington, VA: Society for Technical Communication.

Drucker, P. (2001, Nov. 1). The next society: A survey of the near future. *The Economist*, pp. 3-5.

Du, Y., Zhang, D., & Xia, T. (2005). The Optimal-Location Query. *In Proceedings of the Symposium on Spatial and Temporal Databases, (SSTD 2005),* Angra dos Reis, Brazil, 22-24 August, 2005, LNCS 3633, (pp. 163-180). Springer.

Dubois, D., & Prade, H. M. (2000). *Fundamentals of fuzzy sets,* Kluwer Academic.

DuChjarme (2006). Relational database integration with RDF/OWL, http://2006.xmlconference.org/programme/presentations/188.html

Duda, R., Hart, E., & Stork, D. (2000). *Pattern recognition*. New York: Wiley.

Dullea, J., Song, Li-Y., & Lamprou, I. (2003). An analysis of structural validity in entity-relationship modeling. *Data and Knowledge Engineering, 47*(2), 167-205.

Dung, P. M. (1995). On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and *n*-person games. *Artificial Intelligence, 77*(2), 321-358.

Dung, P. M. (1996). Integrating data from possibly inconsistent databases. *COOPIS* (pp. 58-65).

Dunleavy, P., Margetts, H., Bastow, S., Callaghan, R., & Yared, H. (2002). *Government on the Web II.* London School of Economics, LSE Research Online, Technical Report, London, UK.

Dunn, C. L., & Grabski, S. V. (1998). The effect of field independence on conceptual modeling performance. Advances in Accounting Information Systems, 6, 65-77.

Dunn, C. L., & Grabski, S. V. (2000). Perceived semantic expressiveness of accounting systems and task accuracy effects. International Journal of Accounting Information Systems, 1(2), 79-87.

Dunn, C. L., & Grabski, S. V. (2002). Empirical research in semantically modeled accounting systems. In V. Arnold & S. G. Sutton (Eds.), Researching Accounting as an Information Systems Discipline. Sarasota, FL: American Accounting Association (pp. 157-180).

Dunn, C. L., & McCarthy, W. E. (1997). The REA accounting model: Intellectual heritage and prospects for progress. Journal of Information Systems, 11(1), 31- 51.

Dunn, C. L., & McCarthy, W. E. (2000). Symbols used for economic storytelling: A progression from artifactual to more natural accounting systems. Working paper, Michigan State University.

Dunn, C. L., Cherrington, J. O, & Hollander, A. S. (2005). Enterprise Information Systems: A Pattern-based Approach, 3rd edition. New York: McGraw-Hill Irwin.

Dunn, R., Harrison, A. R., & White, J. C. (1990). Positional accuracy and measurement error in digital databases of land use: an empirical study. *International Journal of Geographic Information Systems, 4*(4), 385- 398.

Duo, S. (1996). Some characters of Buddhism different from the other religion. *Tibetan Folklore, 1996*(03), 52-64.

Duo, S. (2006). Several Key Features of the Difference between Buddhism and Other Religions. *N.W.Ethno-National Studies, 2006*(02).

Dupont, S. (1997). *Génération de modèles numériques de terrain par interférométrie ROS*. Thèse, Université de Nice- Sophia Antipolis.

Dutch, A., Fernandez, M., & Florescu, D. (1999, May). A query language for XML, A. Levy, D.Suciu (Eds.). In *Proc. 8th World Wide Web Conf.,* 77-91.

Dwork, C., McSherry, F., Nissim, K., & Smith, A. (2006). *Calibrating noise to sensitivity in private data analysis.* Proceedings of Theory of Cryptography (TCC'06), New York.

Dyer, J. G., Lindemann, M., Perez, R., Sailer, R., Smith, S. W., van Doorn, L., et al. (2001). The IBM secure coprocessor: Overview and retrospective. *IEEE Computer,* pp. 57-66.

Eadon, G., Chong, E. I., Shankar, S., Raghavan, A., Srinivasan, J., & Das, S. (2008). Supporting Table Partitioning by Reference in Oracle. *To appear in ACM SIGMOD International Conference on Management of Data (SIGMOD'08).*

Eastman, C., & Jansen, B (2003). Coverage, Relevance, and Ranking: The Impact of Query Operators on Web Search Engine Results. *ACM Transactions on Information Systems, 21*(4), 383–411.

ebXML (2007). Electronic Business using eXtensible Markup Language (ebXML). www.ebxml.org

ebXML BP (2007). OASIS ebXML Business Process. www.oasis-open.org/committees/tc_home.php?wg_abbrev=ebxml-bp

Eder, J., & Koncilia, C. (2000). *Evolution of dimension data in temporal data warehouses* [technical report]. Austria: University of Klagenfurt, Austria.

Eder, J., & Koncilia, C. (2001). *Changes of dimension data in temporal data warehouses.* Proceedings of the DaWak'01, Munich, Germany, 284–293.

Eder, J., Koncilia, C., & Morzy, T. (2002). *The COMET metamodel for temporal data warehouses.* Proceedings of the 14th International Conference on Advanced Information Systems Engineering (CAiSE 02), Toronto, Canada, 2348, 83–99.

Edwards, H. M., & Munro, M. (1995). Deriving a logical model for a system using recast method. *Proceedings of the 2nd IEEE WC on Reverse Engineering.*

Egenhofer, M. (1993). A model for detailed binary topological relationships. *Geomatica, 47*(3-4), 261–273.

Egenhofer, M. (2002). *Toward the semantic geospatial Web.* Proceedings of the 10th ACM International Symposium on Advances in Geographic Information Systems, 1–4.

Eidenberger, H. (2006). Evaluation and analysis of similarity measures for content-based visual information retrieval. *Multimedia Systems, 12*(2), 71-87.

Eisen, M. B., Spellman, P. T., Brown, P. O., & Botstein, D. (1998). Cluster analysis and display of genome-wide expression patterns. *Proceedings of the National Academy of Science, 95,* 14863-14868.

Eisenberg, A., Melton, J., Kulkarni, K., & Zemke, F. (2004). SQL:2003 has been published. *ACM SIGMOD Record, 33*(1), 119-126.

Eisenberg, A., Melton, J., Kulkarni, K., Michels, J. E., & Zemke, F. (2004). SQL: 2003 has been published. *ACM SIGMOD Record* 33(1), 119-126.

Elmasri, R., & Navathe, S. B. (2006). *Fundamentals of database systems,* 5th edition. Addison-Wesley.

Elmasri, R., Weeldreyer J., & Hevner, A. (1985). The category concept: An extension to the Entity-Relationship model. *Data and Knowledge Engineering, 1,* 75-116.

Elnikety, S., Dropsho, S., & Zwaenepoel, W. (2007). Tashkent+: Memory-aware Load Balancing and Update Filtering in Replicated Databases. *EuroSys Conference,* (pp. 399-412).

Elnikety, S., Zwaenepoel, W., & Pedone, F. (2005). Database Replication Using Generalized Snapshot Isolation. *Symposium on Reliable Distributed Systems,* (pp. 73-84).

Elvang-Goransson, M., & Hunter, A. (1995). Argumentative logics: Reasoning from classically inconsistent information. *Data and Knowledge Engineering, 16*(1), 125- 145.

Engardio, P. (2006). Let's offshore the lawyers. *BusinessWeek*, Retrieved April 3, 2008 from http://www.businessweek.com/magazine/content/06_38/b4001061.htm?chan=search

Engels, G., Gogolla, M., Hohenstein, U., Hulsmann, K., Lohr-Richter, P., Saake, G., & Ehrich, H-D. (1992/93). Conceptual Modeling of database applications using an extended ER model. *Data and Knowledge Engineering 9*, 157-204.

Erdik, M., Fahjan, Y., Ozel, O., Alcik, H., Mert, A., & Gul, M. (2003). Istanbul earthquake rapid response and early warning system. *Bulletin of Earthquake Engineering, 1*(1), 157-163.

Erdmann, M. (2001). *Ontologien zur konzeptuellen Modllierung der Semantik von XML*. PhD dissertation,iversity of Karlsruhe.

ESRI. (2003). *Spatial data standards and GIS interoperability.* Retrieved from www.isotc211.org/Outreach/Newsletter/Newsletter_04_2004/Appendix_4.pdf

Esselink, B. (2000). *A practical guide to localization.* Philadelphia, PA: John Benjamins.

Ester, M., Kriegel, H. P., & Xu, X. (1995). Knowledge discovery in large spatial databases: Focusing techniques for efficient class identification. In *Lecture notes in computer science: Proceedings of 4th International Symposium on Large Spatial Databases (SSD'95)* (pp. 67-82). Springer.

Ester, M., Kriegel, H.-P., Sander, J. o. r., & Xu, X. (1996). *A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise.* Paper presented at the second International Conference on Knowledge Discovery and Data Mining (KDD'96).

Ester, M., Kriegel, H.-P., Sander, J., & Xu, X. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. *Proceedings of the 2nd ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD'96)* (pp. 226-231).

Estrin, D., Culler, D., Pister, K., & Sukhatme, G. (2002). Connecting the Physical World with Pervasive Networks. *IEEE Pervasive Computing, 1*(1), 59-69.

Estrin, D., Govindan, R., Heidemann, J., & Kumar, S. (1999). Next Century Challenges: Scalable Coordination in Sensor Networks. *Proceedings of the ACM/IEEE International Conference on Mobile Computing and Networking. Seattle, Washington*, USA, (pp. 263-270).

Estrin, D., Govindan, R., Heidemann, J., & Kumar, S. (1999). Next century challenges: Scalable coordination in sensor networks. *Proceedings of the International Conference on Mobile Computing and Networks (MobiCom), 1999*, 256-262.

Eswaran, K. P., & Chamberlin, D. D. (1975). Functional specifications of a subsystem for database integrity. *Proceedings of the 1st International Conference on Very Large Data Bases (VLDB 1), 1*(1), 48-68.

Eswaran, K.P., & Chamberlin, D.D. (1975). *Functional specifications of a subsystem for data base integrity.* Proceedings of the 1st International Conference on Very Large Data Bases (VLDB 1), 48–68.

Etzion, O., Jajodia, S., & Sripada, S. (1998). *Temporal databases: Research and practice*. Springer Verlag.

Eugster, P., Felber, P., Guerraoui, R., & Kermarrec, A.-M. (2003). The many faces of publish/subscribe. *ACM Computing Surveys (CSUR), 35*(2).

Euler, T., & Scholz, M. (2004). Using ontologies in a kdd workbench. In P. Buitelaar, et al. (Eds.), *Workshop on Knowledge Discovery and Ontologies at ECML/PKDD* '04, (pp. 103-108).

Euzenat, J., & Shvaiko, P. (2007). *Ontology matching.* Heidelberg: Spinger-Verlag, (DE), isbn 3-540-49611-4. (pp. 341).

Euzenat, J., & Shvaiko, P. (2007). *Ontology matching.* Springer Verlag

Evfimievski, A., Gehrke, J., & Srikant, R. (2003). Limiting privacy breaches in privacy preserving data mining. *Proceedings of ACM Symposium on Principles of Database Systems (PODS'03)* (pp. 211-222).

Evfimievski, A., Srikant, R., Agrawal, R., & Gehrke, J. (2002). Privacy preserving mining of association rules. *Proceedings of 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'02)* (pp. 217-228).

Fagin, R. (1996). Combining fuzzy information from multiple systems. *Proc. Fifteenth ACM Symposium on Principles of Database Systems*, 216-226.

Fagin, R., Kolaitis, P., Miller, R.J., & Popa, L. (2003). *Data exchange: Semantics and query answering.* Proceedings of the ICDT 2003, 207–224.

Fagin, R., Mendelzon, A. O., & Ullman, J. D. (1982). A simplified universal relation assumption and its properties. *ACM Transactions on Database Systems, 7*(3), 343-360.

Faïz, S. (1999). Systèmes d'informations géographiques : Information qualité et datamining. *Editions C.L.E,* 362.

Faïz, S., & Mahmoudi, K. (2005, September). Semantic enrichement of geohraphical databases. In L. Rivero, J. Doorn, V. et Ferraggine (Eds.) *Encyclopedia of database technologies and applications*, Idea Group, Etats-Unis, (pp. 587-592).

Falkovych, K., Sabou, & Stuckenschmidt, H. (2003). UML for the Semantic Web: Transformation-based approaches. In *Knowledge Transformation for the Semantic Web*, B. Omelayenko & M. Klein (eds.). Amsterdam: IOS Press.

Faloutsos, C. (1985). Access Methods for Text. *ACM Computing Surveys,* 17(1), 49-74.

Faloutsos, C. (1992). Signature Files. In: *Information Retrieval: Data Structures & Algorithms*, edited by W.B. Frakes and R. Baeza-Yates, Prentice Hall, New Jersey, 44-65.

Faloutsos, C. and Chan, R. (1988). Fast Text Access Methods for Optical and Large Magnetic Disks: Designs and Performance Comparison, in: *Proc. 14th Intl. Conf. on VLDB,* Aug. 1988, pp. 280-293.

Faloutsos, C., Lee, R., Plaisant, C. and Shneiderman, B. (1990). Incorporating string search in hypertext system: User interface and signature file design issues. *Hyper-Media,* 2(3), 183-200.

Faloutsos, C., Ranganathan, M., & Manolopoulos, Y. (1994). Fast subsequence matching in time-series databases. *Proceedings of ACM SIGMOD,* 419-429. Minneapolis, Minnesota, USA.

Fan, W., Wallace, L., Rich, S., & Zhang, Z. (2006). Tapping the power of text mining. *Communications of the ACM, 49*(9), 76–82.

Favre, C., Bentayeb, F., & Boussaïd, O. (2007). *Dimension hierarchies updates in data warehouses: A user-driven approach.* Proceedings of the 9th International Conference on Enterprise Information Systems (ICEIS 07), Funchal, Madeira, Portugal.

Fayyad, U. et al. (1996). *Advances in Knowledge Discovery and Data Mining.* Merlo Park, CA: AAAI Press.

Fayyad, U. M., Piatetsky-Shapiro, G., Smyth, P., & Uthurusamy, R. (1996). *Advances in knowledge discovery and data mining.* AAAI Press/MIT Press, Menlo Park, California, USA.

Fayyad, U.M., Piatetsky-Shapiro, G., Smyth, P., & Uthurusamy, R. (1996). *From data mining to knowledge discovery: An overview.* AAAI/MIT Press.

Fdez-Riverola, F., & Corchado, J. M. (2003). Forecasting system for red tides: A hybrid autonomous AI model. *Applied Artificial Intelligence, 17*(10), 955–982.

Federal Geographic Data Committee. (1995). *Development of a national digital geospatial data framework.* Washington, DC: Author.

Feigenbaum, J., Ishai, Y., Malkin, T., Nissim, K., Strauss, M., & Wright R. (2001). Secure multiparty computation of approximations. *Proceedings of 28th International Colloquium on Automata, Languages and Programming* (pp. 927-938).

Feldman, Y. A., & Reouven, J. (2003). A knowledge-based approach for index selection in relational databases. *Expert System with Applications, 25*(1), 15-37.

Fellbaum, C. (1998). *WordNet: An electronic lexical database (language, speech, and communication).* The MIT Press.

Fensel, D., Bussler, C. (2002). The Web Service Modeling Framework WSMF. *Electronic Commerce Research and Applications*, Vol 1, No. 2. Elsevier Science

Feras, A. H. H. (2005). Integrity constraints maintenance for parallel databases. Ph.D. Thesis, Universiti Putra Malaysia, Malaysia.

Ferber, J. (1999). *Multi-agent systems: An introduction to distributed artificial intelligence.* 1st ed. Addison-Wesley Professional.

Fern, X., & Brodley, C. (2003). *Random Projection for High Dimensional Data Clustering: A Cluster Ensemble Approach.* Paper presented at the twentieth International Conference on Machine Learning (ICML'03).

Fernandez, M., Florescu, D., Kang, J., Levy, A., & Suciu, D. (1997). *STRUDEL: A Web site management system.* Proceedings of the ACM SIGMOD International Conference on Management of Data, 26, 549–552.

Ferragina, P., & Grossi, R. (1999). The string B-tree: A new data structure for string search in external memory and its applications. *Journal of the ACM, 46*(2), 236-280.

Ferraro, G. (2002). *Global brains: Knowledge and competencies for the 21st century.* Charlotte, NC: Intercultural Associates.

Fidalgo, R., Times, V., Silva, J., & Souza, F. (2004). *GeoDWFrame: A framework for guiding the design of geographical dimensional schemes.* Proceedings of the 6th International Conference on Data Warehousing and Knowledge Discovery, 26–37.

Fikes, R., McGuinness, D. L., & Waldinger, R. (2002, January). A first-order logic semantics for semantic Web markup languages [Knowledge Systems Laboratory Tech. Rep. No. 02-01]. Retrieved September 16, 2004, from http://www.ksl.stanford.edu/KSL_ Abstracts/KSL-02-01.html

Fileto, R. (2001). *Issues on interoperability and integration of heterogeneous geographical data.* Proceedings of the 3rd Brazilian Symposium on GeoInformatics.

Fillmore, C. (1968). The Case for Case, In Bach and Harms (Ed.), *Universals in Linguistic Theory.* New York: Holt, Rinehart, and Winston, (pp. 1-88).

Finkel, R. A., Marek, V. W., & Truszczynski, M. (2004). Constraint lingo: Towards high-level constraint programming. *Software: Practice and Experience* (pp. 1481-1504).

Finkelstein, S. J., Schkolnick, M., & Tiberio, P. (1988). Physical database design for relational databases. *ACM Transactions on Database Systems, 13*(1), 91-128.

Finkenzeller, K. (2003). *RFID handbook: Fundamentals and applications in contactless smart cards and identification* (2nd ed.). John Wiley & Sons.

Finnigan, P. (2002). SQL injection and Oracle, part one (Tech. Rep.). *Security Focus.* Retrieved August 30, 2006, from http://www.securityfocus.com/infocus/1644

Firat, A., Madnick, S., & Grosof, B. (2002). Knowledge integration to overcome ontological heterogeneity: Challenges from financial information systems. *Twenty-Third International Conference on Information Systems, ICIS.*

Fitting M (1985). A Kripke-Kleene semantics for logic programs. Journal of Logic Programming, 4:295–312.

Flajolet, P., & Martin, G. N. (1985). Probabilistic counting algorithms for data base applications. *Journal of Computer and Systems Science, 31*(2), 20-34.

Flesca, S., Manco, G., Masciari, E., Pontieri, L., & Pugliese, A. (2005). Fast detection of XML structural similarity. *IEEE Transactions on Knowledge and Data Engineering, 17*(2), 160-175.

Florescu, D., & Kossman, D. (1999). Storing and querying XML data using an RDMBS. *IEEE Data Engineering Bulletin, 22*(3), 27-34, 1999.

Florescu, D., Levy, A., & Mendelzon, A. (1998). Database techniques for the World-Wide Web: a survey. *SIGMOD Rec., 27*(3), 59-74.

Fo, R. (1992). Dependent origination and nature origination. *Dharmaghosa (The Voice of Dharma), 1992*(04), 7-15.

Fonseca, F. (2001). *Ontology-driven geographic information systems.* Unpublished doctoral dissertation, University of Maine, ME.

Fonseca, F., & Egenhofer, M. (1999). Ontology-driven information systems. *7th ACM Symposium on Advances in GIS* (pp. 14-19).

Fonseca, F., Davis, C., & Camara, C. (2003). Bridging ontologies and conceptual schemas in geographic information integration. *GeoInformatica, 7*(4), 307-321.

Fonseca, F., Egenhofer, M., Agouris, P., & Camara, C. (2002). Using ontologies for integrated geographic information systems. *Transactions in GIS, 3*(6).

Fonseca, F., Egenhofer, M., Davis, C., & Câmara, G. (2002). Semantic granularity in ontology-driven geographic information systems. *Annals of Mathematics and Artificial Intelligence, 36*(1-2), 131-151.

Fontana, E., & Dennebouy, Y. (1995). Schema Evolution by using Timestamped Versions and Lazy Strategy. *Onzièmes Journées Bases de Données Avancées,* 29 Août - 1er Septembre 1995, Nancy, France INRIA.

Forsberg, B. (2005, March 13). The future is South Korea: Technology firms try out latest in world's most wired society. *San Francisco Chronicle*. Retrieved November 3, 2005, from http://www.sfgate.com/cgi-bin/article.cgi?f=/c/a/2005/03/13/BROADBAND.TMP

Foster, I., & Kesselman, C. (Eds.). (1994). *The grid: Blueprint for a new computing infrastructure*. San Francisco: Morgan Kaufmann.

Foster, I., Kesselman, C., & Tuecke, S. (2001). The anatomy of the grid: Enabling scalable virtual organizations. *International Journal of Supercomputer Applications, 15*(3), 200-222.

Frakes, W., & Baeza-Yates, R. (1992). *Information retrieval: Data structures & algorithms*. Prentice Hall.

Franconi E., Grandi F., & Mandreoli F. (2000). *Schema Evolution and Versioning: A logical and Computational Characterisation at the 9th International Workshop on Foundations of Models and Languages for Data and Objects*, FoMLaDO/DEMM2000, Dagstuhl Castle, Germany, September 2000.

Franconi, E., Grandi, F., & Mandreoli, F. (2000). A general framework for evolving schemata support. *Proceedings of SEBD 2000*,.371-384.

Franconi, E., Kuper, G. M., Lopatenko, A., & Zaihrayeu, I. (2003). A robust logical and computational characterisation of peer to-peer database systems. *DBISP2* (pp. 64-76).

Frank, M. R., Omiecinski, E., & Navathe, S. B. (1992). Adaptive and automated index selection in RDBMS. In *Lecture notes in computer science: Vol. 580. 3rd International Conference on Extending Database Technology, (EDBT 1992)* (pp. 277-292). Berlin, Germany: Springer.

Franklin, S., & Graesser, A. (1996). Is it an Agent, or just a Program? A Taxonomy for Autonomous Agents. *Proceedings of ATAL'96*, (pp. 21-35).

Franz, M., & Kistler, T.(1997). Slim Binaries. *Communications of the ACM*, ACM, 40(12), 87-94.

Freitas, G. M., Laender, A. H. F., & Campos, M. L. (2002). MD2: Getting users involved in the development of data warehouse application. *Proceedings of the 4th International Workshop on Design and Management of Data Warehouses*, 3-12.

Frentzos, E. (2003). *Indexing objects moving on fixed networks*. Proceedings of the SSTD 2003, 289–305.

Frentzos, E. (2003). Indexing objects moving on fixed networks. *Proc. of the 8th International Symposium on Spatial and Temporal Databases (SSTD'2003)*, 289-305. Santorini, Hellas.

Freytag J., & Necib, C. (2004). Using ontologies for database query reformulation. *Proc. ADBIS Conference*.

Friedman, M., Levy, A., & Millstein, T. (1999). Navigational plans for data integration. *AAAI/IAAI*, 67-73.

Friedman, N., Getoor, L., Koller, D., & Pfeffer, A. (1999). *Learning probabilistic relational models*. Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence, Orlando, Florida, 1300–1309.

Frisch, A. M., & Allen, J. F. (1982). Knowledge retrieval as limited inference. *Proceedings of the 6th Conference on Automated Deduction* (pp. 274-291).

Frost & Sullivan. (2006). Location-based services offer high revenue potential in competitive mobile communications market. *Directions Magazine*. (January 17): Accessed November 10, 2006 at http://www.directionsmag.com/press.releases/index.php?duty=Show&id=13403&trv=1

Fung, B. Y. M., Ye, Y., & Zhang, L. (2003). Classification of heterogeneous gene expression data. *ACM-SIGKDD Explorations, 5*(2), 69-78.

Gadia, S. K (1988). A homogeneous relational model and query languages for temporal databases. *ACM Transactions on Database Systems, 13*(4), 418-448.

Gaede, V., & Gunther, O. (1998). Multidimensional Access Methods . *ACM Computing Surveys, 30*(2), 170-231.

Galanis, L., Wang, Y., Jeffery, S. R., & DeWitt, D. J. (2003). Locating data sources in large distributed systems. *Proceedings of the 29th International Conference on Very Large Data Bases* (pp. 874-885).

Galante, R. D. M., Edelweiss, N., et al. (2002). Change Management for aTemporal Versioned

Galitsky, B. A., Kuznetsov, S. O., & Vinogradov, D. V. (2005). *JASMINE: A hybrid reasoning tool for discovering causal links in biological data*. Retrieved from http://www.dcs.bbk.ac.uk/~galitsky/Jasmine.

Gallant, S. I. (1993). *Neural Network Learning and Expert Systems.* MIT press

Galperin, M. (2007). The molecular biology database collection: 2007 update. *Nucleic Acids Research, 35*(Database-Issue), 3-4.

Ganapathiraju, M. K. (2002). *Relevance of cluster size in MMR based summarizer.* Technical Report 11-742, Language Technologies Institute, Carnegie Mellon University, Pittsburgh.

Ganter B. & Wille R. (1999). Formal Concept Analysis: Mathematical Foundations. Springer, Heidelberg.

Ganter, B., & Wille, R. (1999). *Formal concept analysis: Mathematical foundations.* Springer.

Ganti, V., Lee, M., & Ramakrishnan, R. (2000). ICICLES: Self-tuning samples for approximate query answering. *Proceedings of the 26th International Conference on Very Large Data Bases* (pp. 176-187).

Gao, L., & Wang, X. S. (2002b). Improving the performance of continuous queries on fast data streams: Time series case. *SIGMOD/DMKD Workshop.* Madison, Wisconsin, USA.

Gao, L., & Wang, X. S.(2002). Continually evaluating similarity-based pattern queries on a streaming time series. *Proceedings of ACM SIGMOD*, 370-381. Madison, Wisconsin, USA.

Gao, L., Yao, Z., & Wang, X. S. (2002c). Evaluating continuous nearest neighbor queries for streaming time series via pre-fetching *Proceedings of CIKM,* 485-492. McLean, Virginia, USA.

Gao, X., & Hurson, A.R. (2005). Location Dependent Query Proxy. *Proc. of the 20th ACM Int. Symp. on Applied Computing,* (pp. 1120-1124).

Garcia – Molina, H., & Salem, K. (1992). **Main Memory Database Systems: An Overview.** IEEE Transactions on Knowledge and Data Engineering, 4(6), 509-516.

Garcia-Molina H., Labio W., Wiener L. J., & Zhuge, Y. (1998). Distributed and Parallel Computing Issues in Data Warehousing. *Proceedings of the Seventeenth Annual ACM Symposium on Principles of Distributed Computing,* (p. 7).

Garfinkel, S., & Rosenberg, B. (2005). *RFID: Applications, security, and privacy.* Addison-Wesley Professional.

Garofalakis, et al. (1999). SPIRIT: Sequential Pattern Mining with Regular Expression Constraints. *Proceedings of the 25th International Conference on Very Large Data Bases Conference*, UK, (pp. 223-234).

Garofalakis, M. N., & Gibbons, P. B. (2002). Wavelet synopses with error guarantees. *Proceedings of the 2002 ACM International Conference on Management of Data* (pp. 476-487).

Garofalakis, M. N., Gehrke, J., & Rastogi, R. (2002). Querying and mining data streams: You only get one look. *Proceedings of the 28th International Conference on Very Large Data Bases* (p. 635).

Garofalakis, M.N., & Kumar, A. (2004). Deterministic Wavelet Thresholding for Maximum-Error Metrics. *Proc. of the 23rd ACM Int. Symp. on Principles of Database Systems,* (pp. 166-176).

Garrett, J. (2005). Ajax: a new approach to web applications. AdaptivePath. Retrieved October 8, 2007, from *http://www.adaptivepath.com/publications/essays/archives/000385.php*

Gates, B. (2005). Building software that is interoperable by design [technical report]. Microsoft Corporation Web site. Retrieved December 2006, from http://www.microsoft.com/mscorp/execmail/2005/02-03interoperability.mspx.

Gauch, S., Wang, J., & Rachakonda, S. M. (1999). A corpus analysis approach for automatic query expansion and its extension to multiple databases. *ACM Transactions on Information Systems, 17*(3), 250-269.

GDE geographic data exchange standards. (2002). Retrieved October 2006 from http://www.iecapc.jp/06/diffusenew/standards/gis.html

Geerts, G. L., & McCarthy, W. E. (1991). Database accounting systems. IT and Accounting: The Impact of Information Technology. In B. C. Williams & B. J. Spaul (Eds.), London: Chapman & Hall, (pp. 159-183).

Geerts, G. L., & McCarthy, W. E. (1999). An accounting object infrastructure for knowledge-based enterprise models. IEEE Intelligent Systems & Their Applications, (July-August), (pp. 89-94).

Geerts, G. L., & McCarthy, W. E. (2000). Augmented intensional reasoning in knowledge-based accounting systems. Journal of Information Systems, 14(2), 127-150.

Geerts, G. L., & McCarthy, W. E. (2001). Using object templates from the REA accounting model to engineer business processes and tasks. The Review of Business Information Systems, 5(4), 89-108.

Geerts, G. L., & McCarthy, W. E. (2002). An ontological analysis of the primitives of the extended-REA enterprise information architecture. International Journal of Accounting Information Systems, 3, 1-16.

Geerts, G. L., & McCarthy, W. E. (2004). The ontological foundation of REA enterprise information systems. Working paper, Michigan State University.

Gehrke, J., & Madden, S. (2004). Query Processing in Sensor Networks. *IEEE Pervasive Computing, 3*(1), 46-55.

Gelfond, M., & Lifschitz, V. (1988). The stable model semantics for logic programming. *ICLP* (pp. 1070-1080).

Gelfond, M., & Lifschitz, V. (1991). Classical negation in logic programs and disjunctive databases. *New Generation Computing, 9*(3/4), 365-385.

Gen Yee, W., & Frieder, O. (2005). On search in peer-to-peer file sharing systems. *Proceedings of the 20th ACM Symposium on Applied Computing* (pp. 1023-1030).

Gennari, J., Musen, M. A., Fergerson, R. W., Grosso, W. E., Crubezy, M., Eriksson, H., et al. (2002). The evolution of protege: An environment for knowledge-based systems development. *International Journal of Human-Computer Studies.*

Georgakopoulos, D. (2004). Teamware: An evaluation of key technologies and open problems. *Distributed and Parallel Databases, 15*(1), 9-44.

Georgakopoulos, D., Hornick, M., & Manola, F. (1996). Customizing transaction models and mechanisms in a programmable environment supporting reliable workflow automation. *IEEE Transactions on Data and Knowledge Engineering, 8*(4), 630-649.

George, B., Kim, S., & Shekhar, S. (2007). Evaluation of Iceberg Distance Joins. *In Proceedings of the 10th International Symposium on Spatial and Temporal Databases (SSTD 2007)*, Boston, MA, USA, July 16-18, 2007, Lecture Notes in Computer Science 4605, (pp. 460-477), Springer.

Georgeff, M. P., & Lansky, A. L. (1987). Reactive Reasoning and Planning. In *Proc. of the Sixth National Conference on Artificial Intelligence* (*AAAI-87*), (pp. 677-682).

Gerard, G. J. (2005). The REA pattern, knowledge structures, and conceptual modeling performance. Journal of Information Systems, 19, 57-77.

Getta, J.R. (2000). *Query scrambling in distributed multidatabase systems.* Proceedings of the 11th International Workshop on Database and Expert Systems Applications, DEXA' 2000, 647–652.

Getta, J.R. (2005). *On adaptive and online data integration.* Proceedings of the 21st International Conference on Data Engineering, International Workshop on Self-Managing Database Systems, 1212–1220.

Getta, J.R., & Vossough, E. (2004). Optimization of data stream processing. *SIGMOD Record, 33,* 34–39.

Gibbons, P. B. (2001). Distinct sampling for highly-accurate answers to distinct values queries and event reports. *Proceedings of the 27th International Conference on Very Large Data Bases* (pp. 13-24).

Gibbons, P. B., & Matias, Y. (1998). New sampling-based summary statistics for improving approximate query answers. *Proceedings of the 1998 ACM International Conference on Management of Data* (pp. 331-342).

Gibbons, P. B., Karp, B., Ke, Y., Nath, S., & Seshan, S. (2003). IrisNet: An architecture for a world-wide sensor web. *IEEE Pervasive Computing, 2*(4), 22-33.

Gibbons, P. B., Matias, Y., & Poosala, V. (1997). Fast incremental maintenance of approximate histograms. *Proceedings of the 23rd International Conference on Very Large Data Bases* (pp. 466-475).

Gibbons, P.B., & Matias, Y. (1998). New Sampling-Based Summary Statistics for Improving Approximate Query Answers. *Proc. of the 1998 ACM Int. Conf. on Management of Data,* (pp.331-342).

Gifford, D. K. (1979). Weighted Voting for Replicated Data. *7th ACM Symposium on Operating System Principles,* (pp. 150-162).

Gilbert, A. C., Kotidis, Y., Muthukrishnan, S., & Strauss, M. J. (2003). One-pass wavelet decompositions of data streams. *IEEE TKDE, 15*(3), 541-554.

Gillette, D. (1999, December). Web design for international audiences. *Intercom,* pp. 15-17.

Giorgini, P., Rizzi, S., & Garzetti, M. (2005). Goal-oriented requirements analysis for data warehouse design. *Proceedings of the 8h ACM International Workshop on Data Warehousing and OLAP*, 47-56.

Giunchiglia, F., Yatskevich, M., & Giunchiglia, E. (2005). Efficient semantic matching. *In A. Gomez-Perez and J. Euzenat, editors, ESWC 2005, volume LNCS 3532*, 272-289. Springer-Verlag.

Glenisson, P., Mathys, J., & De Moor, B. (2003). Meta-clustering of gene expression data and literature based information. *SIGKDD Explorations, 5*(2), 101-112.

Godfrey, P., & Gryz, J. (1999). Answering queries by semantic caches. In T. Bench- Capon, G. Soda, and A. M. Tjoa (Eds.), *Database and Expert Systems Applications: Proceedings of the 10th International Conference, DEXA'99*, Florence, Italy, August/September 1999, volume 1677 of Lecture Notes in Computer Science, (pp. 485–498), Heidelberg. Springer-Verlag.

Gogolla, M., & Hohenstein, U. (1991). Towards a Semantic View of an Extended Entity Relationship Model. *ACM Transactions on Database Systems, 16*(3), 369-416.

Gogolla, M., Meyer, B., & Westerman, G. D. (1991). Drafting Extended EntityRelationship Schemas with QUEER. In T. J. Teorey, (Ed.), *Proc. 10th Int. Conf. on Entity-Relationship Approach*, (pp. 561-585).

Goh, C. H. (1996). *Representing and reasoning about semantic conflicts in heterogeneous information sources.* Phd, MIT, Massachusetts Institute of Technology, Sloan School of Management.

Goh, C. H., Bressan, S., Siegel, M., & Madnick, S. E. (1999). Context interchange: New features and formalisms for the intelligent integration of information. *ACM Transactions on Information Systems, 17*(3), 270-293.

Goil, S., Nagesh, H., & Choudhary, A. (1999). *MAFIA: Efficient and scalable subspace clustering for very large data sets.* Technical Report CPDC-TR-9906-010: Northwestern University.

Gold, C. M. (2006). What is GIS and what is not? *Transactions in GIS, 10*(4), 505-519.

Goldman, R., & Widom, J. (1997). *DataGuides: Enabling query formulation and optimization in semistructured databases.* Proceedings of the 23rd International Conference on Very Large Data Bases (VLDB 97), Athens, Greece, 436–445.

Goldreich, O. (2004). *Foundations of cryptography* (Vol. I, II). Cambridge University Press.

Goldreich, O., Micali, S., & Wigderson, A. (1987). How to play any mental game. *Proceedings of 19th Annual ACM Conference on Theory of Computing (STOC'87)* (pp. 218-229).

Goldstein, J., & Larson, P. A. (2001). Optimizing queries using materialized views: A practical, scalable solution. *ACM SIGMOD International Conference on Management of Data (SIGMOD 2001)* (pp. 331-342).

Golfarelli, M., & Rizzi, S. (1998). A methodological framework for data warehouse design. *Proceedings of the 1st ACM International Workshop on Data Warehousing and OLAP*, 3-9.

Golfarelli, M., Lechtenborger, J., Rizzi, S., & Vossen, G. (2006). Schema versioning in datawarehouses: Enabling cross-version querying via schema augmentation. *Data and Knowledge Engineering, 59*(2), 435–459.

Golfarelli, M., Lechtenborger, J., Rizzi, S., & Vossen, G. (2006). Schema versioning in datawarehouses: Enabling cross-version querying via schema augmentation. *Data and Knowledge Engineering, 59*(2), 435–459.

Golfarelli, M., Rizzi, S., & Saltarelli, E. (2002). Index selection for data warehousing. *CEUR Workshop Proceedings: Vol. 58. 4th International Workshop on Design and Management of Data Warehouses (DMDW 2002)* (pp. 33-42).

Golub, G. H., & Loan, C. F. V. (1996). *Matrix computation.* The John Hopkins University Press.

Gómez-Pérez, A., González-Cabrero, R., & Lama, M. (2004). ODE SWS: A framework for designing and composing Semantic Web services. *IEEE Intelligent Systems, 19*(4), 24-31.

Gong, X., Yan, Y., Qian, W., & Zhou, A. (2005). Bloom filter-based XML packets filtering for millions of path queries. *Proceedings of the 21st IEEE International Conference on Data Engineering* (pp. 890-901).

González Císaro, S., & Nigro, H. O. (2008). Architecture for Symbolic Object Warehouse. In J. Wang (Ed.), *Encyclopedia of Data Warehousing and Mining - 2nd Edition.* Hershey PA: Idea Group Publishing.

Goodchild, M., Egenhofer, M., Fegeas, R., & Kottman, C. (1999). Interoperating geographic information systems. Norwell: Kluwer Academic Publishers.

Goodman, D., Borras, J., Mandayam, N., & Yates, R. (1997). INFOSTATIONS: A New System Model for Data and Messaging Services. *Proc. of the IEEE Vehicular Technology Int. Conf.,* 2, 969-973.

Gorla, N., & Lam, Y.W. (2004). Who should work with whom? Building effective software project teams. *Communications of the ACM, 47*(6), 79–82.

Gottgtroy P. et al. (2005). Enhancing data analysis with Ontologies and Olap. *Proceedings Data Mining 2005 Conference*, Skialhos, Grecia.

Govindan, R., Hellerstein, J., Hong, W., Madden, S., Franklin, M., & Shenker, S. (2002). *The sensor network as a database* (Tech. Rep. No. 02-771). University of Southern California, Los Angeles, CA, 2002.

Grabmeier, J., & Rudolph, A. (2002). Techniques of Cluster Algorithms in Data Mining. *Data Mining and Knowledge Discovery, 6*(4), 303-360.

Grandi, F., Mandreoli, F., & Tiberio, P. (2005). *Data and Knowledge Engineering, 54*(3), 327-354.

Granell, C., Gould, M., & Esbrí, M.A. (2007a). Geospatial Web service chaining. In H. Karimi (Ed.), *Handbook of research on geoinformatics.* Hershey: Information Science Reference.

Granell, C., Gould, M., Manso, M.A., & Bernabé, M.A. (2007b). Spatial data infrastructures. In H. Karimi (Ed.), *Handbook of research on geoinformatics.* Hershey: Information Science Reference.

Grant J. and Subrahmanian V.S.(2000). Applications of Paraconsistency in Data and Knowledge Bases, Synthese 125:121-132.

Grant, E. S., Cennamaneni, R., & Reza, H. (2006). Towards analyzing UML class diagram models to object-relational database systems transformation. In *Proceedings of the 24th IASTED International Multi-Conference Databases and Applications* (pp. 129-134). Innsbruck: ACTA Press.

Grant, J., & Hunter, A. (2006). Measuring inconsistency in knowledge bases. *Journal of Intelligent Information Systems, 27,* 159-184.

Grant, J., & Subrahmanian, V. S. (1995). Reasoning in inconsistent knowledge bases. *IEEE-TKDE, 7*(1), 177-189.

Grant, J., & Subrahmanian, V. S. (2000). Applications of paraconsistency in data and knowledge bases. *Synthese, 125,* 121-132.

Gray, J., & Reuter, A. (1993). *Transaction Processing: Concepts and Techniques.* Morgan Kaufmann.

Gray, J., Bosworth, A., Layman, A., & Pirahesh, H. (1997). Data Cube: A Relational Aggregation Operator Generalizing Group-By, Cross-Tab and Sub-Totals. *Proc. of the 12nd IEEE Int. Conf. on Data Engineering,* (pp. 152-159).

Gray, J., Helland, P., O'Neil, P. E., & Shasha, D. (1996). The Dangers of Replication and a Solution. In H. V. Jagadish, & I. S. Mumick (Eds.), *Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data*, Montreal, Quebec, Canada, June 4-6, 1996, *SIGMOD Record, 25*(2),173–182. ACM Press, New York, NY, USA, Juni 1996.

Gray, J., Helland, P., O'Neil, P., & Shasha, D. (1996). The Dangers of Replication and a Solution. *ACM SIGMOD Conference,* (pp. 173-182).

Gray, P. M. D., Kerschberg, L., King, P., & Poulovassilis, A. (Eds.). (2004). *The functional approach to data management: Modeling, analyzing, and integrating heterogeneous data.* Heidelberg, Germany: Springer.

Gray, P. M. D., King, P. J. H., & Kerschberg, L. (Eds.). (1999). Functional approach to intelligent information systems. *Journal of Intelligent Information Systems, 12,* 107-111.

Greco S., & Zumpano, E. (2000) Querying inconsistent databases. *Proceedings of the 7th International Conference of Logic for Programming and Automated Reasoning (LPAR 2000), Lecture Notes in Computer Science, 1955* (pp.308-325).

Greco, G., Greco, S., & Zumpano, E. (2001). A logic programming approach to the integration, repairing and querying of inconsistent databases. *ICLP* (pp. 348-364).

Greco, S., & Saccà, D. (1997). NP-optimization problems in datalog. *International Logic Programming Symposium* (pp. 181-195).

Greco, S., & Zumpano, E. (2000). Querying inconsistent database. *LPAR* (pp. 308-325).

Greco, S., Molinaro, C., Trubitsyna, I., & Zumpano, E. (2006). Implementation and experimentation of the logic language NP datalog. *International Conference on Database and Expert Systems Applications* (pp. 622-633).

Greco, S., Pontieri, L., & Zumpano, E. (2001). Integrating and managing conflicting data. *Ershov Memorial Conference* (pp. 349-362).

Grefen, P. W. P. J. (1993). Combining theory and practice in integrity control: A declarative approach to the specification of a transaction modification subsystem. *Proceedings of the 19th International Conference on Very Large Data Bases*, Dublin, (pp. 581-591).

Grefen, P.W.P.J. (1990). *Design considerations for integrity constraint handling in PRISMA/DB1.* Prisma Project Document P508. The Netherlands.

Grefen, P.W.P.J. (1993). *Combining theory and practice in integrity control: A declarative approach to the specification of a transaction modification subsystem.* Proceedings of the 19th International Conference on Very Large Data Bases, Ireland, 581–591.

Gribble, S., Halevy, A., Ives, Z., Rodrig, M., & Suciu, D. (2001). What can databases do for peer-to-peer? *WebDB* (pp. 31-36).

Grossman, L.R., & Gou, Y. (2001). Parallel methods for scaling data mining algorithms to large data sets. In J.M. Zytkow (Ed.), *Handbook on data mining and knowledge discovery.* Oxford University Press.

Gruber, T. (1993). A translation approach to portable ontology specifications. *Knowledge Acquisition, 5*(2), 199–220.

Gruber, T. (1993). Towards principles for the design of ontologies used for knowledge sharing. *Proc of International Workshop on Formal Ontology*, Padova, Italy.

Gruber, T. (2002). *What is an ontology?* Retrieved 2006 from http://www.ksl.stanford.edu/kst/what-is-an-ontology.html

Gruber, T. (2008). Ontology. In Ling Liu and M. Tamer Özsu (Eds.) Encyclopedia of Database Systems, Springer-Verlag. Found online at http://tomgruber.org/writing/ontology-definition-2007.htm (accessed 10/5/07).

Grün, K., Karlinger, M., & Schrefl, M. (2006). Schema-aware labelling of XML documents for efficient query and update processing in SemCrypt. *International Journal of Computer Systems, Science, and Engineering, 21*(1), 65-82.

Guarino, N. (1998). Formal ontology in information systems. *Formal Ontology in Information Systems: Proceedings of FOIS'98* (pp. 3-15).

Guha, S., Meyerson, A., Mishra, N., Motwani, R., & O'Callaghan, L. (2003). Clustering data streams: Theory and practice. *IEEE TKDE, 15*(3), 515-528.

Guha, S., Meyerson, A., Mishra, N., Motwani, R., & O'Callaghan, L. (2003). Clustering Data Streams: Theory and Practice. I*EEE Transactions on Knowledge and Data Engineering, 15*(3), 515-528.

Guha, S., Rastogi, R., & Shim, K. (1998). *CURE: an efficient clustering algorithm for large databases.* Paper presented at SIGMOD '98: the 1998 ACM SIGMOD international conference on Management of data, New York, NY, USA.

Guha, S., Rastogi, R., & Shim, K. (1999). *ROCK: A Robust Clustering Algorithm for Categorical Attributes.* Paper presented at the 15th International Conference on Data Engineering, 23-26 March 1999,

Guillaume, D., & Murtagh, F. (2000). Clustering of XML documents. *Computer Physics Communications, 127*, 215-227.

Guizzardi, G., Herre, H., & Wagner, G. (2002). *Towards Ontological Foundations for UML conceptual models.* ODBASE 2002.

Gummadi, P. K., Dunn, R. J., Saroiu, S., Gribble, S. D., Levy, H. M., & Zahorjan, J. (2003). Measurement, modeling, and analysis of a peer-to-peer file-sharing workload. *Proceedings of the 19th ACM Symposium on Operating Systems Principles* (pp. 314-329).

Gündem, T. I. (1999). Near optimal multiple choice index selection for relational databases. *Computers & Mathematics with Applications, 37*(2), 111-120.

Gunopulos, D., Kollios, G., Tsotras, V.J., & Domeniconi, C. (2000). Approximating Multi-Dimensional Aggregate Range Queries over Real Attributes. *Proc. of the 2000 ACM Int. Conf. on Management of Data*, (pp. 463-474).

Guo, S., Sun, W., & Weiss, M. A. (1996). Solving satisfiability and implication problems in database systems. *ACM Transactions on Database Systems (TODS), 21*(2), 270–293.

Gupta, A. (1994). *Partial information based integrity constraint checking* [doctoral thesis]. Stanford University.

Gupta, A. (1994). *Partial information based integrity constraint checking.* Ph.D. Thesis, Stanford University.

Gupta, A., & Mumick, I. S. (Eds.) (1999). *Materialized views: Techniques, implementations, and applications.* MIT Press.

Gupta, A., Agrawal, D., & El Abbadi, A. (2003). Approximate range selection queries in peer-to-peer systems. *Proceedings of the 1st Biennial Conference on Innovative Data Systems Research*. Retrieved from http://www.db.cs.wisc.edu/cidr/cidr2003/program/p13.pdf

Gupta, A., Mumick, I. S., & Subrahmanian, V. S. (1993). Maintaining views incrementally. *ACM SIGMOD Record, 22*(2), 157–166

Gupta, H., & Mumick, I. S. (2005). Selection of views to materialize in a data warehouse. *IEEE Transactions on Knowledge and Data Engineering, 17*(1), 24-43.

Gupta, R., Haritsa, J. R., & Ramamritham, K. (1997). More optimism about real-time distributed commit processing (Technical Report TR – 97 – 04). *Database System Lab, Supercomputer Education and Research Centre, I.I.Sc. Bangalore, India*.

Guptill, S. C., & Morrison, J. L. (1995). *Elements of spatial data quality.* Oxford: Elsevier.

Gustafsson, T., Hallqvist, H., & Hansson, J. (2005). *A similarity-aware multiversion concurrency control and updating algorithm for up-to-date snapshots of data.* Proceedings of the 17th Euromicro Conference on Real-Time Systems, 229–238.

Gutiérrez, G. A., Navarro, G., Rodríguez, A., González, A. F., & Orellana, J. (2005). A spatio-temporal access method based on snapshots and events. *Proc. Of ACM GIS*, 115-124.

Gutiérrez-Naranjo, M. A., Alonso-Jiménez, J. A., & Borrego- Díaz, J. (2003). A quasimetric for machine learning. In F. J. Garijo, J. C. Riquelme, & M. Toro (Eds.), *Advances in Artificial Intelligence (IBERAMIA 2002), Lecture Notes in Computer Science, 2527* (pp. 193-203).

Güting, R. (1994). An Introduction to Spatial Database Systems. *VLDB Journal, 3*(4), 357-399.

Güting, R. H., & Schneider, M. (2005). Moving objects databases. Morgan Kaufmann Publishers.

Güting, R. H., Almeida, V. T. de, & Ding, Z. (2006). Modeling and querying moving objects in networks. *VLDB Journal, 15*(2), 165-190.

Güting, R.H., De Almeida, V.T., & Ding, Z. (2006). Modeling and querying moving objects in networks. *VLDB J., 15*(2), 165–190.

Guttman, A. (1984). R-trees: A Dynamic Index Structure for Spatial Searching. *In Proceedings of the 1984 ACM SIGMOD International Conference on Management of Data (SIGMOD 1984)*, Boston, Massachusetts, 18-21 June, 1984, (pp. 47-57), ACM Press.

Guttman, A. (1984). R-trees: A dynamic index structure for spatial searching. *Proceedings of ACM SIGMOD International Conference on Management of Data* (pp. 47-54).

Gyssens, M., & Lakshmanan, L. V. S. (1997). A foundation for multi-dimensional databases. *VLDB'97* (pp. 106-115).

Gyssens, M., Paredaens, J., & Gucht, D. van. (1990). A graph-oriented object database model. *Proceedings of 1990 ACM Conference on Principles of Database Systems* (pp. 417-424).

Haarslev, V., & Möller, R. (2003, October 20). Racer: A core inference engine for the Semantic Web. In *Proceedings of the 2nd International Workshop on Evaluation of Ontology Tools (EON2003)*, Sanibel Island, FL..

Haas, L. M., Lin, E. T., & Roth, M. A. (2002). Data Integration through Database Federation. *IBM Systems Journal, 41*(4), 578-595.

Haas, P.J., & Hellerstein, J.M. (1999). *Ripple joins for online aggregation.* Proceedings of the ACM SIGMOD International Conference on Management of Data, 287–298.

Hadjieleftheriou, M., Kollios, G., Tsotras, V. J., & Gunopoulos, D. (2006). Indexing spatiotemporal archives. *VLDB Journal, 15*(2), 143-164

Hage, C., Jensen, C. S., Pedersen, T. B., Speičys, L., & Timko, I. (2003). Integrated data management for mobile services in the real world. *Proceedings of 29th International Conference on Very Large Databases* (pp. 1019-1030).

Hahn, U., Jarke, M., & Rose, T. (1991). Teamwork support in a knowledge-based information systems environment. *Transactions on Software Engineering, 17*(5), 467–482.

Hailpern, B., & Tarr, P. (2006). Model-driven development: The good, the bad, and the ugly. *IBM System Journal, 45*(3).

Hainaut, J.-L. (2002). *Introduction to database reverse engineering.* Retrieved February 2007 from http://www.info.fundp.ac.be/~dbm/publication/2002/DBRE-2002.pdf

Hainaut, J.-L. (2006). The transformational approach to database engineering. In R. Lämmel, J. Saraiva, & J. Visser (Eds.), *Lecture notes in computer science: Vol. 4143. Generative and transformational techniques in software engineering* (pp. 89-138). Springer-Verlag.

Hainaut, J.-L., Chandelon, M., Tonneau, C., & Joris, M. (1993, May). Contribution to a theory of database reverse engineering. Proceedings of the IEEE Working Conference on Reverse Engineering, Baltimore.

Hainaut, J.-L., Chandelon, M., Tonneau, C., & Joris, M. (1993b). Transformation-based database reverse engineering. *Proceedings of the 12th International Conference on Entity-Relationship Approach* (pp. 1-12).

Hainaut, J.-L., Roland, D., Hick J.-M., Henrard, J., & Englebert, V. (1996). Database reverse engineering: From requirements to CARE tools. *Journal of Automated Software Engineering, 3*(1).

Hakimpour, F. (2003). *Using ontologies to resolve semantic heterogeneity for integrating spatial database schemata* [doctoral thesis]. Zurich University.

Hakimpour, F., & Geppert, A. (2002). *Global schema generation using formal ontologies.* Proceedings of the ER2002, LNCS 2503, 307–321.

Hakimpour, F., & Timpf, S. (2001, April). *Using ontologies for resolution of semantic heterogeneity in GIS.* 4th AGILE Conference on Geographic Information Science, Brno, Czech Republic.

Halaschek, C., Aleman-Meza, B., Arpinar, I. B., & Sheth, A. P. (2004). Discovering and ranking semantic associations over a large RDF metabase. *Proceedings of the 30th International Conference on Very Large Data Bases* (pp. 1317-1320).

Halevi, S., Krauthgamer, R., Kushilevitz, E., & Nissim, K. (2001). Private approximation of NP-hard functions. *Proceedings of the 33rd Annual ACM Symposium on Theory of Computing (STOC'01)* (pp. 550-559).

Halevy, A. Y., Ives, Z. G., Mork, P., & Tatarinov, I. (2003). Piazza: Data management infrastructure for semantic Web applications. *Proceedings of the 12th International World Wide Web Conference* (pp. 556-567).

Halevy, A., Ives, Z., Suciu, D., & Tatarinov, I. (2003). Schema mediation in peer data management systems. *ICDT* (pp. 505-516).

Halevy, A.Y. (2001). Answering queries using views: A survey. *VLDB J, 10*(4), 270–294.

Halkidi, M., Batistakis, Y., & Vazirgiannis, M. (2001). On Clustering Validation Techniques. *Journal of Intelligent Information Systems, 2001, 17,* 107-145.

Hammer, J., Garcia-Molina, H., Ireland, K., Papakonstantinou, Y., Ullman, J., & Widom, J. (1995). *Information translation, mediation, and mosaic-based browsing in the TSIMMIS system.* Proceedings of the ACM SIGMOD International Conference on Management of Data, San Jose, California, 483.

Han J., & Kamber M. (2006). Data Mining: Concepts and Techniques. Morgan Kaufmann Publishers.

Han J., Cheng H., Xin D., & Yan X. (2007). Frequent Pattern Mining: Current Status and Future Directions. Journal of Data Mining and Knowledge Discovery, 15:55-86.

Han J., Pei J. & Yin Y. (2000). Mining Frequent Patterns without Candidate Generation. Proceedings 2000 ACM-SIGMOD International Conference Management of Data (SIGMOD '00), Dallas, TX.

Han, J., & Kamber, M. (2000). *Data mining: concepts and techniques.* San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.

Han, J., et al. (1996). *DBMiner: A system for mining knowledge in large relational databases.* Proceedings of the International Conference on Data Mining and Knowledge Discovery (KDD'96), Portland, Oregon, 250–255.

Hanandeh, F.A.H. (2006). *Integrity constraints maintenance for parallel databases* [doctoral thesis]. Malaysia: Universiti Putra Malaysia.

Hanandeh, F.A.H., Ibrahim, H., Mamat, A., & Johari, R. (2004). Virtual rule partitioning method for maintaining database integrity. *The International Arab Journal of Information Technology, 1*(1), 103–108.

Hanczar, B., Courtine, M., Benis, A., Hennegar, C., Clement, K., & Zucker, J.-D. (2003). Improving classification of microarray data using prototype-based feature selection. *ACM-SIGKDD Explorations, 5*(2), 23-30.

Harinarayan, V., Rajaraman, A., & Ullman, J. D. (1996). Implementing data cubes efficiently. *ACM SIGMOD International Conference on Management of Data (SIGMOD 1996)* (pp. 205-216).

Haritsa, J. R., Ramamritham, K. & Gupta, R. (2000). The PROMPT Real-time Commit Protocol. IEEE Transactions on Parallel and Distributed Systems, 11(2), 160-181.

Harren, M., Hellerstein, J., Huebsch, R., Loo, B. T., Shenker, S., & Stoica, I. (2002). Complex queries in DHT-based peer-to-peer networks. *Proceedings of the International Workshop on Peer-to-Peer Systems (IPTPS), 2002.*

Harvey, F. (1997). *Quality Needs More Than Standards. Data Quality in Geographic Information: From Error to Uncertainty.* M. Goodchild & R. Jeansoulin (Eds.). Paris: Hermès. (pp. 37-42).

Hass, L, & Lin, E. (2002). IBM federated database technology. Retrieved August 01, 2005, from http://www-106.ibm.com/developerworks/db2/library/techarticle/0203haas/0203haas.html.

Haugen, R., & McCarthy, W. E. (2000). REA: A semantic model for internet supply chain collaboration. Presented at Business Object Component Design and Implementation Workshop VI: Enterprise Application Integration which is part of The ACM Conference on Object-Oriented Programming, Systems, Languages, and Applications, October 15-19, Minneapolis, Minnesota.

Hawkins, D. (1980). *Identification of outliers.* London: Chapman & Hall.

Hawthorn, P., Simons, B., et al. (2006). *Statewide Databases of Registered Voters: Study Of Accuracy, Privacy, Usability, Security, and Reliability Issues.* U.S. Public Policy Committee of the Association for Computing Machinery, Washington, D.C., U.S.

Hayes, N., & Walsham, G. (1999). Safe Enclaves, Political Enclaves and Knowledge Working. *First International Critical Management Studies Conference*, Manchester, England.

Hayes, P. (Ed.). (2004). RDF semantics. *W3C Recommendation.* Retrieved from http://www.w3.org/TR/2004/REC-rdf-mt-20040210

He, H., & Yang, J. (2004). *Multiresolution indexing of XML for frequent queries.* Proceedings of the 20th International Conference on Data Engineering (ICDE 04), Boston, Massachusetts, 683–694.

Hearst, M. A. (1997). TextTiling: Segmenting text into multi-paragraph subtopic passages. *Computational linguistics, 23*(1), 33-46.

Heeseok, L., & Cheonsoo, Y. (2000). A form driven object-oriented reverse engineering methodology. *Information Systems, 25*(3), 235-259.

Heidemann, J., Silva, F., Intanagonwiwat, C., Govindan, R., Estrin, D., & Ganesan, D. (2001). Building efficient wireless sensor networks with low-level naming. *Proceedings of the Symposium on Operating Systems Principles (SOSP), 2001,* 146-159.

Hellerstein, J. M., Haas, P. J., & Wang, H. J. (1997). Online aggregation. *Proceedings of the 1997 ACM International Conference on Management of Data* (pp. 171-182).

Hemrich, M. (2002). A New Face for Each Show: Make Up Your Content by Effective Variants Engineering. In *XML Europe 2002.* Available at http://www.idealliance.org/papers/xmle02/ (verified on September 7, 2004).

Henrard, J. (2003). *Program understanding in database reverse engineering.* Unpublished doctoral dissertation, University of Namur. Retrieved February 2007 from http://www.info.fundp.ac.be/~dbm/publication/2003/jhe_thesis.pdf

Henschen, L. J., McCune, W. W., & Naqvi, S. A. (1984). Compiling constraint-checking programs from first-order formulas. *Advances in Database Theory, 2,* 145-169, Plenum Press.

Henschen, L.J., McCune, W.W., & Naqvi, S.A. (1984). Compiling constraint-checking programs from first-order formulas. *Advances in Database Theory, 2,* 145–169.

Hess, G. N., Iochpe, C., & Oliveira, J. P. M. (2004). Analysis patterns for the geographic database conceptual

schema: An ontology aided approach. *Proceedings of the International Conference on Semantics of a Networked World (ICSNW)* (pp. 323-324).

Hess, G.N., & Iochpe, C. (2002). *Ontology-driven resolution of semantic heterogeneities in GDB conceptual schemas.* Proceedings of the GEOINFO'04: VI Brazilian Symposium on GeoInformatics, Campos do Jordao, 247–263.

Hewitt, C., & de Jong, P. (1982). Open Systems. *On Conceptual Modelling (Intervale),* (pp. 147-164).

Hey, T., & Trefethen, A.E. (2005). Cyberinfrastructure for e-science. *Science, 308*(5723), 817–821.

Hibbs, M. A., Dirksen, N. C., Li, K., & Troyanskaya, O. G. (2005). Visualization methods for statistical analysis of microarray clusters. *BMC Bioinformatics, 6*(115), 10.

Hick J. M., & Hainaut J. L. (2003). Strategy for Database Application Evolution: The DB-MAIN approach. *Lecture Notes in Computer Science, 2813*, 291-306. Springer-Verlag Publishing.

Hick, J.-M. (2001). *Evolution of relational database applications.* Unpublished doctoral dissertation, University of Namur. Retrieved February 2007 from http://www.info.fundp.ac.be/~dbm/publication/2001/these-jmh.pdf

Hickey, T. (2000). Constraint-based termination analysis for cyclic active database rules. *Proc. DOOD'2000: 6th. International Conference on Rules and Objects in Databases, LNAI, 1861,* 1121-1136.

Hicks, D. L., Leggett, J. J., Nürnberg, P. J., & Schnase, J. L. (1998). A Hypermedia Version Control Framework. *ACM Transactions on Information Systems, 16*(2), 127-160.

Hill, T., & Lewicki, P. (2007). *Statistics: Methods and Applications.* Tulsa, OK: StatSoft.

Hinneburg, A., & Keim, D. A. (1998). *An Efficient Approach to Clustering in Large Multimedia Databases with Noise.* Paper presented at the fourth International Conference on Knowledge Discovery and Data Mining (KDD'98).

Hinneburg, A., & Keim, D. A. (1999). *Optimal Grid-Clustering: Towards Breaking the Curse of Dimensionality in High-Dimensional Clustering.* Paper presented at the VLDB'99: the 25th International Conference on Very Large Data Bases, September 7-10, 1999, Edinburgh, Scotland, UK.

Hintikka, J., Halonen, I., and Mutanen, A. (2002). Interrogative Logic as a General Theory of Reasoning', in *Handbook of the logic of argument and inference. The turn towards the practical, 1,* 295-337.

Hirano, A., Welch, R., & Lang, H. (2002). *Mapping from ASTER stereo image data: DEM validation and accuracy assessment*. Photogrammetry and Remote Sensing, in press.

Hirsh, H., & Noordewier, M. (1994). Using Background Knowledge to Improve Inductive Learning of DNA Sequences. *Proceedings of the 10th IEEE Conference on Artificial Intelligence for Applications,* (pp. 351-357).

Hjaltason, G. R., & Samet, H. (1999). Distance Browsing in Spatial Databases. *ACM Transactions on Database Systems, 24*(2), 265-318.

Ho, C.-T., Agrawal, R., Megiddo, N., & Srikant, R. (1997). Range queries in OLAP data cubes. *Proceedings of the 1997 ACM International Conference on Management of Data* (pp. 73-88).

Hoeffding, W. (1963). Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association, 58*(301), 13-30.

Hoffmann, C. M., & O'Donnell, M. J. (1982). Pattern matching in trees. *J. ACM,* 29(1), 68-95.

Höhle, U. (1995). Commutative, residuated l-monoids. *Non-Classical Logics and Their Applications to Fuzzy Subsets*, 53-106. Kluwer Academic Publishers.

Hohpe, G., & Woolf, B. (2003). *Enterprise Integration Patterns: Designing, Building, and Deploying Messaging Solutions.* Addison-Wesley Professional.

Höpfner, H. (2005). Towards update relevance checks in a context aware mobile information system. In A. B. Cremers, R. Manthey, P. Martini, & V. Steinhage (Eds.), *Proceedings of the 35rd annual conference of the German computer society*, number P-68 in Lecture Notes in Informatics (LNI) - Proceedings, (pp. 553–557), Bonn, Germany. Gesellschaft für Informatik, Köllen Druck+Verlag GmbH.

Höpfner, H. (2006). Update Relevance under the Multiset Semantics of RDBMS. In T. Kirste, B. König-Ries, K. Pousttchi, K. Turowski (Eds.), *Proceedings of the 1st conference on mobility and mobile information systems,* number P-76 in Lecture Notes in Informatics (LNI) -

Proceedings, (pp. 33–44), Bonn, Germany. Gesellschaft für Informatik, Köllen Druck+Verlag GmbH.

Höpfner, H., & Bunse, C. (2007). Resource Substitution for the Realization of Mobile Information Systems. *In proceedings of the 2nd international conference on software and data technologies, Volume: software engineering*, July 22-25, 2007, Barcelona Spain, pages 283-289, INSTICC Press

Höpfner, H., Türker, C., & König-Ries, B. (2005). *Mobile Datenbanken und Informationssysteme — Konzepte und Techniken.* dpunkt.verlag, Heidelberg, Germany. in German.

Horridge, M. (2004). *A practical guide to building OWL ontologies with the protégé-OWL plugin* (Edition 1.0). Manchester: The University of Manchester.

Horrocks, I., Patel-Schneider, P. F., Bechhofer, S., & Tsarkov, D. (2005). OWL Rules: A proposal and prototype implementation. *Journal of Web Semantics: Science, Services and Agents on the World Wide Web, 3*, 23-40.

Horrocks, I., Patel-Schneider, P. F., Boley, H., Tabet, S., Grosof, B., & Dean, M. (2004). *SWRL: A Semantic Web Rule Language Combining OWL and RuleML – W3C Member Submission 21 May 2004.* W3C (http://www.w3.org/Submission/SWRL/).

Hotho, A. et al. (2003). Ontologies Improve Text Document Clustering. *In Proceedings of the 3rd IEEE Conference on Data Mining* (pp. 541-544), Melbourne, FL, USA.

Hottier, P. (1996). *Précis de statistiques*. ENSG, IGN, France.

Hottier, P. (1996). Qualité géométrique de la planimétrie. Contrôle ponctuel et contrôle linéaire. Dossier: La notion de précision dans le GIS. *Journal Géomètre 6*, juin, Ordre des Géomètres-Experts Français, (pp. 34-42).

Hottier, P. (l996). *La méthode du contrôle linéaire en planimétrie - Propositions pour une typologie des détails anormaux.* Rapport de recherche, ENSG, IGN, France.

Houle, J. L., Cadigan, W., Henry, S., Pinnamaneni, A., & Lundahl, S. (2004. March 10). *Database Mining in the Human Genome Initiative. Whitepaper*, Bio-databases.com, Amita Corporation. Available: http://www.biodatabases.com/ whitepaper.html

Houle, M. E. & Sakuma, J. (2005). Fast approximate similarity search in extremely high-dimensional data sets. *Proceedings of IEEE ICDE*, 619-630, Tokyo, Japan.

Hsu, A., & Imielinski, T. (1985). *Integrity checking for multiple updates.* Proceedings of the 1985 ACM SIGMOD International Conference on the Management of Data, 152–168.

Hu, H., Lee, D.L., & Xu, J. (2006). *Fast nearest neighbor search on road networks.* Proceedings of the EDBT 2006, 186–203.

Huang Z., Harmelen F., & Teije A. (2006). *Reasoning with Inconsistent Ontologies: Framework, Prototype and Experiment. Semantic Web Technologies: Trends and Research in Ontology-based Systems*, (pp. 71-93).

Huang, C.-J., Liu, M.-C., Chu, S.-S., & Cheng, C.-L. (2007). An intelligent learning diagnosis system for Web-based thematic learning platform. *Computers & Education*, *48*(4), 658-679

Huang, J. (1991). Real time transaction processing: Design, implementation and performance evaluation. *PhD thesis, University of Massachusetts*.

Huang, X., Jensen, C.S., & Saltenis, S. (2005). *The islands approach to nearest neighbor querying in spatial networks.* Proceedings of the SSTD 2005, 73–90.

Huang, X., Jensen, C.S., & Saltenis, S. (2006). *Multiple k nearest neighbor query processing in spatial network databases.* Proceedings of the ADBIS 2006, 266–281.

Huang, Y. W., Jing, N., & Rundensteiner, E. A. (1997). Spatial Joins Using R-trees: Breadth-First Traversal with Global Optimizations. *In Proceedings of 23rd International Conference on Very Large Data Bases (VLDB 1997)*, Athens, Greece, 25-29 August, 1997, (pp. 396-405), Morgan Kaufmann, San Francisco.

Huang, Z. (1998). Extensions to the k-Means Algorithm for Clustering Large Data Sets with Categorical Values. *Data Mining and Knowledge Discovery, 2*(3), 283-304.

Hull, R., & Zhou, G. (1996). *A framework for supporting data integration using the materialized and virtual approaches.* Proceedings of the ACM SIGMOD Conference on Management of Data, Montreal, Canada, 481–492.

Hümmer, W., Bauer, A., & Harde, G. (2003). *XCube: XML for data warehouses.* Proceedings of the 6th International

Workshop on Data Warehousing and OLAP (DOLAP 03), New Orleans, Louisiana, 33–40.

Hunter, A. (1998). Paraconsistent logics. In D. Gabbay & Ph. Smets (Eds.), *Handbook of defeasible reasoning and uncertain information* (pp. 13-44). Dordrecht: Kluwer.

Hunter, A. (2003). Evaluating the significance of inconsistencies. *Proceedings of the International Joint Conference on AI (IJCAI'03)* (pp. 468-473). San Francisco: Morgan Kaufmann.

Hunter, L. (2004). Life and Its Molecules: A Brief Introduction. *AI Magazine, 25*(1), 9-22.

Hurtado, C., & Gutierrez, C. (2007). Handling structural heterogeneity in OLAP. In R. Wrembel, & C. Koncilia (Eds.), *Data warehouses and OLAP: Concepts, architectures and solutions* (pp. 27–57). Hershey, PA: Idea Group Publishing.

Hurtado, C.A., Mendelzon, A.O., & Vaisman, A.A. (1999). *Maintaining data cubes under dimension updates.* Proceedings of the 15th International Conference on Data Engineering (ICDE 99), Sydney, Australia, 346–355.

Hurtado, C.A., Mendelzon, A.O., & Vaisman, A.A. (1999). *Updating OLAP dimensions*. Proceedings of the DOLAP'99, Kansas City, Missouri, 60–66.

Hüsemann, B., Lechtenbörger, J., & Vossen, G. (2000). *Conceptual data warehouse design*. Proceedings of the 2nd International Workshop on Design and Management of Data Warehouses, 6.

Hyland, R., Clifton, C., & Holland, R. (1999). GeoNODE: Visualizing news in geospatial environments. *In Proceedings of the Federal Data Mining Symposium and Exhibition '99, AFCEA,* Washington D.C.

IBM (2004). *IBM DB2 Everyplace Application and Development Guide Version 8.2.* IBM Corporation.

IBM (2007). International Business Machines Corporation. www.ibm.com

Ibrahim, H. (1998). *Semantic integrity constraints enforcement for a distributed database* [doctoral thesis]. UK: University of Cardiff.

Ibrahim, H. (2002). *A strategy for semantic integrity checking in distributed databases.* Proceedings of the Ninth International Conference on Parallel and Distributed Systems, IEEE Computer Society, China.

Ibrahim, H. (2002). A strategy for semantic integrity checking in distributed databases. *Proceedings of the Ninth International Conference on Parallel and Distributed Systems.* Republic of China, IEEE Computer Society.

Ibrahim, H. (2006). Checking integrity constraints – How it differs in centralized, distributed and parallel databases. *Proceedings of the Second International Workshop on Logical Aspects and Applications of Integrity Constraints,* Krakow, Poland, (pp. 563-568).

Ibrahim, H. (2007). *AbSIS—An agent-based semantic integrity subsystem for managing mobile databases' constraints.* Proceedings of the 2007 World Congress in Computer Science, Computer Engineering, and Applied Computing—the 2007 International Conference on Information and Knowledge Engineering (IKE'07), Las Vegas, Nevada.

Ibrahim, H., Gray, W. A., & Fiddian, N. J. (2001). Optimizing fragment constraints – A performance evaluation. *International Journal of Intelligent Systems – Verification and Validation Issues in Databases, Knowledge-Based Systems, and Ontologies, 16*(3), 285-306. John Wiley & Sons Inc.

Ibrahim, H., Gray, W.A., & Fiddian, N.J. (1996). *The development of a semantic integrity constraint subsystem for a distributed database (SICSDD).* Proceedings of the 14th British National Conference on Databases, UK, 74–91.

Ibrahim, H., Gray, W.A., & Fiddian, N.J. (1998). *SICSDD—A semantic integrity constraint subsystem for a distributed database.* Proceedings of the 1998 International Conference on Parallel and Distributed Processing Techniques and Applications, 1575–1582.

ICDM. (2004). *Foundation of Data Mining Workshop call for papers.* Retrieved from http://biomig.csie.cgu.edu.tw/meeting/2004.07.25.htm

Iksal, S., & Garlatti, S. (2002). Revisiting and Versioning in Virtual Special Reports. *Lecture Notes in Computer Science, 2266,* 264-279.

Imieli´nski T. and Lipski W.(1984). Incomplete information in relational databases. J. ACM, 31(4):761–791.

Imielinski, T., & Mannila, H (1996). A database perspective on knowledge discovery. *Communications of the ACM, 39*(11), 58–64.

Imielinski, T., Viswanathan, S., & Badrinath, B.R. (1997). Data on Air: Organization and Access. *IEEE Transactions on Knowledge and Data Engineering,* IEEE, 9(3), 352-372.

In D. C. Andrews (Ed.), *International dimensions of technical communication* (pp. 69-86).Arlington, VA: Society for Technical Communication.

Inmon, W. (1992). *Building the data warehouse*. Wellesley, Massachusetts: QED Technical Publishing Group.

Inmon, W. (2002). *Building the data warehouse*. John Wiley & Sons Publishers

Inmon, W. H. (1996). The DW and data mining. *Communications of ACM, 39*(11), 49-50.

Inmon, W.H. (2005). *Building the Data Warehouse* (5th ed.). Indianapolis, IN: Wiley.

Institute of Electrical and Electronics Engineers (IEEE). (1990). *IEEE standard dictionary: A compilation of IEEE standard computer glossaries.* New York: Author.

Intanagonwiwat, C., Govindan, R., & Estrin, D. (2000). Directed Diffusion: A Scalable and Robust Communication Paradigm for Sensor Networks. *Proceedings of the Sixth Annual International Conference on Mobile Computing and Networking* (MobiCOM '00), August 2000, Boston, MA.

International Organization for Standardization (ISO). (2003). *Database language SQL, ISO/IEC 9075.*

Internet usage in Asia. (2008). *Internet World Stats.* Retrieved April 3, 2008, from http://www.internetworldstats.com/stats3.htm

Ioannidis, Y. E., & Poosala, V. (1995). Balancing histogram optimality and practicality for query result size estimation. *Proceedings of the 1995 ACM International Conference on Management of Data* (pp. 233-244).

Ioannidis, Y. E., & Poosala, V. (1999). Histogram-based approximation of set-valued query answers. *Proceedings of the 25th International Conference on Very Large Data Bases* (pp. 174-185).

Ioannidis, Y. E., & Poosala, V. (1999). Histogram-based Approximation of Set-Valued Query Answers. *Proc. of the 25th Int. Conf. on Very Large Data Bases,* (pp. 174-185).

Ip, M. Y. L., Saxton, L. V., & Raghavan, V. V. (1983). On the selection of an optimal set of indexes. *IEEE Transactions on Software Engineering, 9*(2), 135-143.

Irún, L., Muñoz, F. D., & Bernabéu, J. M. (2003). An Improved Optimistic and Fault-Tolerant Replication Protocol. *Lecture Notes in Computer Science, 2822,* 188-200.

Ishikawa, Y., Kitagawa, H. and Ohbo, N. (1993). Evaluation of signature files as set access facilities in OODBs. In *Proc. of ACM SIGMOD Int. Conf. on Management of Data,* Washington D.C., 247- 256.

ISO Standard (2003). Information technology – Database languages – SQL: 2003 International organization for standardization.

ISO/IEC 9075-14:2003. *Part 14: XML-Related Specifications (SQL/XML).* International Organization for Standardization.

Ives, Z.G., Florescu, D., Friedman, M., Levy, A.Y., & Weld, D.S. (1999). *An adaptive query execution system for data integration.* Proceedings of the ACM SIGMOD International Conference on Management of Data, 299–310.

Iyengar, V. S. (2002). Transforming data to satisfy privacy constraints. *Proceedings of 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'02)* (pp. 279-288).

J2EE Connector Architecture (2004). J2EE Connector Architecture. java.sun.com/j2ee/connector/

Jacox, E. H., & Samet, H. (2007). Spatial joins techniques. *ACM Transactions on Database Systems, 32*(1), 7 1-44.

Jagadish, H. V., Koudas, N., Muthukrishnan, S., Poosala, V., Sevcik, K., & Suel, T. (1998). Optimal histograms with quality guarantees. *Proceedings of the 24th International Conference on Very Large Data Bases* (pp. 275-286).

Jain, A. K., Murty, M. N., & Flynn, P. J. (1999). Data clustering: a review. *ACM Computing Surveys, 31*(3), 264-323.

Jajodia, S., & Mutchler, D. (1990). Dynamic Voting Algorithms for Maintaining the Consistency of a Replicated Database. *ACM Transactions on Database Systems, 15*(2), 230-280.

Janke, J.-H., & Wadsack, J. P. (1999). Varlet: Human-centered tool for database reengineering. *WCRE*.

Janowicz, K. (2005). Extending semantic similarity measurement by thematic roles. *Proceedings of the GeoS'05: First International Conference on GeoSpatial Semantic* (pp. 137-152).

Jarazabek, S., & Hitz, M. (1998). Business-oriented component-based software development and evolution. *DEXXA Workshop*.

Jaro, M. A. (1989). Advances in record linking methodology as applied to the 1985 census of Tampa Florida. *Journal of the American Statistical Society, 64*, 1183-1210.

JCA (2007). J2EE Connector Architecture. java.sun.com/j2ee/connector/index.jsp

Jensen, C. S., & Saltenis, S. (2002). Towards increasingly update efficient moving-object indexing. *IEEE Data Engineering Bulletin, 25*(2), 35-40.

Jensen, C. S., Kolář, J., Pedersen, T. B., & Timko, I. (2003). Nearest neighbor queries in road networks. *Proceedings of 11th ACM International Symposium on Advances in Geographic Information Systems* (pp. 1-8).

Jensen, C.S., Klygis, A., Pedersen, T., & Timko, I. (2004). Multidimensional data modeling for location-based services. *VLDB Journal, 13*(1), 1–21.

Jensen, C.S., Kolárvr, J., Pedersen, T.B., & Timko, I. (2003). Nearest neighbor queries in road networks. Proceedings of the ACM-GIS 2003, 1–8.

Jensen, O. G., & Bohlen, M. H. (2004). Lossless Conditional Schema Evolution. Conceptual Modeling - ER 2004. *23rd International Conference on Conceptual Modeling*, Shanghai, China, Springer.

Jepson, B. (2002). *Beware SQL injection in Web applications.* Retrieved September 12, 2006, from http://www.oreillynet.com/mac/blog/2002/06/beware_sql_injection_in_web_ap.html

Jeusfeld, M. A., & Johnen, U. A. (1994). *An executable meta model for reengineering of database schemas.* Proceedings of Conference on the Entity-Relationship Approach, Manchester, England.

Jiang, D., Pei, J., & Zhang, A. (2003). Toward interactive exploration of gene expression patterns. *ACM-SIGKDD Explorations, 5*(2), 79-90.

Jiang, H., Lu, H., Wang, W., & Ooi, B.C. (2003). *XR-Tree: Indexing XML data for efficient structural joins.* Proceedings of the 19th International Conference on Data Engineering (ICDE 03), Bangalore, India, 253–263.

Jiang, J.J., Klein, G., & Pick, R.A. (2003). The impact of IS Department organizational environments upon project team performances. *Information and Management, 40*(3), 213–220.

Jin, H.; Wong, M., & Leung, K. (2005). Scalable Model-Based Clustering for Large Databases Based on Data Summarization. *IEEE Transactions on Pattern Anal. Mach. Intell, 27*, 1710-1719. IEEE Computer Society.

Jin, W., Tung, A. K. H., & Han, J. (2001). Finding top *n* local outliers in large database. *Proceedings of the 7th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD'01)* (pp. 293-298).

Johnson, D. S. (1990). A catalog of complexity classes. In J. van Leewen (Ed.), *Handbook of theoretical computer science* (Vol. 1, pp. 67-161). North-Holland.

Johnstone, D., Huff, S., & Hope, B. (2006). IT Projects: Conflict, Governance, and Systems Thinking. *The 39th Hawaii International Conference on System Sciences.*

Joris, M. (1992). Phenix: Methods and tools for database reverse engineering. *Proceedings 5th International Conference on Software Engineering and Applications.*

Joshi, A. (2000). On Proxy Agents, Mobility and Web Access. *ACM/Baltzer Journal on Mobile Networks and Applications*, Springer Science, 5(4), 233-241.

Joshi, A. K., & Schabes, Y. (1997). Tree-Adjoining Grammars. In G. Rozenberg & A. Salomaa (Eds.), *Handbook of Formal Languages, 3*, 69–124. Springer, Berlin, New York, 1997.

Jovanovic, J., Gasevic, D., & Devedzic, V. (2004). A GUI for Jess. *Expert Systems With Applications, 26*(4), 625-637

JPivot. (n.d.). Retrieved July 10, 2006, from http//jpivot.sourceforge.net

Kabra, N., & DeWitt, D.J. (1998). *Efficient mid-query re-optimization of sub-optimal query execution plans.* Proceedings of the ACM SIGMOD International Conference on Management of Data, 106–117.

Kajan, E., & Stoimenov, L. (2005). Towards ontology-driven architectural framework for B2B. *Communications of the ACM, 48*(12), 60-66.

Kakas, A., Kowalski, R., & Toni, F. (1992). Abductive logic programming. *Journal of Log. Comput., 2*(6), 719-770.

Käki, M. (2005). Findex: search result categories help users when document ranking fails. In *CHI -05: Proc. of the SIGCHI conference on Human factors in computing systems* (pp. 131–140), Portland, OR, USA.

Kalfoglou, Y., & Schorlemmer, M. (2003). Ontology mapping: The state of the art. *The Knowledge Engineering Review, 18*(1), 1–31.

Kalogeraki, V., Gunopulos, D., & Zeinalipour-Yazti, D. (2002). A local search mechanism for peer-to-peer networks. *Proceedings of the 11th ACM International Conference on Information and Knowledge Management* (pp. 300-307).

Kamath, G. R. (2000, May). The India paradox. *Intercom,* pp. 10-11.

Kane, B., Su, H., & Rundernsteiner, A. E. (2002). Consistently updating XML documents using incremental constraint check queries. In Fourth ACM CIKM International *Workshop on Web Information and Data Management (WIDM'02)* (pp. 1-8). Virginia, USA: ACM.

Kao, B., & Garcia – Monila, H. (1995). An overview of real-time database systems. *Advances in Real-Time Systems*, 463-486.

Kaplan, R. B. (2001). Foreword: What in the world is contrastive rhetoric? In C. G. Panetta (Ed.), *Contrastive rhetoric revisited and redefined* (pp. vii-xx). Mahwah, NJ: Lawrence Erlbaum Associates.

Kappel, G., Kapsammer, E., & Retschitzegger, W. (2001a). Architectural Issues for Integrating XML and Relational Database Systems – The X-Ray Approach. *Proceedings of the Workshop on XML Technologies and Software Engineering,* Toronto.

Kappel, G., Kapsammer, E., & Retschitzegger, W. (2001b). XML and Relational Database Systems – A Comparison of Concepts'. *Proceedings of the 2nd International Conference on Internet Computing (IC).* CSREA Press, Las Vegas, USA.

Karahasanovic, A. (2001). Identifying Impacts of Database Schema Changes on Application, in *Proc. of the 8th Doctoral Consortium at the CAiSE*01,* (pp. 93-104).

Karahasanovic, A. (2001). SEMT-A Tool for finding Impacts of Schema Changes. NWPER'2000. *Ninth Nordic Workshop on Programming Environment Research,* Lillehammer, Norway.

Karayiannis, N. B. (1995). *Generalized fuzzy k-means algorithms and their application in image compression.* Paper presented at the Society of Photo-Optical Instrumentation Engineers (SPIE) Conference.

Karlapalem, K., Navathe, S. B., & Ammar, M. H. (1996). Optimal Redesign Policies to Support Dynamic Processing of Applications on a Distributed Relational Database System. *Information Systems, 21*(4), 353-367.

Karvounarakis, G., Alexaki, S., Christophides, V., Plexousakis, D., & Scholl, M. (2002). RQL: A declarative query language for RDF. *Proceedings of the 11th International World Wide Web Conference* (pp. 592-603).

Karypis, G., Han, E.-H., & Kumar, V. (1999). Chameleon: hierarchical clustering using dynamic modeling. *Computer, 32*(8), 68-75.

Katayama, N., & Satoh, S. (1997). The SR-tree: An index structure for high-dimensional nearest neighbor queries. *Proceedings of 1997 ACM SIGMOD* (pp. 369-380).

Kaufman, L., & Rousseeuw, P. J. (1990). Finding groups in data. an introduction to cluster analysis: Wiley Series in Probability and Mathematical Statistics. *Applied Probability and Statistics*, New York: Wiley, 1990.

Kaushik, R., Bohannon, P., Naughton, J., & Korth, H. (2002, June). Covering indexes for branching path queries. In *ACM SIGMOD.*

Kaushik, R., Shenoy, D., Bohannon, P., & Gudes, E. (2002). *Exploiting local similarity to efficiently index paths in graph-structured data.* Proceedings of the 18th International Conference on Data Engineering (ICDE 02), San Jose, California, 129–140.

Kavouras, M., Kokla, M., & Tomai, M. (2003). Comparing categories among geographic ontologies. *Computers & Geosciences, Special Issue, Geospatial Research in Europe: AGILE 2003, 31*(2), 145–154.

Kayan, Ersan, & Ulusoy, O. (1999). An Evaluation of Real Time Transaction Management Issues in Mobile Database Systems. Computer Journal, 42(6).

Keller, A. M., & Basu, J. (1996). A predicate-based caching scheme for client-server database architectures. *The VLDB Journal, 5*(1), 35–47.

Kemme, B., & Alonso, G. (2000). A New Approach to Developing and Implementing Eager Database Replication Protocols. *ACM Transactions on Database Systems, 25*(3), 333-379.

Kent, W. (1981). Consequences of assuming a universal relation. *ACM Transactions on Database Systems, 6*(4), 539-556.

Kenthapadi, K., Mishra, N., & Nissim, K. (2005). Simulatable auditing. *Proceedings of ACM Symposium on Principles of Database Systems (PODS'05)* (pp. 118-127).

Keogh, E. J., & Pazzani, M. J. (1999). An indexing scheme for fast similarity search in large time series databases. *Proceedings of SSDBM*, 56-67, Clevelant, Ohio, USA.

Keogh, E., Chakrabarti, K., Pazzani, M., & Mehrotra, S. (2000). Dimensionality reduction for fast similarity search in large time series databases. *Knowledge and Information Systems, 3*(3), 263-286.

Kephart, J. O. (2005). Research challeneges of autonomic computing. *In Proceedings of the 27th International Conference on Software engineering,* 15-22.

Kervyn, F. (2001). Modeling topography with SAR interferometry: Illustration of a favorable and less favorable environment. *Computer & Geosciences, 27,* 1039-1050.

Khan, K., & Khang Y. (2004). *Managing Corporate Information Systems Evolution and Maintenance*. Hershey, PA: Idea Group Publishing.

Kiely, D. (2002). *Guarding against SQL injection attacks.* Retrieved September 5, 2006, from http://www.itworld.com/nl/windows_sec/03252002/

Kilpelainen, P. and Mannila, H (1995). Ordered and unordered tree inclusion. *SIAM Journal of Computing*, 24:340-356.

Kim, J.C., & Park, S. (1996). Recovery method using shadow paging in MMDBS. *Korea Information Science Society Journal, 14*(6), 428–431.

Kim, J.L., & Park, T. (1993). An efficient algorithm for checkpointing recovery in distributed systems. *IEEE Transactions on Parallel and Distributed Systems, 4*(8), 955–960.

Kim, J.L., & Park, T. (1994). *An efficient recovery scheme for locking-based distributed database systems.* Proceedings of the 13th Symposium on Reliable Distributed Systems, 116–123.

Kim, W. Y., Choi, B.-J., Hong, E. K., Kim, S.-K., & Lee, D. (2003). A taxonomy of dirty data. *Data Mining Knowledge Discovery, 7*(1), 81-99.

Kim, W., & Lochovsky, F. H. (Eds.) (1989). *Object-Oriented Concepts, Databases, and Applications*. ACM Press and Addison-Wesley.

Kim, Y. K., & March, S. T. (1995). Comparing data modeling formalisms. Communications of the ACM, 38(6), 103-115.

Kimball, R. (1996). Slowly changing dimensions. *DBMS* and *Internet Systems.*

Kimball, R. (1996). *The data warehouse toolkit.* John Wiley & Sons.

Kimball, R., & Ross, M. (2002). *The data warehouse toolkit* (2nd ed.). New York: John Wiley and Sons, Inc.

Kimball, R., Reeves, L., Ross, M., & Thornthwaite, W. (1998). *The data warehouse lifecycle toolkit: Expert methods for designing, developing, and deploying data warehouses.* John Wiley & Sons Publishers.

Kirchberg, M., Schewe, K-D., Tretiakov, A., & Wang, B. (2007). A multi-level architecture for distributed object bases. *Data and Knowledge Engineering, 60*(1), 150-184.

Klein, D., Kamvar, S. D., & Manning, C. D. (2002). *From Instance-level Constraints to Space-Level Constraints: Making the Most of Prior Knowledge in Data Clustering.* Paper presented at ICML '02: the nineteenth International Conference on Machine Learning, Morgan Kaufmann Publishers Inc., 2002, 307-314.

Klein, M. (2001). *Combining and relating ontologies: An analysis of problems and solutions.* Proceedings of the IJCAI-2001 Workshop on Ontologies and Information Sharing, Seattle, Washington.

Kleinberg, J. M., Papadimitriou, C. H., & Raghavan, P. (2000). Auditing Boolean attributes. *Proceedings of ACM Symposium on Principles of Database Systems (PODS'00)* (pp. 86-91).

Kleinberg, J. M., Papadimitriou, C. H., & Raghavan, P. (2001). On the value of private information. *Proceedings of the 8th Conference on Theoretical Aspects of Rationality and Knowledge.*

Klettke, M., & Meyer, H. (2000). XML and Object-Relational Database Systems – Enhancing Structural Mappings Based on Statistics. In *Lecture Notes in Computer Science, 1997*, 151–170. Springer-Verlag.

Klien, E., Einspanier, U., Lutz, M., & Hübner, S. (2004). *An architecture for ontology-based discovery and retrieval of geographic information.* Proceedings of the 7th Agile Conference on Geographic Information Science, Heraklion, Crete.

Knight, K. M. (2003). Two information measures for inconsistent sets. *Journal of Logic, Language and Information, 12*(2), 227-248.

Knorr, E. M., & Ng, R. T. (1998). Algorithms for mining distance-based outliers in large dataset. *Proceedings of the 24th International Conference on Very Large Data Bases (VLDB'98)* (pp. 392-403).

Knorr, E. M., & Ng, R. T. (1999). Finding intentional knowledge of distance-based outliers. *Proceedings of the 25th International Conference on Very Large Data Bases (VLDB'99)* (pp. 211-222).

Knuth, D. E. (1969). *The art of computer programming, 1.* Reading: Addison-Wesley.

Kocberber, S., and Can, F. (1999). Compressed Multi-Framed Signature Files: An Index Structure for Fast Information Retrieval, in: *Proc. ACM SAC'99,* Texas, USA, 1999, pp. 221-226.

Kohonen, T. (1988). *Self-organized formation of topologically correct feature maps.* Neurocomputing: foundations of research, MIT Press, 1988, (pp. 509-521).

Kokkinidis, G., & Christophides, V. (2004). Semantic query routing and processing in P2P database systems: The ICS-FORTH SQPeer middleware. *Proceedings of the 2004 International Workshop on Peer-to-Peer Computing and Databases, in conjunction with the 9th International Conference on Extending Database Technology* (pp. 486-495).

Kolahdouzan, M., & Shahabi, C. (2004). Voronoi-based nearest neighbor search for spatial network databases. *Proceedings of 30th International Conference on Very Large Databases* (pp. 840-851).

Kolahdouzan, M.R., & Shahabi, C. (2004). *Voronoi-based K nearest neighbor search for spatial network databases.* Proceedings of the VLDB 2004, 840–851.

Kolahdouzan, M.R., & Shahabi, C. (2005). Alternative solutions for continuous K nearest neighbor queries in spatial network databases. *GeoInformatica, 9*(4), 321–341.

Kolaitis, P. G., & Thakur, M. N. (1994). Logical definability of NP optimization problems. *Information and Computation, 115*, 321-353.

Kolatch, E. (2001). *Clustering Algorithms for Spatial Databases: A Survey.*Unpublished manuscript. Department of Computer Science, University of Maryland, College Park. URL: "http://citeseer.ist.psu.edu/kolatch-01clustering.html"

Kollios, G., Gunopoulos, D., & Tsotras, V. J. (1999). On indexing mobile objects. *Proc. of the 18th ACM Symposium on Principles of Database Systems (PODS'1999),* 261-272. Philadelphia, PA.

Koloniari, G., & Pitoura, E. (2004). Content-based routing of path queries in peer-to-peer systems. *Proceedings of the 9th International Conference on Extending Database Technology* (pp. 29-47).

Koncilia, C. (2003). A bi-temporal data warehouse model [CAiSE'03 short paper]. Proceedings of the CEUR Workshop, 74, 77–80.

Kontaki, M., & Papadopoulos, A. N. (2004). Efficient similarity search in streaming time sequences. *Proceedings of SSDBM,* 63-72, Santorini, Greece.

Kontaki, M., Papadopoulos, A. N., & Manolopoulos, Y. (2007). Adaptive similarity search in streaming time series with sliding window. *DKE* to appear.

Kooi, R. P. (1980). *The optimization of queries in relational databases.* Unpublished doctoral dissertation, CWR University.

Korn, F., & Muthukrishnan, S. (2000). Influence Sets Based on Reverse Nearest Neighbor Queries. *In Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data (SIGMOD 2000),* Dallas, Texas, USA, 16-18 May, 2000, (pp. 201-212), ACM Press.

Kost, S. (2004). An introduction to SQL injection attacks for Oracle developers. *Help Net Security.* Retrieved August 1, 2006, from http://www.net-security.org/article.php?id=633

Kotidis, Y., & Roussopoulos, N. (1999). Dynamat: A dynamic view management system for data warehouses. *ACM SIGMOD International Conference on Management of Data (SIGMOD 1999)* (pp. 371-382).

Koudas, N., & Srivastava, D. (2005). Data stream query processing: A tutorial. *Proceedings of the 29th International Conference on Very Large Data Bases* (p. 1149).

Kourouthanassis, P., & Roussos, G. (2003). Developing Consumer-Friendly Pervasive Retail Systems. *IEEE Pervasive Computing, 2*(2), 32-39.

Kraak, M. J. (2004). The role of the map in a Web-GIS environment. *Journal of Geographical Systems, 6*(2), 83-93.

Krallinger, M., & Valencia, A. (2005). Text-mining and information-retrieval services for molecular biology. *Genome Biology, 6*(224).

Kratika, J., Ljubic, I., & Tosic, D. (2003). A genetic algorithm for the index selection problem. In *Lecture notes in computer science: Vol. 2611. Applications of Evolutionary Computing (EvoWorkshops 2003)* (pp. 281-291). Berlin, Germany: Springer.

Kriegel, H., Kroger, P., Pryakhin, A., & Schubert, M. (2005). *Effective and Efficient Distributed Model-Based Clustering.* Paper presented at ICDM '05: the Fifth IEEE International Conference on Data Mining, IEEE Computer Society, 2005, 258-265.

Krishnamachari, B., Estrin, D., & Walker, S. (2002). The Impact of Data Aggregation in Wireless Sensor Networks. *International Workshop on Distributed Event-Based Systems*, Vienna, Austria, July.

Kryszkiewicz, M. (2001). Comparative study of alternativetypes of knowledge reduction in inconsistent systems. *International Journal of Intelligent Systems, 16*(1), 105-120.

Ku, W.-S., Zimmermann, R., Wang, H., & Wan, C.-N. (2005). *Adaptive nearest neighbor queries in travel time networks.* Proceedings of the ACM-GIS 2005, 210–219.

Kubach, U., & Rothermel, K. (2001). Exploiting location information for infostation-based hoarding. In C. Rose, (Eds.), *Proceedings of the 7th Annual International Conference on Mobile Computing and Networking*, July 16-21, 2001, Rome, Italy Rome, Italy, (pp. 15–27), New York, NY, USA. ACM Press.

Kuenning, G. H., & Popek, G. J. (1997). Automated Hoarding for Mobile Computers. In W. M. Waite (Eds.), *Proceedings of the 16th ACM Symposium on Operating Systems Principles*, (pp. 264–275), New York, NY, USA. ACM Press.

Kuhn, T. S. (1970). *The structure of scientific revolutions*, 2nd Edition. University of Chicago Press.

Kuo, R. J., Wu, P. C., & Wang, C. P. (2000). Fuzzy neural networks for learning fuzzy if-then rules. *Applied Artificial Intelligence, 14*(6).

Kupfer, A., Eckstein, S. et al. (2006). *Handling Changes of of Database Schemas and Correspondin Ontologies Advances in Conceptual Modeling - Theory and Practice, ER 2006 Workshops BP-UML*, CoMoGIS, COSS, ECDM, OIS, QoIS, SemWAT, November 6-9, Tucson, AZ, USA, Springer.

Kuznetsov S. O., & Obiedkov S.A. (2002). Comparing Performance of Algorithms for Generating Concept Lattices. Journal of Experimentation and Theoretical Artificial Intelligence.

Kwiatkowski, J., & Puchalski, I. (1998). Pre-processing COBOL programs for reverse engineering in a software maintenance tool. *COTSR*.

Kyu-Young, W. (1987). Index selection in rational databases. In S. P. Ghosh, Y. Kambayashi, & K. Tanaka (Eds.), *Foundation of data organization* (pp. 497-500). New York: Plenum Publishing.

Labio, W., Quass, D., & Adelberg, B. (1997). Physical database design for data warehouses. *13th International Conference on Data Engineering (ICDE 1997)* (pp. 277-288).

Lacroix, Z. (2002). Biological data integration: Wrapping data and tools. *IEEE Transactions on Information Technology in Biomedicine, 6*(2), 123-128.

Lai, Y. (2003). The interdependent arising is the fundamental theory of Buddhism. *Social Science Front, 2003*(5), 50-52.

Lakshmanan, L. V. S., Sadri, F., & Subramanian, I. N. (1993). On the logical foundations of schema integration and evolution in heterogeneous database systems. *Proceedings of Third International Conference of Deductive and Object-Oriented Databases* (*DOOD'03*), 81-100.

Lam, K. Y. (1994). Concurrency control in distributed real-time database systems. *PhD Thesis, City University of Hong Kong, Hong Kong.*

Lam, K. Y., Hung, S. L., & Son, S. H. (1997). On using real-time static locking protocols for distributed real-time databases. *Real-Time Systems, 13,* 141-166.

Lam, K. Y., Pang, C., Son, S. H., & Cao, J. (1999). Resolving executing committing conflicts in distributed real-time database systems. *Journals of Computer, 42*(8), 674-692.

Lam, K.Y., & Kuo, T.W. (Eds.). (2001). *Real-time database architecture and techniques.* Boston: Kluwer Academic Publishers.

Lam, K.Y., Kuo, T.W., Tsang, W.H., & Law, G.C.K. (2000). Concurrency control in mobile distributed real-time database systems. *Information Systems, 25*(4), 261–286.

Lam, K.Y., Pang, C., Son, S. H. & Cao, J. (1999). Resolving Executing -Committing Conflicts in Distributed Real - time Database Systems. Journals of Computer, 42(8), 674-692.

Lambert, D. (1993). Measures of disclosure risk and harm. *Journal of Official Statistics, 9*(2), 313-331.

Langley, P. (2000). The computational support of scientific discovery. *International Journal of Human-Computer Studies, 53,* 393-410.

Larkey, L. S. (1999). A patent search and classification system. In *Proc. of DL-99, 4th ACM Conference on Digital Libraries* (pp. 179–187), Berkeley, CA, USA.

Larman, C., & Basili, V. R. (2003). Iterative and incremental development: A brief history. *IEEE Computer, 36*(6), 47-56.

Larson, A. P. (1998). Federated databases systems for managing distributed, heterogeneous, and autonomous databases. *ACM Computing Surveys, 22,* 183-236.

Lassila, O. (1999). Resource Description Framework. *Proc in XML'99 Conference*, Philadelphia.

Lassila, O., Hendler, J. (2007). Embracing Web 3.0. IEEE Internet Computing, May/June 2007

Latour, B., & Wolgar, S. (1979). *Laboratory Life: The Social Construction of Scientific Facts.* Sage Pulishing, Los Angeles.

Laugero, G., & Globe, A. (2002). *Enterprise content services: a practical approach to connecting content management to business strategy.* Boston, MA: Addison-Wesley.

Laurini, R., & Milleret-Raffort, F. (1993). *Les bases de données en géomatique.* Editions

Laurini, R., et al. (1998). Spatial multi-database topological continuity and indexing: A step towards seamless GIS data interoperability. *International Journal of Geographic Information Science, 12*(4), 373-402.

Lautemann, S.-E. (1996). An Introduction to Schema Versioning in OODBMS. *Object-Oriented Databases 2 workshop*, Zurich, Switzerland, IEEE-CS Press.

Lavraĉ, N., & Flash, P. (2000). *An extended transformation approach to Inductive Logic Programming.* (Tech. Rep. No. 00-002). UK: University of Bristol.

Le, B. T., Dieng-Kuntz, R., & Gandon, F. (2004). On ontology matching problems for building a corporate Semantic Web in a multi-communities organization. *In ICEIS 2004 Software Agents and Internet Computing,* 236-243.

Lee, D. (1999). Computational geometry II. In M. Atallah (Ed.), *Algorithms and Theory of Computation Handbook* (pp. 20-1 to 20-31). CRC Press.

Lee, D.L, Kim, Y.M., and Patel, G. (1995). Efficient Signature File Methods for Text Retrieval, *IEEE Transaction on Knowledge and Data Engineering*, Vol. 7, NO. 3, June pp.423-435.

Lee, K. C. K., Leong, H. V., & Si, A. (1999). Semantic query caching in a mobile environment. *ACM SIGMOBILE Mobile Computing and Communications Review, 3*(2), 28–36.

Lee, W. and Lee, D.L. (1992). Signature File Methods for Indexing Object-Oriented Database Systems. *Proc. ICIC'92 - 2nd Int. Conf. on Data and Knowledge Engineering: Theory and Application*, Hong Kong, 616-622.

Lehman, M. M., & Stenning V. (1997). Laws of Software Evolution Revisited. *Lecture Notes in Computer Science, 1149,* 108-124. Springer-Verlag Publ.

Lemmens, L, Wytzisk, A., de By, R., Granell, C., Gould, M., & van Oosterom, P. (2006). Integrating semantic and

syntactic descriptions to chain geographic services. *IEEE Internet Computing, 10*(5), 42–52.

Lenzerini, M. (2002). Data integration: A theoretical perspective. *PODS* (pp. 233-246).

Leone, N., Pfeifer, G., Faber, W., Calimeri, F., Dell'Armi, T., Eiter, T., et al. (2002). The DLV system. *European Conference on Logics in Artificial Intelligence* (pp. 537-540).

Lerner, B. S. (2000). A model for compound type changes encountered in schema evolution. *ACM Transactions on Database Systems, 25*(1), 83-127.

Lerner, B. S., & Habermann, A. N. (1990). *Beyond schema evolution to database reorganisation. Conference on Object-Oriented Programming Systems, Languages, and Applications / European Conference on Object-Oriented Programming,* Ottawa, Canada.

Lessons from afar. (2003, May 8). *The Economist.* Retrieved September 12, 2005, from http://www.economist.com/business/globalexecutive/displaystory.cfm?story_id=1762562&tranMode=none

Lester, M., & Chrisman, N. (1991). Not all slivers are skinny: a comparison of two methods for detecting positional error in categorical maps. *GIS/LIS, 2,* 648-656.

Letz, C., Henn, E.T., & Vossen, G. (2002). *Consistency in data warehouse dimensions.* Proceedings of the International Symposium on Database Engineering & Applications (IDEAS 02), Edmonton, Canada, 224–232.

Leung, H., Chung, F., Chan, S., & Luk, R. (2005). XML document clustering using common XPath. *Proceedings of the IEEE International Workshop on Challenges in Web Information Retrieval and Integration* (pp. 91-96).

Leuschel, M., & de Schreye, D. (1998). Creating specialised integrity checks through partial evaluation of meta-interpreters. *JLP, 36*(2), 149-193.

Levenshtein, V. (1966). Binary codes capable of correcting deletions, insertions and reversals. *Soviet Physics: Doklady, 10*(10), 707-710.

Levinsohn, A. (2000). Geospatial interoperability: The holy grail of GIS. *GeoEurope.* Retrieved 2006 from http://www.geoplace.com/gw/2000/1000/1000data.asp

Levy, A. (2000). Logic-based techniques in data integration. In J. Minker (Ed.), *Logic-based artificial intelligence* (pp. 575-596). Dordrecht: Kluwer. Link, S. (2003). Consistency enforcement in databases. *Proceedings of the 2nd International Workshop on Semantics in Databases 2001, Lecture Notes in Computer Science, 2582* (pp. 139-159).

Lewis, D. D. (1995). The TREC-4 filtering track: description and analysis. In *Proc. of TREC-4, 4th Text Retrieval Conference*, (pp. 165–180), Gaithersburg, MD, USA.

Lewis, P. M., Bernstein, A., & Kifer, M. (2002). *Database transaction processing: An application oriented approach.* Boston: Addison Wesley.

Li, C., & Wang, X. S. (1996). A data model for supporting on-line analytical processing. *Proceedings of the Conference on Information and Knowledge Management* (pp. 81-88).

Li, F., Cheng, D., Hadjieleftheriou, M., Kollios, G., & Teng, S.-H. (2005). *On trip planning queries in spatial databases.* Proceedings of the SSTD 2005, 273–290.

Li, H., Lee, M.L., Hsu, W., & Chen, C. (2004). An evaluation of XML indexes for structural join. *SIGMOD Record, 33*(3), 28–33.

Li, H., Lui, C., & Orlowska, M. (1998). *A query systems for object-relational databases.* Proceedings of the 9th Australasian Database Conference (ADC'98), Pert, Australia,.39–50.

Li, M., Lee, W.-C., & Sivasubramaniam, A. (2003). Neighborhood signatures for searching P2P networks. *Proceedings of the 7th IEEE International Database Engineering and Applications Symposium* (pp. 149-159).

Li, Q., & Moon, B. (2001, September). Indexing and querying XML data for regular path expressions. In *Proc. VLDB*, 361-370.

Li, X. (1999). A Survey of Schema Evolution in Object-Oriented Databases. *TOOLS, 31,* 362-371.

Li, X. (1999). A survey of schema evolution in object-oriented databases. *Proceedings of TOOLS'31,* 362-371.

Li, X., & Lin, H. (2006). Indexing network-constrained trajectories for connectivity-based queries. *International Journal of Geographical Information Science, 20*(3), 303–328.

Li, Y., & Zhong, N. (2006). Mining Ontology for Automatically Acquiring Web User Information Needs. *IEEE*

Transactions on Knowledge and Data Engineering, 18(4), 554-568.

Lian, W., Cheung, D. W., Mamoulis, N., & Yiu, S.-M. (2004). An efficient and scalable algorithm for clustering XML documents by structure. *IEEE Transactions on Knowledge and Data Engineering, 16*(1), 82-96.

Lian, X., Chen, L., Yu, J. X., Wang, G., & Yu, G. (2007). Similarity match over high speed time-series streams. *Proceedings of IEEE ICDE,* 1086-1095. Istanbul, Turkey.

Lifschitz, S., & Sales, M. A. V. (2005). Autonomic index management. *In Proceedings of the Second International Conference on Autonomic Computing (ICAC'05),*304-305.

Lightstone, S., Schiefer, B., Zilio, D., & Kleewein, J. (2003). Autonomic computing for relational database: The ten-year vision. In *Proceedings of the IEEE Workshop Autonomic Computing Principles and Architectures (AUCOPA'03),* 419-424.

Liknes, G., Hugg, R., Sun, J., Cullen, W., & Reese, C. (2000, June). *Techniques for conversion of weather data into ESRI formats.* Proceedings of the ESRI International Users Conference, San Diego, CA.

Lim D., Cho, D. and Hong B. (2007). Indexing Moving Objects for Trajectory Retrieval on Location-Based Services. *IEICE Transactions on Information and Systems 2007 E90-D(9)*:1388-1397

Lim, S. J., & Ng, Y. K. (2001). An Automated Change-Detection Algorithm for HTML Documents Based on Semantic Hierarchies. In *The 17th International Conference on Data Engineering (ICDE 2001),* (pp. 303-312), Heidelberg, Germany.

Lin, C-Y. (2004). ROUGE: A package for automatic evaluation of summaries. *Proceedings of Workshop on Text Summarization Branches Out, Post-Conference Workshop of ACL*, Barcelona, Spain, 74-81.

Lin, D., Jensen, C. S., Ooi, B. C., & Saltenis, S. (2005). Efficient indexing of the historical, present and future positions of moving objects. *Proc. of Mobile Data Management (MDM'2005),* 59-66.

Lin, J. (1996a). Integration of weighted knowledge bases. *Artificial Intelligence, 83*(2), 363-378.

Lin, J. (1996b). A semantics for reasoning consistently in the presence of inconsistency. *AI, 86*(1), 75-95.

Lin, J., & Mendelzon, A. O. (1999). Knowledge base merging by majority. In R. Pareschi & B. Fronhoefer (Eds.), *Dynamic worlds.* Kluwer.

Lin, K., Jagadish, H. V., & Faloutsos, C. (1994). The TV-tree: An index structure for high dimensional data. *The VLDB Journal, 3*(4), 517-542.

Lin, K.-I., Jagadish, H. V., & Faloutsos, C. (1994). The TV-tree: An index structure for high-dimensional data. *VLDB Journal 3*(4), 517-542.

Lin, Y., Kemme, B., Patiño-Martínez, M., & Jiménez-Peris, R. (2005). Middleware-Based Data Replication providing Snapshot Isolation. *ACM SIGMOD Conference,* (pp. 419-430).

Lindell, Y., & Pinkas, B. (2000). Privacy preserving data mining. In *Lecture notes in computer science: Vol. 1880. Proceedings of Advances in Cryptology: Crypto'00* (pp. 20-24). Springer-Verlag.

Lisi, F. A. (2006). Practice of inductive reasoning on the semantic Web: A system for semantic Web mining. In J. J. Alferes, J. Bailey, W. May, & U. Schwertel (Eds.), *Lecture Notes in Computer Science, 4187,* 242-256. Berlin: Springer.

List, B., Schiefer, J., & Min Tjoa, A. (2000). Process-oriented requirement analysis supporting the data warehouse design process - A use case-driven approach. *Proceedings of the 11th International Conference on Database and Expert Systems Applications,* 593-603.

Litwin, P. (2002). *Guard against SQL injection attacks.* Retrieved July 23, 2006, from http://www.aspnetpro.com/opinion/2002/08/asp200208pl_o/asp200208pl_o.asp

Litwin, W., Mark, L., & Roussoupoulos, N. (1990). Interoperability of multiple autonomous databases. *ACM Computing Surveys, 22*(3), 267-293.

Liu K.-C. and Sunderraman R.(1990). Indefinite and maybe information in relational databases. ACM Trans. Database Syst., 15(1):1–39.

Liu K.-C. and Sunderraman R.(1991). A generalized relational model for indefinite and maybe information. IEEE Trans. Knowl. Data Eng., 3(1):65–77.

Liu, B., Lee, W.-C., & Lee, D.L., (2005). Distributed Caching of Multi-Dimensional Data in Mobile Environments. *Proc. of the 6th ACM Int. Conf. on Mobile Data Management,* (pp. 229-233).

Liu, C.-T., Chang, S.-K. et al. (1994). Database Schema Evolution using EVER Diagrams. *AVI '94, Proceedings of the Workshop on Advanced Visual Interfaces,* June 1-4, 1994., Bari, Italy, ACM.

Liu, F., Do, T.T., & Hua, K.A. (2006). *Dynamic range query in spatial network environments.* Proceedings of the DEXA 2006, 254–265.

Liu, P., Amman, P., & Jajodia, S. (2000). Rewriting histories: Recovering from malicious transactions. *Distributed and Parallel Databases, 8*(1), 7–40.

Liu, X., & Ferhatosmanoglu, H. (2003). Efficient k-NN search on streaming data series. *Proceedings of SSTD,* 83-101. Santorini, Greece.

Lloyd J.W (1987). Foundations of Logic Programming. Springer Verlag, second edition.

Lloyd, F. (1987). *Foundations of logic programming.* Berlin: Springer.

Lloyd, J. W., Sonenberg, L., & Topor, R. W. (1987). Integrity constraint checking in stratified databases. *JLP, 4*(4), 331-343.

Lo, M. L., & Ravishankar, C. V. (1994). Spatial Joins Using Seeded Trees. *In Proceedings of the 1994 ACM SIGMOD International Conference on Management of Data (SIGMOD 1994),* Minneapolis, Minnesota, USA, 24-27 May, 1994, (pp. 209-220), ACM Press.

Lo, M. L., & Ravishankar, C. V. (1996). Spatial Hash-Joins. *In Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data (SIGMOD 1996),* Montreal, Quebec, Canada, June 4-6, 1996, (pp. 247-258), ACM Press.

Lockemann, P. C., & Witte, R. (2005). Agents and Databases: Friends or Foes? *Proceedings of IDEAS'05,* (pp. 137-147).

Loo, B. T., Huebsch, R., Hellerstein, J. M., Stoica, I., & Shenker, S. (2004). Enhancing P2P file-sharing with an Internet-scale query processor. *Proceedings of the 30th International Conference on Very Large Data Bases* (pp. 432-443).

Loomis, M. E. S., & Chaudhri, A. B. (1997). *Object Databases in Practice.* Prentice-Hall.

Lorentzos, N. A., & Johnson, R. G. (1988). Extending relational algebra to manipulate temporal data. *Information Systems, 13*(3), 289-296.

Lorentzos, N. A., & Mitsopoulos, Y. G. (1997). SQL extension for interval data. *IEEE Transactions on Knowledge and Data Engineering, 9*(3), 480-499.

Lovett, S., & Simmons, L. C., & Kali, R. (1999). Guanxi Versus the Market: Ethics and Efficiency. *Journal of International Business Studies, 30*(2), 231–248.

Lowekamp, B., Tierney, B., & Cottrell, L. (2004). *A hierarchy of network performance characteristics for grid applications and services* (Global Grid Forum Proposed Recommendation No. GFD-R.023).

Lu, J., Dong, C. et al. (1999). Equivalent Object-Oriented Schema Evolution Approach Using the Path-Independence Language. *31st International Conference on Technology of Object-Oriented Languages and Systems,* Nanjing, China, IEEE Computer Society.

Lu, J., Ling, T. W., Chan, C. Y., & Chan, T. (2005). From region encoding to extended dewey: On efficient processing of XML twig pattern matching. In *Proc. VLDB,* 193-204.

Luck, M., McBurney, P., & Preist, C. (2004). A Manifesto for Agent Technology: Towards Next Generation Computing. *Autonomous Agents and Multi-Agent Systems, 9*(3), 203-252.

Lui, K. M. & Chan, K. C. C. (2003). Inexperienced software team and global software team. In A. Gunasekaran, O. Khalil, and S. M. Rahman (Eds.), *Knowledge and Information Technology Management: Human and Social Perspectives* (pp. 305-323). Hershey, PA: Idea Group Publishing.

Luján-Mora, S., & Trujillo, J. (2003). A comprehensive method for data warehouse design. *Proceedings of the 5th International Workshop on Design and Management of Data Warehouses.*

Luján-Mora, S., Trujillo, J., & Song, I. (2006). A UML profile for multidimensional modeling in data warehouses. *Data & Knowledge Engineering, 59*(3), 725–769.

Luscombe, N. M., Greenbaum, D., & Gerstein, M. (2001). What is Bioinformatics? A Proposed Definition

and Overview of the Field. *Methods of Information in Medicine, 40*(4), 346-358.

Lutz, M. (2007). Ontology-based descriptions for semantic discovery and composition of geoprocessing services. *Geoinformatica, 11*(1), 1–36.

Luu, T., Klemm, F., Podnar, I., Rajman, M., & Aberer, K. (2006). ALVIS Peers: A Scalable Full-text Peer-to-Peer Retrieval Engine. *Proc. of ACM P2PIR'06* (pp. 41–48), Arlington, VA, USA.

Lv, Q., Cao, P., Cohen, E., Li, K., & Shenker, S. (2002). Search and replication in unstructured peer-to-peer networks. *Proceedings of the 16th ACM International Conference on Supercomputing* (pp. 84-95).

Lv, Q., Josephson, W., Wang, Z., Charikar, M., & Li, K. (2007). Multi-probe LSH: Efficient indexing for high-dimensional similarity search. *Proceedings of VLDB,* 950-961. Vienna, Austria.

Ma, Q., & Wang, J. T. L. (1999). Biological Data Mining Using Bayesian Neural Networks: A Case Study. *International Journal on Artificial Intelligence Tools, Special Issue on Biocomputing, 8*(4), 433-451.

MacGregor, R., & Ko, I.-Y. (2003). Representing contextualized data using semantic Web tools. *Electronic Proceedings of the ISWC'03 Workshop on Practical and Scaleable Semantic Web Systems Services.* Retrieved September 16, 2004, from http://km.aifb.unikarlsruhe.de/ws/psss03/proceedings/macgregor-etal. Pdf

Machanavajjhala, A., Gehrke, J., Kifer, D., & Venkitasubramaniam, M. (2006). *l-diversity: Privacy beyond k-anonymity.* Proceedings of 22nd IEEE International Conference on Data Engineering (ICDE'06), Atlanta, GA.

Macqueen, J. B. (1967). *Some methods of classification and analysis of multivariate observations.* Paper presented at the the Fifth Berkeley Symposium on Mathematical Statistics and Probability.

Madden, S., Franklin, M. J, Hellerstein, J. M., & Hong, W. (2003). The Design of an Acquisitional Query Processor for Sensor Networks. *Proceedings of the 2003 ACM SIGMOD international conference on Management of data*, San Diego, CA, (pp. 491-502).

Madeira, S. C., & Oliveira, A. L. (2004). Biclustering Algorithms for Biological Data Analysis: A Survey. *IEEE/ACM Transactions Computational Biology and Bioinformatics, 1*(1), 24-45.

Madhavan, J., & Halevy, A. Y. (2003). Composing mappings among data sources. *VLDB* (pp. 572-583).

Madhavan, J., Halevy, A., Cohen, S., Dong, X., Jeffery, S., Ko, D., & Yu, C. (2006). Structured data meets the Web: a few observations. *IEEE Date Engineering Bulletin, 29*(4), 19-26.

Madiraju, P., & Sunderraman, R. (2004). *A mobile agent approach for global database constraint*

Madnick, S.E., & Siegel, M. (1998). The COntext INterchange (COIN) project: Data extraction and interpretation from semi-structured Web sources.

Magkanaraki, A., Tannen, V., Christophides, V., & Plexousakis, D. (2003). Viewing the semantic Web through RVL lenses. *Proceedings of the 2nd International Semantic Web Conference* (pp. 96-112).

Magnanelli, M., & Norrie, M. C. (2000). Databases for Agents and Agents for Databases. *Proc. 2nd Intl. Bi-Conference Workshop on Agent-Oriented Information Systems* (AOIS-2000).

Mahboubi, H., Aouiche, K., & Darmont, J. (2006a). *Un index de jointure pour les entrepôts de données XML.* Proceedings of the *6èmes Journées Francophones Extraction et Gestion des Connaissances (EGC 06), Lille, France, E-6, 89–94.*

Mahboubi, H., Aouiche, K., & Darmont, J. (2006b). Materialized view selection by query clustering in XML data warehouses. Proceedings of the *4th International Multiconference on Computer Science and Information Technology (CSIT 06), Amman, Jordan,* 68–77.

Mahmoudi, K., & Faïz, S. (2006_a). *Une approche distribuée pour l'extraction de connaissances: Application à l'enrichissement de l'aspect factuel des BDG.* Revue des Nouvelles Technologies de l'Information, Editions Cépaduès, Janvier, (pp. 107-118).

Mahmoudi, K., & Faïz, S. (2006_b). L'apport de l'information spatiale pour l'enrichissement des bases de données. INFORSID'2006, Hammamet, Tunisie, juin, (pp. 323-338).

Mahmoudi, K., & Faïz, S. (2006_c). SDET: A semantic data enrichment tool application to geographical databases. *International Conference On Signal-Image Technology*

& Internet–Based Systems (SITIS'2006), IEEE, ACM, Hammamet, Tunisie, Décembre, (pp. 88- 97).

Mahmoudi, K., & Faïz, S. (2007). L'outil SDET pour le complètement des données descriptives liées aux bases de données géographiques. Actes 7èmes Journées d'Extraction et Gestion des Connaissances (EGC'07), Session Demo, Namur Belgique, Janvier, (pp. 179-180).

Maier D. (1983) The Theory of Relational Databases, Computer Science Press.

Maier, A., & Simmen, D. (2001). *DB2 Optimization in Support of Full Text Search.* Bulletin of IEEE on Data Engineering.

Maier, D., Ullman, J. D., & Vardi, M. Y. (1984). On the foundation of the universal relation model. *ACM Transactions on Database Systems (TODS), 9*(2), 283-308.

Malinowski, E. (2009). Different kinds of hierarchies in multidimensional models.

Malinowski, E., & Zimányi, E. (2004). *Representing spatiality in a conceptual multidimensional model.* Proceedings of the 12th ACM Symposium on Advances in Geographic Information Systems, 12–21.

Malinowski, E., & Zimányi, E. (2006). *A conceptual solution for representing time in data warehouse dimensions.* Proceedings of the 3rd Asia-Pacific Conference on Conceptual Modeling, Hobart, Australia, 45–54.

Malinowski, E., & Zimányi, E. (2006). Hierarchies in a multidimensional model: From conceptual modeling to logical representation. *Data and Knowledge Engineering, 59*(2), 348-377.

Malinowski, E., & Zimányi. E. (2008). *Advanced data warehouse design: from conventional to spatial and temporal applications.* Springer.

Malinowski, E., & Zimányi. E. (2008). *Advanced data warehouse design: From conventional to spatial and temporal applications.* Springer

Mamoulis, N. Papadias, D., & Arkoumanis, D. (2004). Complex Spatial Query Processing. *GeoInformatica, 8*(4), 311-346.

Mamoulis, N., & Papadias, D. (2001). Multiway spatial joins. *ACM Transactions on Database Systems, 26*(4), 424-475.

Mamoulis, N., & Papadias, D. (2003). Slot Index Spatial Join. *IEEE Transactions Knowledge Data Engineering, 15*(1), 211-231.

Mamoulis, N., Bakiras, S., & Kalnis, P. (2005). Evaluation of Top-k OLAP Queries Using Aggregate R-trees. *In Proceedings of the Symposium on Spatial and Temporal Databases, (SSTD 2005),* Angra dos Reis, Brazil, 22-24 August, 2005, LNCS 3633, (pp. 236-253), Springer.

Maniatis, A., Vassiliadis, P., Skiadopoulos, S., Vassiliou, Y., Mavrogonatos, G., & Michalarias, I. (2005). A Presentation Model & Non-Traditional Visualization for OLAP. *International Journal of Data Warehousing and Mining,* 1(1), 1-36. Idea Group.

Manjunath, R., & Gurumurthy, K. S. (2002). Information geometry of differentially fed artificial neural networks. *TENCON'02.*

Mann, W. C., & Thompson, S. A. (1988). Rhetorical structure theory: Toward a functional theory of text organization. *An Interdisciplinary Journal for the Study of Text 8*(2), 243-281.

Manning, Ch. D., Raghavan, P., & Schütze, H. (2007). *Introduction to Information Retrieval*. Cambridge University Press.

Manola, F., & Miller, E. (2004). *RDF primer – W3C Recommendation 10 February 2004.* W3C (http://www.w3.org/TR/rdf-primer/).

Manolopoulos, Y., Nanopoulos, A., Papadopoulos, A. N., & Theodoridis, Y. (2006). *R-Trees: Theory and Applications.* Advanced Information and Knowledge Processing Series, Springer.

Manolopoulos, Y., Papadopoulos, A. N., & Vassilakopoulos, M. (2005). *Spatial Database: Technologies, Techniques and Trends.* Idea Group Publishing.

Mapping Objects to Data Models with the UML. (2001). Rational Software Corporation, Santa Clara, CA, USA.

Marcos, E., Vela, B. & Cavero J. M (2001). Extending UML for object-relational database design. In M. Gogolla & C. Kobryn (eds.). *UML 2001* (pp. 225-239). SpringerVerlag.

Marcos, E., Vela, B., & Cavero, J. M. (2003). A methodological approach for object-relational database design using UML. *Software and Systems Modeling, 2*(1), 59-72.

Marcu, D. (1999). Discourse trees are good indicators of importance in text. Mani and Maybury (eds.), *Advances in Automatic Text Summarization*, the MIT Press, (pp. 123-136).

Margetts, H. (2001). The Cyber Party. *The Causes and Consequences of Organizational Innovation in European Political Parties Workshop, ECPR Joint Sessions of Workshops,* Grenoble.

Marian, A., Abiteboul, S., Cobéna, G., & Mignet, L. (2001). Change-centric management of versions in an XML warehouse. *In 27th International Conference on Very Large Databases (VLDB'01)* (pp. 581-590). Roma, Italy: Morgan Kauffman.

Mark, D.M., Skupin, A., & Smith, B. (2001). Features, objects, and other things: Ontological distinctions in the geographic domain. In *Spatial information theory: Foundations of geographic information science* [Lecture Notes in Computer Science (2205)] (pp. 488–502). Berlin: Springer-Verlag.

Markowitz, V., & Shoshani, A. (1992). Representing Extended Entity-Relationship Structures in Relational Databases: A Modular Approach. *ACM Transactions on Database Systems, 17*(3), 423-464.

Marquis, P., & Porquet, N. (2003). Resource-bounded paraconsistent inference. *Annals of Mathematics and Artificial Intelligence, 39*(4), 349-384.

Martin, P., Powley, X. U., & Tian, W. (2006). Automated configuration of multiple buffer pools. *The Computer Journal. IEEE, 49*(4).

Martinenghi, D. (2005). *Advanced techniques for efficient data integrity checking* [doctoral thesis]. Roskilde University.

Martinenghi, D., Christiansen, H., & Decker, H. (2006). *Integrity checking and maintenance in relational and deductive databases—and beyond.* In Z. Ma (Ed.) Intelligent databases: Technologies and Applications (pp. 238-285). Hershey PA: IGI Global.

Martínez González, M., de la Fuente, P., & Derniame, J.-C. (2003b). XML as a means to support information extraction from legal documents. *International Journal of Computer Systems Science and Engineering, 18*(5), 263-277.

Martínez González, M., de la Fuente, P., Derniame, J., & Pedrero, A. (2003a). Relationship-based dynamic versioning of evolving legal documents. In *Web-knowledge Management and Decision Support*, volume 2543 of *Lecture Notes on Artificial Intelligence* (pp. 298—314). Springer-Verlag.

Massonnet, D., & Feigl, K. L. (1995). Radar interferometry and its applications to changes in the Earth surface. *Reviews of Geophysics, 36,* 441–500.

Massonnet, D., Vadon, H., & Rossi, M. (1996). Reduction of the need for phase unwrapping in radar interferometry. *IEEE Transaction on Geosciences and Remote Sensing, 34,* 489-497.

Matias, Y., Vitter, J. S., & Wang, M. (1998). Wavelet-based histograms for selectivity estimation. *Proceedings of the 1998 ACM International Conference on Management of Data* (pp. 448-459).

Matias, Y., Vitter, J. S., & Wang, M. (2000). Dynamic maintenance of wavelet-based histograms. *Proceedings of the 26th International Conference on Very Large Data Bases* (pp. 101-110).

Matousek, K., Mordacik, J., & Janku, L. (2001). On implementing the data warehouse: GIS integration. *Proceedings of the World Multiconference on Systemics, Cybernetics and Informatics: Information Systems Development, 1,* 206-210.

Mayol, E., & Teniente, E. (2003). Consistency preserving updates in deductive databases. *Data and Knowledge Engineering, 47*(1), 61-103.

Mazón, J.N., & Trujillo, J. (2006). *Enriching datawarehouse dimension hierarchies by using semantic relations.* Proceedings of the 23rd British National Conference on Databases (BNCOD 06), Belfast, Northern Ireland, 4042, 278–281.

Mazumdar, S. (1993). *Optimizing distributed integrity constraints.* Proceedings of the 3rd International Symposium on Database Systems for Advanced Applications, Korea, 327–334.

Mazumdar, S., & Chrysanthis, P. K. (2004). Localization of integrity constraints in mobile databases and specification in PRO-MOTION. *Proceedings of the Mobile Networks and Applications,* (pp. 481-490).

McBrien, P.J., & Poulovassilis, A. (2003). *Data integration by bi-directional schema transformation rules.* Proceedings of the ICDE'03.

McCann, R., Doan, A., Varadarajan, V., & Kramnik, A. (2003). Building data integration systems via mass collaboration. *International Workshop on the Web and Databases.* California: WebDB

McCarroll, N. J. (1995). *Semantic integrity enforcement in parallel database machines.* Ph.D. Thesis, University of Sheffield.

McCarroll, N.F. (1995). *Semantic integrity enforcement in parallel database machines* [doctoral thesis]. UK: University of Sheffield.

McCarthy, H., & Miller, P., & Skidmore, P. (Eds.) (2004). *Network logic: Who governs in an interconnected world?* London: Demos. Individual essays available at http://www.demos.co.uk/catalogue/networks/.

McCarthy, W. E. (1982). The REA accounting model: A generalized framework for accounting systems in a shared data environment. The Accounting Review, 57(3), 554-578.

McCune, W. W., & Henschen, L. J. (1989). Maintaining state constraints in relational databases: A proof theoretic basis. *Journal of the Association for Computing Machinery, 36*(1), 46-68.

McFadden, F. R., Hoffer, J. A., & Prescott, M. B. (1999). *Modern Database Management.* Boston, MA: Addison-Wesley Publishing.

McGuinness, D. L., Fikes, R., Hendler, J., & Stein, L. A. (2002). DAML+OIL: An ontology language for the Semantic Web. *IEEE Intelligent Systems, 17*(5), 72-80.

McHugh, J., & Widom, J. (1999). Query optimization for XML. In *Proc. of VLDB.*

McIlraith, S.A., Son, T.C., & Zeng, H. (2001). Semantic Web services. *IEEE Intelligent Systems, 16*(2), 46–53.

McKenzie, E., & Snodgrass, R. (1990). Schema Evolution and the Relational Algebra. *Inf. Syst., 15*(2), 207-232.

McKenzie, E., & Snodgrass, R. T. (1991). An evaluation of relational algebras incorporating the time dimension in databases. *ACM Computing Surveys, 23*(4), 501-543.

McKenzie, L.E., & Snodgrass, R.T. (1990). Schema evolution and the relational algebra. *Information Systems Journal, 15*(2), 207–232.

McLaughlin, B., & Edelson, J. (2006). *Java and XML.* Third Edition. O'Reilly.

Mealling, M. (2003). Object Name Service (ONS) 1.0. *Auto-ID Center, Boston MA, USA.*

Medcraft, P., Schiel, U., & Baptista, P. (2003). DIA: Data integration using agents. *Databases and Information Systems Integration. ICEIS,* 79-86.

Megretskaya, I. A. (1988). Construction of natural classification tests for knowledge base generation. In Y. Pecherskij (Ed.), *Problems of Expert System Application in the National Economy* (pp. 89-93). Kishinev: Institute mathematics of Moldavia's Academy of Sciences.

Mehoudj, K., & Ou-Halima, M. (1995). Migrating data-oriented applications to a relational database management system. *Proceedings of the Third International Workshop on Advances in Databases and Object-Oriented Databases* (pp. 102-108).

Meier, W. (2002). *eXist: An open source native XML database.* Proceedings of the Web, Web-Services, and Database Systems, NODe 2002 Web and Database-Related Workshops, Erfurt, Germany, 2593, 169–183.

Meka, A., & Singh, A. K. (2005). DIST: A distributed spatio-temporal index structure for sensor networks. *Proc. Int. Conf. on Information and Knowledge Management* (CIKM' 2005), 139-146.

Melton, J., & Simon, A. (2001). SQL: 1999. *Understanding relational language components.* Morgan Kaufmann Publishers.

Mena, E., Kashyap, V., Sheth, A., & Illarramendi, A. (2000). Observer: An approach for query processing in global information systems based on interoperation across pre-existing ontologies. *Kluwer Academic Publishers,* Boston, (pp. 1-49).

Mendelzon, A.O., & Vaisman, A.A. (2000). *Temporal queries in OLAP.* Proceedings of the 26th International Conference on Very Large Data Bases (VLDB 00), Cairo, Egypt, 242–253.

Meng, W., Yu, C., & Liu, K.-L. (2002). Building efficient and effective metasearch engines. *ACM Computing Surveys, 34*(1), 48-84.

Mens, T., & Tourwé, T. (2004). A survey of software refactoring. *IEEE Transactions on Software Engineering, 30*(2), 126-139.

Meo, R. (2005). *Inductive databases: Towards a new generation of databases for knowledge discovery.* Proceedings of the DEXA'05: 16th International Workshop on Databases and Expert System Applications, 1003–1007.

Meo, R., Psaila, G., & Ceri, S. (1996). A new SQL-like operator for mining association rules. *The VLDB Journal,* 122–133.

Merton, R. (1979). Foreword. In E. Garfield (Ed.), *Citation Indexing: Its Theory and Application in Science, Technology, and Humanities, vii–xi.* John Wiley and Sons, New York.

MetaCarta (2005): GTS versus MetaCarta GeoTagger. MetaCarta, Inc.

Meyer, J-J. Ch. (2004). Intelligent Agents: Issues and Logics. *Logics for Emerging Applications of Databases,* (pp. 131-165).

Microsoft (2007) SharePoint Server home page retrieved 14th March, 2008, from http://office.microsoft.com/en-us/sharepointserver/FX100492001033.aspx

Microsoft (2007). SQL Server Full Text Search Engine, http://technet.microsoft.com/en-us/library/ms345119.aspx

Middleton, S. E., Shadbolt, N. R., & Roure D. C. de (2004). Ontological User Profiling in Recommender Systems. *ACM Transaction on Information Systems, 22*(1), 54-88.

Milán-Franco, J. M., Jiménez-Peris, R., Patiño-Martínez, M., & Kemme, B. (2004). Adaptive Middleware for Data Replication. *International Conference on Middleware,* (pp. 175-194).

Mill, J. S. (1900). *The system of logic.* Moscow, Russia: Russian Publishing Company "Book Affair".

Miller, G. (1990). Wordnet: An on-line lexical database. *International Journal of Lexicogra-phy (special issue), 3*(4), 235-312.

Miller, R. J., Haas, L. M., & Hernandez, M. L. (2000). *Schema mapping as query discovery.* Proceedings of the 26th VLDB Conference, Cairo, Egypt.

Millham, R. (2005). *Evolution of batch-oriented COBOL systems into object-oriented systems through the unified modelling language.* Unpublished doctoral dissertation, De Montfort University, Leicester, England.

Milo, T., & Suciu, D. (1999). *Index structures for path expressions.* Proceedings of the 7th International Conference on Database Theory (ICDT 99), Jerusalem, Israel, 1540, 277–295.

Min, J., Chung, C., & Shim, K. (2005). An adaptive path index for XML data using the query workload. *Information Systems, 30*(6), 467–487.

Minker, J. (1999). Logic and databases: A 20 year retrospective. In F. Pirri & H. Levesque (Eds.), *Logical foundations for cognitive agents* (pp. 234-299). Berlin: Springer.

Mishra, N., & Sandler, M. (2006). Privacy via pseudorandom sketches. *Proceedings of ACM Symposium on Principles of Database Systems (PODS'06)* (pp. 143-152).

Mitchell, T. (1997). *Machine learning.* New York: Addison-Wesley.

Mitchell, T. M. (1982). Generalization as search. *Artificial Intelligence, 18,* 203-226.

Mitra, N., (ed.) (2003). *SOAP Version 1.2 Part 0: Primer – W3C Recommendation 24 June 2003.* W3C (http://www.w3.org/TR/soap12-part0/).

Mlynkova, I. (2007). A Journey towards More Efficient Processing of XML Data in (O)RDBMS. In *CIT'07: Proceedings of the 7th IEEE International Conference on Computer and Information Technology,* (pp. 23–28), Aizu-Wakamatsu City, Fukushima, Japan. IEEE Computer Society.

Mlynkova, I., & Pokorny, J. (2004). From XML Schema to Object-Relational Database – an XML Schema-Driven Mapping Algorithm. In *ICWI'04: Proceedings of the 3rd IADIS International Conference WWW/Internet,* (pp. 115–122), Madrid, Spain. IADIS.

Mlynkova, I., Toman, K., & Pokorny, J. (2006). Statistical Analysis of Real XML Data Collections. In *COMAD'06: Proceedings of the 13th International Conference on Management of Data,* (pp. 20–31), New Delhi, India. Tata McGraw-Hill Publishing Company Limited.

Moed, H. F. (2005). Citation Analysis in Research Evaluation. *Information Science and Knowledge Management.* Springer, Dordrecht.

Mokbel, M. F., Ghanem T. M., & Aref, W. G. (2003). Spatio-temporal access method. *IEEE Data Engineering Bulletin, 26*(2): 40-49.

Mokbel, M., & Aref, W., (2005). GPAC: Generic and Progressive Processing of Mobile Queries over Mobile Data. *Proc. of the 6th ACM Int. Conf. on Mobile Data Management,* (pp. 155-163).

Mokbel, M.F., Lu, M., & Aref, W.G. (2004). *Hash-merge join: A nonblocking join algorithm for producing fast and early join results.* Proceedings of the 20th International Conference on Data Engineering ICDE2004, 251–263.

Moody, D., & Kortink, M. (2000). From enterprise models to dimensional models: A methodology for data warehouse and data mart design. *Proceedings of the 2nd International Workshop on Design and Management of Data Warehouses,* 6.

Mookerjee, V.S., & Chiang, I.R. (2002). A dynamic coordination policy for software system construction. *IEEE Transactions on Software Engineering, 28*(7), 684–694.

Morpheus file sharing system. (n.d.). Retrieved from http://www.musiccity.com

Morzy, T., & Wrembel, R. (2004). *On querying versions of multiversion data warehouse*. Proceedings of the 7th ACM International Workshop on Data Warehousing and OLAP (DOLAP 04), Washington, DC, 92–101.

Motik, B., Maedche, A., & Volz, R. (2002). *Conceptual Modeling Approach for Semantics-Driven Enterprise Applications.* ODBASE 2002.

Mouratidis, K., Bakiras, S., & Papadias, D. (2006). *Continuous monitoring of top-k queries over sliding windows.* Proceedings of the SIGMOD Conference 2006, 635–646.

Mouratidis, K., Yiu, M. L., Papadias, D., & Mamoulis, N. (2006). Continuous Nearest Neighbor Monitoring in Road Networks. *In Proceedings of the Very Large Data Bases Conference (VLDB 2006),* Seoul, Korea, 12-15 September, 2006, (pp. 43-54). Morgan Kaufmann, San Francisco.

Mueller, E. T. (2006). *Common Sense Reasoning.* UK: Elsevier.

Muller, N. J. (1993). *Computerized document imaging systems: technology and applications.* Boston, MA: Artech House.

Mullins, C. S. (2005). Database administration. *The complete guide to practices and procedures.* Addison-Wesley.

Munch, B. P. (1995). *Versioning in a software Engineering Database – the Change Oriented Way Division of Computer Systems and Telematics.* The Norwegian Institute of Technology.

Muñoz-Escoí, F. D., Pla-Civera, J., Ruiz-Fuertes, M. I., Irún-Briz, L., Decker, H., Armendáriz-Íñigo, J. E., & González de Mendívil, J. R. (2006). Managing Transaction Conflicts in Middleware-based Database Replication Architectures. *Symposium on Reliable Distributed Systems,* (pp. 401-410).

Muralikrishna, M., & DeWitt, D. J. (1998). Equi-depth histograms for estimating selectivity factors for multi-dimensional queries. *Proceedings of the 1988 ACM International Conference on Management of Data* (pp. 28-36).

Nabli, A., Soussi, A., Feki, J., Ben-Abdallah, H., & Gargouri, F. (2005). *Towards an automatic data mart design.* Proceedings of the 7th International Conference on Enterprise Information Systems (ICEIS 05), Miami, Florida, 226–231.

Naiburg, E. J., & Maksimchuk, R. A. (2002). *UML for Database Design.* Boston, MA: Addison-Wesley Publishing.

Naidenova, X. A. (1996). Reducing machine learning tasks to the approximation of a given classification on a given set of examples. *In Proceedings of the 5th National Conference on Artificial Intelligence, 1,* 275-279. Tatarstan: Kazan University Press.

Naidenova, X. A. (2006). An incremental learning algorithm for inferring logical rules from examples in the framework of common reasoning process. In E. Triantaphyllou, & G. Felici (Eds.), *Data mining and knowledge discovery approaches based on rule induction techniques* (pp. 89-146). USA: Springer.

Naidenova, X. A. (2007). Reducing a class of machine learning algorithms to logical commonsense reasoning operations. In G. Felici, & C. Vercellis (Eds.), *Mathematical Methods for Knowledge Discovery and Data Mining* (pp. 41-64). Hershey – New York: ISR Press.

Naidenova, X. A., & Polegaeva, X. A. (1986). An algorithm of finding best diagnostic tests. In G. E. Mintz, & P. P. Lorents (Eds.), *The 4th All Union Conference on Application of Mathematical Logic Methods* (pp. 63-67). Tallinn, Estonia: Institute of Cybernetics, National Academy of Sciences of Estonia.

Nakamura, H., & Johnson R. E. (1998). Adaptive Framework for the REA Accounting Model, Proceedings of the OOPSLA'98 Business Object Workshop IV, http://jeffsutherland.com/oopsla98/nakamura.html.

Nandit, S., Levy, E., Korth, H. F., & Silberschatz, A. (1994). Adaptive commitment for real-time distributed transaction. *3rd International Conference on Information and Knowledge Management,* 187-194. *Gaithersburg, MD.*

Nascimento, M. A., & Silva, J. R. O.(1998). Towards historical R-trees. *Proc. of the ACM Symposium on Applied Computing (SAC'1998),* 235-240. Atlanta, GA.

Nascimento, M. A., Silva, J. R. O., & Theodoridis, Y. (1999). Evaluation of access structures for discretely moving points. *Proc. of the International Workshop on SpatioTemporal Database Management (STDBM'1999),* 171-188.

National Research Council (2001). Embedded, Everywhere: A Research Agenda for Networked Systems of Embedded Computers. *National Academy Press, Washington DC, USA.*

Navathe, S. B., & Ahmed, R. (1987). TSQL-A language interface for history databases. *Proceedings of the Conference on Temporal Aspects in Information Systems,* 113-128.

Necasky, M. (2006). Conceptual Modeling for XML: A Survey. *Proceedings of the DATESO 2006 Annual International Workshop on Databases, Texts, Specifications, and Objects,* Desna-Cerna Ricka, Czech Republic.

Needleman, S. B., & Wunch, C. D. (1970). Needleman-Wunch algorithm for sequence similarity searches. *Journal of Molecular Biology, 48,* 443-453.

Nejdl, W., Wolpers, M., Siberski, W., Schmitz, C., Schlosser, M., Brunkhorst, I., et al. (2003). Super-peer-based routing and clustering strategies for RDF-based P2P networks. *Proceedings of the 12th International World Wide Web Conference* (pp. 536-543).

Neuliep, J. W. (2000). *Intercultural communication: A contextual approach*. Boston: Houghton Mifflin.

Ng, E. K. K., Fu, A. W.-c., & Wong, R. C.-W. (2005). Projective Clustering by Histograms. *IEEE Transactions on Knowledge and Data Engineering, 17*(3), 369-383.

Ng, R. T., & Han, J. (1994). *Efficient and Effective Clustering Methods for Spatial Data Mining.* Paper presented at VLDB'94: the 20th International Conference on Very Large Data Bases, San Francisco, CA, USA.

Nguyen, G., & Rieu, D. (1988). Schema evolution in object-oriented database systems. *Rapports de Recherche, 947.*

Nicolas, J. M. (1982). Logic for improving integrity checking in relational databases. *Acta Informatica, 18*(3), 227-253.

Nierman, A., & Jagadish, H. (2002). Evaluating structural similarity in XML documents. *Proceedings of the ACM SIGMOD International Workshop on the Web and Databases (WebDB)* (pp. 61-66).

Nieva-García, O., & Benítez-Guerrero, E. (2006). *On querying inductive databases to mine decision rules.* Proceedings of the 6th International Conference on Data Mining and Information Systems (ICDIS'06), 55–65.

Nievergelt, J., & Widmayer, P. (1997). Spatial data structures: concepts and design choices. M. van Kreveld, J. T. NievergeltRoos, & P. Widmayer (Eds.), *Algorithmic Foundations of Geographic Information Systems* (pp. 153-197). Berlin: Springer Verlag.

Nijssen, S., & De Raedt, L.(2006). *IQL: A proposal for an inductive query language*. Proceedings of the 5th International Workshop on Knowledge Discovery (KDID'06), 107–118.

NIST Wireless Communication Technologies Group (2005). *Wireless Ad Hoc Networks Bibliography,* http://w3.antd.nist.gov/wctg/manet/manet_bibliog.html

Noirhomme-Fraiture, M. (2004). Visualisation of Symbolic Data. *Workshop on Applications of Symbolic Data Analysis.* Lisboa Portugal. Retrieved from http://www.

info.fundp.ac.be/asso/dissem/W-ASSO-Lisbon-Visu.pdf. on May 2008.

Nonaka, I., & Takeuchi, H. (1995). *The Knowledge-Creating Company.* New York: Oxford University Press, Inc.

Norvag, K. (2005). Query operators in XML databases. In L. C. Rivero, J. H. Doorn, & V. E. Ferraggine (Ed.), *Encyclopedia of Database Technologies and Applications 2005* (pp. 500-505). Idea Group.

Noy, F. N., Fergerson, R. W., & Musen, M. A. (2000). The knowledge model of protégé-2000: Combining interoperability and flexibility. In *Knowledge Acquisition, Modeling, and Management – Proceedings of EKAW'2000* (LNCS 1937), R. Dieng & O. Corby (Eds.). Berlin: Springer.

Noy, N.F. (2004). Semantic integration: A survey of ontology-based approaches. *SIGMOD Rec., 33*(4), 65–70.

Ntoulas, A., Zerfos, P., & Cho, J. (2005). Downloading textual hidden web content through keyword queries. *5th ACM/IEEE-CS Joint Conference on Digital Libraries (JCDL'05),* 100-109.

O'Leary, D. E. (2004). On the relationship between REA and SAP. International Journal of Accounting Information Systems, 5(1), 65-81.

O'Ryan, C., Schmidt, D.C., & Noseworthy, J.R. (2002). Patterns and performance of a CORBA event service for large-scale distributed interactive simulations. *International Journal of Computer Systems Science and Engineering, 17.*

OASIS SOA RM (2007) www.oasis-open.org/committees/tc_home.php?wg_abbrev=soa-rm

Object Management Group. (2002). Common warehouse metamodel. Retrieved from http://www.omg.org/docs/formal/03-03-02.pdf

Object-Oriented Database. *Conceptual Modeling - ER 2002, 21st International Conference on Conceptual Modeling,* Tampere, Finland, October 7-11, 2002, Proceedings, Tampere, Finland, Springer.

Obrst, L. (2003). Ontologies for semantically interoperable systems. *CIKM'03* (pp. 366-369).

Oezsu, M. T., & Valduriez, P. (1999). *Principles of Distributed Database Systems*, Prentice Hall.

Office for Civil Rights (OCR). (2003). *Summary of the HIPAA privacy rule.* U.S. Department of Health and Human Services.

Ogilvie, P., & Callan, J. (2001). The effectiveness of query expansion for distributed information retrieval. *Proceedings of the 10th ACM International Conference on Information and Knowledge Management* (pp. 183-190).

Olivé, A. (2003). Integrity constraints definition in object-oriented conceptual modeling languages. *Conceptual Modeling - ER 2003, 22nd Int. Conf. on Conceptual Modeling.* Chicago, IL, USA.

Oliveira, S. R. M., & Zaïane, O. R. (2003). Protecting sensitive knowledge by data sanitization. *Proceedings of the 3rd IEEE International Conference on Data Mining (ICDM'03)* (pp. 613-616).

Omelayenko, B., & Fensel, D. (2001). An Analysis of Integration Problems of XML-Based Catalogues for B2B E-commerce. In *Proceedings of the 9th IFIP 2.6 Working Conference on Database (DS-9), Semantic Issues in e-commerce Systems.* Hong Kong, China, 2001

OMG Unified Modeling Language Specification. Version 2.1.1. (2007). Retrieved October 6, 2007, from http://www.omg.org/technology/documents/formal/uml.htm.

Oracle (2004). *Oracle Database Lite, Administration and Deployment Guide 10g (10.0.0).* Oracle Corporation.

Oracle Database Application Developer's Guide - Object-Relational Features. (2005). Retrieved October 6, 2007, from http://download.oracle.com/docs/cd/B19306_01/appdev.102/b14260.pdf.

Oracle Text (2007). *Oracle Text Product Description,* homepage: http://www.oracle.com/technology/products/text/index.html

Oracle9i application developer's guide: Object-relational features release 2 (9.2). (n.d.). Retrieved from http://download-east.oracle.com/docs/cd/B10501_01/appdev.920/a96594/adobjdes.htm#441636

Ore, O. (1944). Galois connexions. *Trans. Amer. Math. Society, 55*(1), 493-513.

Osborn, S. (1989). The role of Polymorphism in schema evolution in an object-oriented database. *IEEE Transactions on Knowledge Data Engineering, 1*(3).

Overstreet, R. (2003). *Protecting yourself from SQL injection attacks.* Retrieved June 16, 2006, from http://www.4guysfromrolla.com/webtech/061902-1.shtml

Ozsoyoglu, M. Z., & Yuan, L-Y (1987). A new normal form for nested relations. *ACM Transactions on Database Systems, 12*(1), January 1987.

Ozsu, M. T., & Valduriez, P. (1999). *Principles of distributed database systems*, (2nd ed.), Prentice Hall.

Özsu, M. T., Peters, R. J., Szafron, D., Irani, B., Lipka, A, & Muñoz, A. (1995). TIGUKAT: A uniform behavioral objectbase management system. *VLDB Journal, 4*(3), 445-492.

Ozsu, M.T., & Valduriez, P. (1999). *Principles of distributed database systems* (2nd Edition). Prentice Hall, Inc.

Pan, D., & Shen, J. Y. (2005). Ontology service-based architecture for continuous knowledge discovery. In *Proceedings of International Conference on Machine Learning and Cybernetics, 4,* 2155-2160.

Pan, Z., & Heflin, J. (2003). DLDB: Extending relational databases to support semantic web queries. *Electronic Proceedings of the ISWC'03 Workshop on Practical and Scalable Semantic web Systems Services.* Retrieved September 16, 2004, from http://km.aifb.uni-karlsruhe.de/ws/psss03/proceedings/pan-et-al.pdf

Panagiotakis, S., Koutsopoulou, M., Alonistioti, A., Houssos, N., Gazis, V., & Merakos, L. (2003). An advanced service provision framework for reconfigurable mobile networks. *International Journal of Mobile Communications, 1*(4), 425-438.

Papadias, D., Kalnis, P., Zhang, J., & Tao, Y. (2001). Efficient OLAP Operations in Spatial Data Warehouses. *In Proceedings of the Symposium on Spatial and Temporal Databases, (SSTD 2001),* Redondo Beach, CA, USA, 12-17 July, 2001, LNCS 2121, (pp. 443-459). Springer.

Papadias, D., Zhang, J., Mamoulis, N., & Tao, Y. (2003). Query processing in spatial network databases. *Proceedings of 29th International Conference on Very Large Databases* (pp. 802-813).

Papadomanolakis, S., & Ailamaki, A. (2004). Autopart: Automating schema design for large scientific databases using data partitioning. *Proceedings of the 16th International Conference on Scientific and Statistical Database Management,* (pp. 383–392).

Papadopoulos, D., Kollios, G., Gunopulos, D., & Tsotras, V. J. (2002). Indexing mobile objects using duality transforms. *IEEE Data Engineering Bulletin, 25*(2), 18-24.

Papoulis, A. (1994). *Probability, random variables, and stochastic processes* (2nd ed.). McGraw-Hill.

Parent, C., Spaccapietra, S., & Zimányi, E. (1999). *Spatio-temporal conceptual models: Data structures + space + time.* Proceedings of the GIS '99: 7th ACM International Symposium on Advances in Geographic Information Systems, New York, 26–33.

Parent, C., Spaccapietra, S., & Zimányi, E. (2005). The murmur project: Modeling and querying multi-representation spatio-temporal databases. *Information Systems.*

Parent, C., Spaccapietra, S., & Zimányi, E. (2006). *Conceptual modeling for traditional and spatio-temporal applications: The MADS approach.* Springer.

Park, B., & Kargupta, H. (2002). Distributed data mining: Algorithms, systems, and applications. In Nong Ye (Ed.), *Data mining handbook* (pp. 341–358). IEA.

Park, B.K., Han, H., & Song, I.Y. (2005). *XML-OLAP: A multidimensional analysis framework for XML warehouses.* Proceedings of the 7th International Conference on Data Warehousing and Knowledge Discovery, (DaWaK 05), Copenhagen, Denmark, 3589, 32–42.

Park, J., & Ram, S. (2004). Information systems interoperability: What lies beneath? *ACM Transactions on Information Systems, 22*(4), 595-632.

Park, S. H., & Luo, Y. (2001). Guanxi and Organizational Dynamics: Organizational Networking in China Firms. *Strategic Management Journal, 22*(5), 455–477.

Park, S., Chu, W. W., Yoon, J., & Hsu, C. (2000). Efficient searches for similar subsequences of different lengths in sequence databases. *Proceedings of IEEE ICDE,* 23-32. San Diego, CA, USA.

Parsons, L., Haque, E., & Liu, H. (2004). Subspace clustering for high dimensional data: A review. *ACM-SIGKDD Explorations Newsletter, 6*(1), 90-105.

Parthasarathy, S. et al. (1999). Incremental and Interactive Sequence Mining. *Proceedings of the 8th International Conference on Information and Knowledge Management,* USA, (pp. 251-258).

Pasquier N., Bastide Y., Taouil R. & Lakhal L. (1999a). Efficient Mining of Association Rules using Closed Itemset Lattices. Journal of Information Systems, 24(1), 25-46.

Pasquier N., Bastide Y., Taouil R. & Lakhal L. (1999b). Discovering Frequent Closed Itemsets for Association Rules. Proceedings of International Conference on Database Theory (ICDT).

Pasquier N., Bastide Y., Taouil R. & Lakhal L. (1999c). Closed Set based Discovery of Small Covers for Association Rules. Proceedings of BDA conference, 361-381.

Patel, J. M., & DeWitt, D. J. (1996). Partition Based Spatial-Merge Join. *In Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data (SIGMOD 1996)*, Montreal, Quebec, Canada, June 4-6, 1996, (pp. 259-270), ACM Press.

Patiño-Martínez, M., Jiménez-Peris, R., Kemme, B., & Alonso, G. (2005). MIDDLE-R: Consistent Database Replication at the Middleware Level. *ACM Transactions on Computer Systems, 23*(4), 375-423.

Paton, N. W., & Diaz, O. (1999). Active Database Systems. *ACM Computing Surveys, 31*(1), 63-103.

Pedersen, K., & Saglie, J. (2005). New Technology in Ageing Parties: Internet Use among Danish and Norwegian Party Members. *Party Politics, 11*(3), 359-377.

Pedersen, T., Jensen, C.S., & Dyreson, C. (2001). A foundation for capturing and querying complex multidimensional data. *Information Systems, 26*(5), 383–423.

Pedersen, T., Patwardhan, S., & Michelizzi, J. (2004). Wordnet::similarity: Measuring the relatedness of concepts. *Proceedings of Fifth Annual Meeting of the North American Chapter of the ACL (NAACL-04).*

Pedersen, T.B., & Tryfona, N. (2001). *Pre-aggregation in spatial data warehouses.* Proceedings of the 7th International Symposium on Advances in Spatial and Temporal Databases, 460–478.

Pedone, F., Guerraoui, R., & Schiper, A. (2003). The Database State Machine Approach. *Distributed and Parallel Databases, 14,* 71-98.

Pei J., Han J. & Mao R., (2000). Closet: An Efficient Algorithm for Mining Frequent Closed Itemsets. Proceedings of the ACM-SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery, 21-30.

Pei, J. et al. (2001). PrefixSpan: Mining Sequential Patterns Efficiently by Prefix-Projected Pattern Growth. *Proceedings of the 2001 International Conference Data Engineering,* Germany, (pp. 215-224).

Pei, J. et al. (2002). Mining Sequential Patterns with Constraints in Large Databases. *Proceedings of the 11th ACM International Conference on Information and Knowledge Management*, USA, (pp. 18-25).

Peissig, J. (2004). guidePort – An Information and Guidance System. In K. Kyamakya, (Ed.), *Proceedings of 1st Workshop on Positioning, Navigation and Communication 2004 (WPNC 04),* University of Hanover, Germany, March 26, 2004, number 0.1 in Hannoversche Beitr¨age zur Nachrichtentechnik, (pp. 1–17), Aachen. NICCIMON, IEEE, VDI, Shaker Verlag GmbH.

Pelanis, M., Saltenis, S., & Jensen, C. S. (2006). Indexing the past, present and anticipated future positions of moving objects. *ACM Transactions on Database Systems 31*(1), 255-298

Peled, A. (2001). Networks, Coalition, and Institution: the Politics of Technological Innovation in the Public Sector. *Information Technology & People, 14*(2), 184-205.

Pemberton, S. et al. (2002) XHTML *1.0 The Extensible HyperText Markup Language (Second Edition).* W3C. http://www.w3.org/TR/xhtml1/.

Penney, D.J., & Stein, J. (1987). *Class modification in the gemstone object-oriented DBMS*. Proceedings of the OOPSLA'87, 111–117.

Perez-Pena, R. (2003, April 4). An early warning system for diseases in New York. *New York Times.*

Perez-Schofield, J. B. G., Rosello, E. G. et al. (2002). Managing Schema evolution in a container-based persistent system. *Softw., Pract. Exper., 32*(14), 1395-1410.

Perich, F., Joshi, A., Finin, T., & Yesha, Y. (2004). On Data Management in Pervasive Computing Environments. *IEEE Transactions on Knowledge and Data Engineering,* IEEE, 16(5), 621-634.

Pestana, G., Mira da Silva, M., & Bédard, Y. (2005). *Spatial OLAP modeling: An overview base on spatial objects changing over time.* Proceedings of the IEEE 3rd International Conference on Computational Cybernetics, 149–154.

Peters, R. J., & Özsu, M. T. (1997). An axiomatic model of dynamic schema evolution in objectbase systems. *ACM Trans. Database Syst., 22*(1), 75-114.

Peters, R. J., & Özsu, M. T. (1997). An Axiomatic Model of Dynamic Schema Evolution in Object-based Systems. *ACM TODS, 22*(1), 75-114

Petit, J.-M., Kouloumdjian, J., Bouliaut, J.-F., & Toumani, F. (1994). Using queries to improve database reverse engineering. *Proceedings of the 13th International Conference on ER Approach*.

Pfoser, D. (2002). Indexing the trajectories of moving object. *IEEE Data Engineering Bulletin, 25*(2): 3-9.

Pfoser, D., & Jensen, C.S. (2003). *Indexing of network constrained moving objects.* Proceedings of the ACM-GIS 2003, 25–32.

Pfoser, D., Jensen, C., & Theodoridis, Y. (2000). Novel approaches to the indexing of moving object trajectories. *Proc. of the 26th International Conference on Very Large Databases (VLDB'2000),* 189-200. Cairo, Egypt.

Phillips, D., Zhang, N., Ilyas, I.F., & Ozsu, M.T. (2006). *InterJoin: Exploiting indexes and materialized views in XPath evaluation*. Proceedings of the 18th International Conference on Scientific and Statistical Database Management (SSDBM 06), Vienna, Austria, 13–22.

Piaget, J., & Inhelder, B. (1959). *La genèse des structure logiques élémentaires: classifications et sériations.* Neuchâtel: Delachaux & Niestlé.

Piatetsky-Shapiro, G., & Connell, C. (1984). Accurate estimation of the number of tuples satisfying a condition. *Proceedings of the 1984 ACM International Conference on Management of Data* (pp. 256-266).

Piatetsky-Shapiro, G., & Tamayo, P. (2003). Microarray data mining: Facing the challenges.

PL/SQL user's guide and reference 10g release 1 (10.1). (n.d.). Retrieved from http://download-east.oracle.com/docs/cd/B14117_01/appdev.101/b10807/05_colls.htm#i20446

Plazanet, C. (1996). *Enrichissement des bases de données géographiques: analyse de la géométrie des objets linéaires pour la généralisation cartographique (application aux routes).* PhD thesis, Université de Marne-la-Vallée.

Plexousakis, D. (1993). *Integrity constraint and rule maintenance in temporal deductive knowledge bases.* Proceedings of the 19th International Conference on Very Large Data Bases, Ireland, 146–157.

Pokorný, J. (2002). *XML data warehouse: Modelling and querying.* Proceedings of the 5th International Baltic Conference (BalticDB&IS 02), Tallinn, Estonia, 267–280.

Polaillon, G. (1998). *Organisation et interprétation par les treilles de Gallois de données de type multivalué, intervalle ou histogramme.* Thèse Docteur in Informatique. I'Université Paris IX-Dauphine.

Polyzotis, N., & Garofalakis, M. (2002a). Statistical synopses for graph-structured XML databases. *Proceedings of the ACM SIGMOD International Conference on Management of Data* (pp. 358-369).

Polyzotis, N., & Garofalakis, M. (2002b). Structure and value synopses for XML data graphs. *Proceedings of the International Conference on Very Large Data Bases* (pp. 466-477).

Pomerlani, W. J., & Blaha, M. R. (1993). An approach for reverse engineering of relational databases. *WCRE.*

Ponniah, P. (2003). *Database Design and Development: An Essential Guide for IT Professional.* Indianapolis, IN: Wiley-IEEE Press.

Ponzetto, S. P., & Strube, M. (2006). Exploiting semantic role labeling, WordNet and Wikipedia for coreference resolution. In *Proc. of HLT-NAACL, Human Language Technology Conf. of the NAACL* (pp. 192–199), New York, USA.

Poosala, V. (1997). *Histogram-based estimation techniques in database systems.* Unpublished doctoral dissertation, University of Wisconsin, WI.

Poosala, V., & Ganti, V. (1999). Fast approximate answers to aggregate queries on a data cube. *Proceedings of the 11th IEEE International Conference on Statistical and Scientific Database Management* (pp. 24-33).

Poosala, V., & Ganti, V. (1999). Fast Approximate Answers to Aggregate Queries on a Data Cube. *Proc. of the 11st IEEE Int. Conf. on Statistical and Scientific Database Management,* (pp. 24-33).

Poosala, V., & Ioannidis, Y. E. (1997). Selectivity estimation without the attribute value independence assumption.

Proceedings of the 23rd International Conference on Very Large Databases (pp. 486-495).

Poosala, V., & Ioannidis, Y.E. (1997). Selectivity Estimation without the Attribute Value Independence Assumption. *Proc. of the 23rd Int. Conf. on Very Large Databases,* (pp. 486-495).

Poosala, V., Ioannidis, Y. E., Haas, P. J., & Shekita, E. (1996). Improved histograms for selectivity estimation of range predicates. *Proceedings of the 1996 ACM International Conference on Management of Data* (pp. 294-305).

Popa, L., Velegrakis, Y., Miller, R.J., Hernandez, M.A., & Fagin, R. (2002). *Translating Web data*. Proceedings of the International Conference on Very Large Data Bases (VLDB), 598–609.

Porter, M. (2001). Strategy and the internet. *Harvard Business Review* (March 2001): (pp. 63-78).

Pourabbas, E., & Rafanelli, M. (2003). Hierarchies. In M. Rafanelli. (Ed.), *Multidimensional databases: Problems and solutions* (pp. 91–115). Hershey, PA: Idea Group Publishing.

Powley, W., Martin, P., Ogeer, N., & Tian, W. (2005). Autonomic buffer pool configuration in PostgreSQL. IEEE *International Conference on Systems, Man and Cybernetics*, 53-58.

Poworotznek, E. (2003). Linking of errata: current practices in online physical sciences journals. *Journal of the American Society for Information Science and Technology, 54*(12), 1153-1159.

Pradhan, S. (2003). Connecting databases with argumentation. *Web Knowledge Management and Decision Support, 14th International Conference on Applications of Prolog, (INAP 2001), Lecture Notes in Computer Science, 2543* (pp.170-185).

Preece, A., Hui, K., Gray, A., Jones, D., & Cui, Z. (1999). The KRAFT architecture for knowledge fusion and transformation. *In 19th SGES International Conference on Knowledge-based Systems and Applied Artificial Intelligence (ES'99).* Berlin: Springer.

President's Information Technology Advisory Committee (PITAC). (2004). *Revolutionizing health care through information technology.* Author.

Procopiuc, C. M., Jones, M., Agarwal, P. K., & Murali, T. M. (2002). *A Monte Carlo algorithm for fast projective clustering.* Paper presented at SIGMOD'02: the 2002 ACM SIGMOD international conference on Management of data, New York, NY, USA.

Prodromidis, A.L., Chan, P., & Stolfo, S.J. (2000). Meta-learning in distributed data mining systems: Issues and approaches. In H. Kargupta, & P. Chan (Eds.), *Advances of distributed data mining.* AAAI Press.

Provost, F.J., & Kolluri, V. (1999). A survey of methods for scaling up inductive algorithms. *Data Mining and Knowledge Discovery, 3*(2), 131–169.

Qian, X. (1988). An effective method for integrity constraint simplification. *ICDE '88* (pp. 338-345).

Qian, X. (1989). *Distribution design of integrity constraints.* Proceedings of the 2nd International Conference on Expert Database Systems, 205–226.

Qian, X. (1990). Synthesizing database transaction. *Proceedings of the 16th International Conference on Very Large Data Bases,* Brisbane, (pp. 552-565).

Qin, B., & Liu, Y. (2003). High Performance Distributed Real-time Commit Protocol. Journal of Systems and Software, Elsevier Science Inc., 68(2), 145-152.

Quinlan, J. R. (1983). Learning efficient classification procedures and their application to chess end-games. In *Machine learning: An artificial intelligence approach* (pp. 463-482).

Qun, C., Lim, A., & Ong, K.W. (2003). *D(k)-index: An adaptive structural summary for graph-structured data.* Proceedings of the 2003 ACM SIGMOD International Conference on Management of Data, San Diego, California, 134-144.

Raatikka, V., & Hyvonen, E. (2002). *Ontology-based semantic metadata validation.*

Rabus, B., Eineder, M., Roth, A., & Bamler, R. (2003). The shuttle radar topography mission – a new class of digital elevation models acquired by spaceborne radar. *Photogrammetry & Remote Sensing, 57,* 241-262.

Radev D., Qi, J., Otterbacher, H., & Tam (2003). *Mead reducs: Michigan at DUC 2003.* Edmonton, Alberta, Canada: ACL, (pp. 160-167).

Raghavan, S., & Garcia-Molina, H. (2001). Crawling the hidden Web. *27th International Conference on Very Large Data Bases (VLDB'01),* 129-138.

Rahm, E., & Bernstein, A. (2001). A survey of approaches to automatic schema matching. *VLDB Journal 4*(10), 334-350. New York: Springer-Verlag Inc.

Rahm, E., & Bernstein, P. A. (2006). An online bibliography on schema evolution. *Sigmod Record, 35*(4), 30-31.

Raju, K. V. S. V. N., & Majumdar, A. K. (1988). Fuzzy functional dependencies and lossless join decomposition of fuzzy relational database systems. *ACM Transactions on Database Systems, 13*(2), 129-166

Ram, S., & Park, J. (2004). Semantic Conflict Resolution Ontology (SCROL): An Ontology for Detecting and Resolving Data and Schema-Level Semantic. *IEEE Transaction on Knowledge and Data Engineering, 16*(2), 189-202.

Ramamritham, K. (1993). Real-time Databases. Distributed and Parallel Databases. Special Issue: Research Topics in Distributed and Parallel Databases, 1(2), 199-226.

Ramanath, M., Freire, J., Haritsa, J., & Roy, P. (2003). Searching for Efficient XML-to-Relational Mappings. In *XSym'03: Proceedings of the 1st International XML Database Symposium, 2824,* 19–36, Berlin, Germany. Springer-Verlag.

Ramaswamy, S., Rastogi, R., & Shim, K. (2000). Efficient algorithms for mining outliers from large data sets. *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data (SIGMOD'00)* (pp. 427-438).

Ramez, E., & Shamkant, N. (2002). *Fundamentos de sistemas de bases de datos.* Addison Wesley.

Rao, J., Zhang, C., Megiddo, N., & Lohman, G. (2002). Automating Physical Database Design in a Parallel Database. *Proceedings of the ACM SIGMOD International Conference on Management of Data,* (pp. 558-569).

Raptopoulou, K., Vassilakopoulos, M., & Manolopoulos, Y. (2004). Towards quadtree-based moving objects databases. To appear in *Proc. of the Eighth East-European Conference on Advances in Databases and Information Systems (ADBIS'2004).* Budapest, Hungary.

Rashid, A., & Sawyer, P. (2000). Object Database Evolution Using Separation of Concerns. *SIGMOD Record, 29*(4): 26-33.

Rational 2000e Using Rose Oracle8. (2000). Santa Clara, CA: Rational Software Corporation.

Ratnasamy, P., Francis, P., Handley, M., Karp, R., & Shenker, S. (2001). A scalable content-addressable network. *Proceedings of the 2001 ACM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications* (pp. 161-172).

Ratnasamy, S., Karp, B., Shenker, S., Estrin, D., Govindan R., Yin, L., & Yu, F. (2003). Data-Centric Storage in Sensornets with GHT, a Geographic Hash Table. *Mobile Networks and Applications, 8*(4), 427-442.

Ravat, F., Teste, O., & Zurfluh, G. (2006). *A multiversion-based multidimensional model.* Proceedings of the 8th International Conference on Data Warehousing and Knowledge Discovery (DaWaK06), Krakow, Poland, 4081, 65–74.

Ray, I., & Xin, T. (2006). Analysis of dependencies in advanced transaction models, *Distributed and Parallel Databases, 20*(1), 5-27.

Rechy-Ramírez, E.-J., & Benítez-Guerrero, E. (2006). *A model and language for bitemporal schema versioning in data warehouses.* Proceedings of the 15th International Conference on Computing (CIC'06), Mexico City, Mexico.

Reiter R.(1987). On closed world data bases. Readings in nonmonotonic reasoning, pages 300–310. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

Reiter, R. (1984). A logical reconstruction of relational database theory. In J. L. Mulopoulos & J. W. Schmidt (Eds.), *On conceptual modelling* (pp. 163-189). New York: Springer.

Relocating the Back Office. (2003, 11 Dec.). *The Economist,* Retrieved December 20, 2003 from, http://www.economist.com/displaystory.cfm?story_id=2282381

Ren, Q., & Dunham, M. H. (1999). Using clustering for effective management of a semantic cache in mobile computing. In S. Banerjee, P. K. Chrysanthis, & E. Pitoura, (Eds.), *Proceedings of the 1st ACM International Workshop on Data Engineering for Wireless and Mobile Access*, (pp. 94–101), New York, NY, USA. ACM Press.

Ren, Q., & Dunham, M. H. (2003). Semantic Caching and Query Processing. *Transactions on Knowledge and Data Engineering, 15*(1), 192–210.

Rennolls, K. (2005). An Intelligent Framework (O-SS-E) For Data Mining, Knowledge Discovery and Business Intelligence. *In Proceeding 2nd International Workshop on Philosophies and Methodologies for Knowledge Discovery, PMKD'05,* (pp. 715-719).

Rhea, S., Wells, C., Eaton, P., Geels, D., Zhao, B., Weatherspoon, H., et al. (2001). Maintenance-free global data storage. *IEEE Internet Computing, 5*(5), 40-49.

Richter, T. (1997). A new algorithm for the ordered tree inclusion problem. In *Proceedings* of the 8th Annual Symposium on Combinatorial Pattern Matching (CPM), in Lecture *Notes of Computer Science (LNCS), volume 1264,* 150-166.

Rigaux, P., Scholl, M., & Voisard, A. (2001). *Spatial databases—With applications to GIS*. San Francisco: Morgan Kaufmann.

Risch, T. (2004). Mediators for querying heterogeneous data. In M.P. Singh (Ed.), *The practical handbook of Internet computing.* Chapman & Hall/CRC.

Rivest, S., Bédard, Y., & Marchand, P. (2001). Toward better support for spatial decision making: Defining the characteristics of spatial on-line analytical processing (SOLAP). *Geomatica, 55*(4), 539–555.

Rivest, S., Bédard, Y., Proulx, M.J., Nadeau, M., Hubert, F., & Pastor, J. (2005). SOLAP technology: Merging business intelligence with geospatial technology for interactive spatio-temporal exploration and analysis of data. *ISPRS Journal of Photogrammetry & Remote Sensing, 60,* 17–33.

Rizvi, S. J., & Haritsa, J. R. (2002). *Maintaining data privacy in association rule mining.* Proceedings of the 28th International Conference on Very Large Data Bases (VLDB'02), Hong Kong, China.

Rizzi, S. (2007). Conceptual modeling solutions for the data warehouse. In R. Wrembel, & C. Koncilia (Eds.), *Data warehouses and OLAP: Concepts, architectures and solutions* (pp. 1–26). Hershey, PA: Idea Group Publishing.

Rizzi, S., & Golfarelli, M. (2006). *What time is it in the data warehouse?* Proceedings of the 8th International Conference on Data Warehousing and Knowledge Discovery (DaWaK 06), Krakow, Poland, 4081, 134–144.

Rizzi, S., & Saltarelli, E. (2003). View materialization vs. indexing: Balancing space constraints in data warehouse design. In *Lecture notes in computer science: Vol. 2681. 15th International Conference on Advanced Information Systems Engineering (CAiSE 2003)* (pp. 502-519). Berlin, Germany: Springer.

Rob, P., & Coronel, C. (2002). *Database systems: Design, implementation, and management.* Boston: Thomas Learning.

Robertson, J. (2003). *So, what is a content management system?* Retrieved 14th March, 2008, from http://www.steptwo.com.au/papers/kmc_what/index.html

Rockley, A., Kostur, P., & Manning, S. (2003). *Managing enterprise content: a unified content strategy.* Indianapolis, IN: New Riders.

Rockwell, S. R., & McCarthy, W. E. (1999). REACH: automated database design integrating first-order theories, reconstructive expertise, and implementation heuristics for accounting information systems. International Journal of Intelligent Systems in Accounting, Finance & Management, 8(3), 181-197.

Roddick, J. F. (1992). Schema evolution in database systems: An annotated bibliography. *ACM SIGMOD Record, 21*(4), 35-40.

Roddick, J. F. (1995). A Survey of Schema Versioning Issues for Database Systems, *Information and Software Technology, 37*(7), 383-393.

Roddick, J. F. et. al. (2000). Evolution and Change in Data Management- Issues and Directions. *ACM SIGMOD Record, 29*(1), 21-25.

Roddick, J. F., Craske, N.G., & Richards, T. J. (1993). A taxonomy for schema versioning based on the relational and entity relationship models. *Proceedings of the 12th International Conference on the Entity-Relationship Approach,* 137-148.

Rodrıguez, M. A., & Egenhofer, M. J. (2003). Determining semantic similarity among entity classes from different ontologies. *IEEE Trans. Knowl. Data Eng., 15*(2), 442-456.

Rodríguez, M.A., & Egenhofer, M.J. (2004). Comparing geospatial entity classes: An asymmetric and context-dependent similarity measure. *International Journal of Geographical Information Science, 18*(3), 229–256.

Rodriguez-Martinez, M., & Rossopoulos, N. (2000). MOCHA: A Self-Extensible Database Middleware System

for Distributed Data Sources. *Proc. of the 2000 ACM Int. Conf. on Management of Data,* (pp. 213-224).

Rogers, S., Girolami, M., Campbell, C., & Breitling, R. (2005). The latent process decomposition of cDAN microarray data sets. *IEEE/ACM Transactions on Computational Biology and Bioinformatics, 2*(2), 143-156.

Ross, K. A., & Stoyanovich, J. (2004). *Symmetric relations and cardinality-bounded multisets in database systems.* Proceedings of the 20th International Conference on Very Large Databases, Toronto, Ontario, Canada.

Roth, M.T., & Schwarz, P. (1997). *Don't scrap it, wrap it! A wrapper architecture for legacy data sources.* Proceedings of the International Conference on Very Large DataBases, 266–275.

Roussopoulos, N., Kelley, S., & Vincent, F. (1995). Nearest Neighbor Queries. *In Proceedings of the 1995 ACM SIGMOD International Conference on Management of Data (SIGMOD 1995),* San Jose, California, USA, 22-25 May, 1995, (pp. 71-79). ACM Press.

Roussos G., Papadogkonas D., Taylor J., Airantzis D., Levene M., & Zoumboulakis M. (2007). Shared Memories: A Trail-based Coordination Server for Robot Teams. *First International Conference on Robot Communication and Coordination (IEEE Robocomm),* 15-17 October, Athens, Greece.

Rumelhart, D. E., & Zipser, D. (1985). Feature discovery by competitive learning. *Cognitive Science, 9,* 75–112.

Rupnik, R., Krisper, M., & Bajec, M. (2004). A new application model for mobile technologies. *International Journal of Information Technology and Management, 3*(2/3/4), 282-291.

Ruppenhofer, J., Ellsworth, M., Petruck, M. R. L., Johnson, Ch. R., & Scheffczyk, J. (2006). *FrameNet II: Extended Theory and Practice.* International Computer Science Institute, Berkeley, USA.

Sabanis, N., & Stevenson, N. (1992). Tools and techniques for data remodeling COBOL applications. *Proceedings 5th International Conference on Software Engineering and Applications.*

Sabin, D., & Weigel, R. (1998). Product Configuration Framework – A survey. *IEEE Intelligente Systems, 13*(4), 42-49.

Sacks-Davis, R., Kent, A., Ramamohanarao, K., Thom, J. and Zobel, J. (1995). Atlas: A nested relational database system for text application. *IEEE Trans. Knowledge and Data Engineering,* 7(3), 454-470.

Sadagopan, N., Krishnamachari, A., & Helmy, A. (2004). *Active Query Forwarding in Sensor Networks.* Ad-Hoc Networks, in press.

Sadri, F., & Kowalski, R. (1988). A theorem-proving approach to database integrity. In J. Minker (Ed.), *Foundations of deductive databases and logic programming* (pp. 313-362). Morgan Kaufmann.

Sakama, C. & Inoue, K. (2000). Prioritized logic programming and its application to commonsense reasoning. *AI, 123*(1-2), 185-222.

Sakama, C. (2005). Ordering default theories and nonmonotonic logic programs. *Theoretical Computer Science, 339,* 127-152.

Sakurai, S. et al. (2008a). Discovery of Time Series Event Patterns based on Time Constraints from Textual Data. *International Journal of Computational Intelligence, 4*(2), 144-151.

Sakurai, S. et al. (2008b). A Sequential Pattern Mining Method based on Sequential Interestingness. *International Journal of Computational Intelligence, 4*(4), 252-260.

Sakurai, S. et al. (2008c). Discovery of Sequential Patterns Coinciding with Analysts' Interests. *Journal of Computers, 3*(7), 1-8.

Salo T., & Hill, J. (2000). Mapping objects to relational databases. *Journal of Object Oriented Programming, 13*(1).

Saltenis, S., Jensen, C. S., Leutenegger, S. T., & Lopez, M. A. (2000). Indexing the positions of continuously moving objects. *Proc. of the ACM SIGMOD International Conference on Management of Data (SIGMOD'2000),* 331-342. Dallas, TX.

Salton, G., Wang, A., & Yang, C. S. (1975). A vector space model for information retrieval. *Journal of American Society for Information Science, 18*(11), 613-620.

Saltzberg, B., & Tsotras, V. J. (1999). Comparison of access methods for time-evolving data. *ACM Computing Surveys, 31*(2), 158-221.

Samarati, P., & Sweeney, L. (1998). *Protecting privacy when disclosing information: k-anonymity and its enforcement through generalization and suppression*. Proceedings of the IEEE Symposium on Research in Security and Privacy, Oakland, CA.

Sampaio, M. C., Dias, P. M., & Baptista, C. S. (2003). Incremental Updates on Mobile Data Warehousing Using Optimized Hierarchical Views and New Aggregation Operators. *Proc. of 17th Int. Conf. on Advanced Information Networking and Applications,* (pp. 78-83).

Sanjay, A., Narasayya, V. R., & Yang, B. (2004). Integrating vertical and horizontal partitioning into automated physical database design. *Proceedings of the ACM International Conference on Management of Data (SIGMOD),* (pp. 359–370).

Sankaranarayanan, J., Alborzi, H., & Samet, H. (2005). *Efficient query processing on spatial networks*. Proceedings of the ACM-GIS 2005, 200–209.

Santini, S., & Jain, R. (1999). Similarity measures. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 21*(9), 871-883.

SAP. (2000). *Multi-dimensional modelling with BW: ASAP for BW accelerator* [technical report]. SAP America Inc. and SAP AG.

Sapia, C., Blaschka, M., Höfling, G., & Dinter, B. (1998). *Extending the E/R model for multidimensional paradigms*. Proceedings of the 17th International Conference on Conceptual Modeling, 105–116.

Sarda, N. L. (1987). Extensions to SQL for historical databases. *IEEE Transactions on Systems, 12*(2), 247-298.

Sarma A.D and Benjelloun O. and Halevy A. and Widom J. (2006). Working Models for Uncertain Data, Proceedings of the 22nd International Conference on Data Engineering ICDE'06, Washington, DC, USA.

Sartiani, C., Manghi, P., Ghelli, G., & Conforti, G. (2004). XPeer: A self-organizing XML P2P database system. *Proceedings of the 2004 International Workshop on Peer-to-Peer Computing and Databases, in conjunction with the 9th International Conference on Extending Database Technology* (pp. 456-465).

Satoshi, H. (1999). The contents of interviews with the project manager of FDWH in IBM Japan. Proceedings of the 1999 SMAP Workshop, San Diego, CA.

Sattler, K., Geist, I., & Schallehn, E. (2003). QUIET: Continuos query-driven index tuning. *In Proceedings Of the 20th VLDB Conference,* 1129-1132. Berlim, Germany.

Sattler, K., Geist, I., & Schallehn, E. (2005). Towards index schemes for self-tuning DBMS. *In Proceedings of the 21st International Conference on Data Engineering Workshops,* 1216-1223.

Satyanarayanan, M. (2001). Pervasive Computing: Vision and Challenges. *IEEE Personal Communications, 8*(4), 10-17.

Satyanarayanan, M., Kistler, J. J., Kumar, P., Okasaki, M. E., Siegel, E. H., & Steere, D. C. (1990). Coda: A Highly Available File System for a Distributed Workstation Environment. *IEEE Transactions on Computers, 39*(4), 447–459.

Savinov, A. (2004). *Principles of the concept-oriented data model*. Moldavian Academy of Sciences, Institute of Mathematics and Informatics.

Savinov, A. (2005a). Concept as a generalization of class and principles of the concept-oriented programming. *Computer Science Journal of Moldova, 13*(3), 292-335.

Savinov, A. (2005b). Hierarchical multidimensional modelling in the concept-oriented data model. *3rd International Conference on Concept Lattices and their Applications (CLA'05)* (pp. 123-134).

Savinov, A. (2005c). Logical navigation in the concept-oriented data model. *Journal of Conceptual Modeling, 36*. Retrieved from http://www.inconcept.com/jcm

Savinov, A. (2006a). Grouping and aggregation in the concept-oriented data model. *21st Annual ACM Symposium on Applied Computing (SAC'06)* (pp. 482-486).

Savinov, A. (2006b). Query by constraint propagation in the concept-oriented data model. *Computer Science Journal of Moldova, 14*(2), 219-238.

Scartascini, C. G., & Oliveira, M. (2003). *Political Institutions, Policymaking Processes and Policy Outcomes: A Guide to Theoretical Modules and Possible Empirics.* Design Paper, Inter-American Development Bank, American Research Network.

Schaffer, T. (2001). Databases and Political Science Research. *Online Information Review, 25*(1), 47-53.

Scheena, M. (2003). *Microarray analysis.* NJ: Wiley.

Scheer, A. W. (1998). Business Process Engineering: Reference Models for Industrial Enterprises. Berlin: Springer-Verlag.

Schiefer, J., List, B., & Bruckner, R. (2002). A holistic approach for managing requirements of data warehouse systems. *Proceedings of the 8th Americas' Conference on Information Systems,* 77-87.

Schmidt, A. R., Waas, F., Kersten, M. L., Florescu, D., Manolescu, I., Carey, M. J., & Busse, R. (2001). The XML benchmark project, Technical Report INS-Ro1o3. *Centrum voor Wiskunde en Informatica.*

Schmidt, A., Waas, F., Kersten, M., Carey, M., Manolescu, I., & Busse, R. (2002). XMark: A benchmark for XML data management. *Proceedings of the 28th International Conference on Very Large Data Bases* (pp. 974-985).

Schmidt, R., & Shahabi, C. (2002a). How to evaluate multiple range-sum queries progressively. *21st ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems (PODS), 2002,* 133-141.

Schmidt, R., & Shahabi, C. (2002b). Propolyne: A fast wavelet-based algorithm for progressive evaluation of polynomial range-sum queries. *Proceedings of the Conference on Extending Database Technology (EDBT), 2002,* 664-681.

Schuster, H., & Heinl, P. (1997). A workflow data distribution strategy for scalable workflow management systems. *ACM Symposium on Applied Computing,* (pp. 174-176).

Schut, P. (Ed.). (2007). OpenGIS Web processing service specification, version 1.0.0. Open Geospatial Consortium. Retrieved from http://portal.opengeospatial.org/files/?artifact_id=21988&version=1

Schwering, A., & Raubal, M. (2005). Spatial relations for semantic similarity measurement. In J. Akoka, S. W. Liddle, I.-Y. Song, M. Bertolotto, I. Comyn-Wattiau, S. S.-S. Cherfi, W.-J. van den Heuvel, B. Thalheim, M. Kolp, P. Bresciani, J. Trujillo, C. Kop, & H. C. Mayr (Eds.), *Lecture notes in computer science* (Vol. 3770, pp. 259-269). Springer.

Schwering, A., & Raubal, M. (2005). *Spatial relations for semantic similarity measurement.* Proceedings of the *ER'05* Perspectives in Conceptual Modeling, 3770, 259–269.

Sebastiani, F. (2002). Machine learning in automated text categorization. *ACM Computing Surveys, 34*(1), 1–47.

Sellis et al. (Eds.) (2003). Access methods and query processing techniques. *Chapter in the book Spatiotemporal Databases: The ChoroChronos Approach, LNCS* 2520. Springer Verlag.

Seo, C., Lee, S., & Kim, H. (2003). An efficient index technique for XML documents using RDBMS. *Information and Software Technology, 45*(2003), 11-22. B.V.: Elsevier Science.

Seo, J., & Schneiderman, B. (2002). Interactively exploring hierarchical clustering results. *IEEE Computer, 35*(7), 80-86.

Seo, J., & Schneiderman, B. (2005). A rank-by-feature framework for interactive exploration of multidimensional data. *Information Visualization, 4*(2), 99-113.

Serna-Encinas, M-T, & Adiba, M. (2005). *Exploiting bitemporal schema versions for managing an historical medical data warehouse: A case study.* Proceedings of the ENC'05, 88–95.

Shadbolt, N., Hall, W., Berners-Lee, T. (2006) The Semantic Web Revisited. *IEEE Intelligent Systems*, Vol. 21, Issue 3, May/June 2006

Shahabi, C., Kolahdouzan, M., & Sharifzadeh, M. (2002). A Road Network Embedding Technique for K-Nearest Neighbor Search in Moving Object Databases. *In Proceedings of the 1996 ACM Symposium on Advances in Geographic Information Systems (ACM GIS 1996),* McLean, VA (near Washington, DC), USA, 8-9 November, 2002, (pp. 94-100), ACM Press.

Shankaranarayanan, G., & Even, A. (2006). The metadata enigma. *Communications of the ACM, 49*(2), 88-94.

Shanker, U., Misra, M., & Sarje, A. K. (2001). Hard real-time distributed database systems: Future directions. *All India Seminar on Recent Trends in Computer Communication Networks, Department of Electronics & Computer Engineering,* 172-177. *Indian Institute of Technology Roorkee, India.*

Shanker, U., Misra, M., & Sarje, A. K. (2005a). A Memory Efficient Fast Distributed Real Time Commit Protocol. 7th International Workshop on Distributed Computing (IWDC 2005), Indian Institute of Technology Kharagpur, India, 500-505.

Shanker, U., Misra, M., & Sarje, A. K. (2005a). Dependency sensitive distributed commit protocol. *8th International Conference on Information Technology (CIT05),* 41-46, *Bhubaneswar, India.*

Shanker, U., Misra, M., & Sarje, A. K. (2005b). Optimizing distributed real-time transaction processing during execution phase. *3rd International Conference on Computer Application (ICCA2005), University of Computer Studies,* 1-7. *Yangon, Myanmar.*

Shanker, U., Misra, M., & Sarje, A. K. (2005b). The MEWS Distributed Real Time Commit Protocol. WSEAS Transactions on COMPUTERS, 4(7), 777-786.

Shanker, U., Misra, M., & Sarje, A. K. (2006a). Some Performance Issues in Distributed Real Time Database Systems. VLDB PhD Workshop (2006), The Convention and Exhibition Center (COEX), Seoul, Korea.

Shanker, U., Misra, M., & Sarje, A. K. (2006a). SWIFT-A new real time commit protocol. *International Journal of Distributed and Parallel Databases, 20*(1), 29-56. Springer Verlag

Shanker, U., Misra, M., & Sarje, A. K. (2006b). OCP-the optimistic commit protocol. *17th Australasian Database Conference (ADC2006), 49,* 193-202. *Hobart, Tasmania, Australia.*

Shanker, U., Misra, M., & Sarje, A. K. (2006d). Some performance issues in distributed real time database systems. *VLDB PhD Workshop (2006), The Convention and Exhibition Center (COEX).* Seoul, Korea.

Shanker, U., Misra, M., & Sarje, A. K. (2007). Distributed real time database systems: Background and literature review. *International Journal of Distributed and Parallel Databases. Springer Verlag* (accepted).

Shanker, U., Misra, M., Sarje, A. K. & Shisondia, R. (2006c). Dependency sensitive shadow SWIFT. *10th International Database Applications and Engineering Symposium (IDEAS 06),* 373-276. Delhi, India.

Shanks, G., Tansley, E., & Weber, R. (2003). Using ontology to validate conceptual models. *Communication of ACM, 46*(10), 85-89.

Shanmugasundaram, J., Shekita, E., Barr, R., Carey, M., Lindsay, B., Pirahesh, H., & Reinwald, B. (2001). Efficiently publishing relational data as XML document. *VLDB Journal 19*(2-3), 133-154.

Shanmugasundaram, J., Tufte, K., Zhang, C., He, G., DeWitt, D. J., & Naughton, J. F. (1999). Relational Databases for Querying XML Documents: Limitations and Opportunities. In *VLDB'99: Proceedings of 25th International Conference on Very Large Data Bases,* (pp. 302-314), San Francisco, CA, USA. Morgan Kaufmann Publishers.

Shannon, C. E. (1949). Communication theory of secrecy systems. *Bell System Technical Journal, 28*(4), 656-715.

Sharaf, M. A., & Chrysanthis, P. K. (2002). Semantic-based Delivery of OLAP Summary Tables in Wireless Environments. *Proc. of the 2002 ACM Int. Conf. on Information and Knowledge Management,* (pp. 84-92).

Sharifzadeh, M. and Shahabi, C. (2006) The Spatial Skyline Queries. *In Proceedings of the Very Large Data Bases Conference (VLDB 2006),* Seoul, Korea, 12-15 September, 2006, (pp. 751-762). Morgan Kaufmann, San Francisco.

Sheikholeslami, G., Chatterjee, S., & Zhang, A. (1998). *WaveCluster: A Multi-Resolution Clustering Approach for Very Large Spatial Databases.* Paper presented at VLDB '98: the 24th International Conference on Very Large Data Bases, San Francisco, CA, USA.

Sheikholeslami, G., Chatterjee, S., & Zhang, A. (1999). WaveCluster: A wavelet based clustering approach for spatial data in very large database. *VLDB Journal, 8*(3-4), 289-304.

Shekhar, S., & Chawla, S. (2003). *Spatial databases: A tour.* Prentice Hall.

Shekhar, S., & Liu, D. R. (1997). CCAM: A connectivity-clustered access method for networks and network computations. *IEEE Transactions on Knowledge and Data Engineering, 9*(1), 102-119.

Shekhar, S., Chawla, S., Ravada, S., Fetterer, A., Liu, X., & Lu, C. T. (1999). Spatial Databases - Accomplishments and Research Needs. *IEEE Transactions Knowledge Data Engineering, 11*(1), 45-55.

Shekhar, S., Kohli, A., & Coyle, M. (1993). Path computation algorithms for advanced traveller information systems. *Proceedings of 9th International Conference on Data Engineering* (pp. 31-39).

Shestakov, D., & Salakoski, T. (2007). On estimating the scale of national deep Web. *18th International Conference on Database and Expert Systems Applications (DEXA'07),* 780-789.

Shestakov, D., Bhowmick, S., & Lim, E.-P. (2005). DEQUE: querying the deep Web. *Data&Knowledge Engineering, 52*(3), 273-311.

Sheth, A. P., & Larson, J. A. (1990). Federated database systems for managing distributed, heterogeneous and autonomous databases. *ACM Computing Surveys, 22*(3), 183-236.

Shipley, S. T., Graffman, I. A., & Ingram, J. K. (2000, June). GIS applications in climate and meteorology. *Proceedings of the ESRI User Conference*, San Diego, CA.

Shipman, D. W. (1981). The functional data model and the data language DAPLEX. *ACM Transactions on Database Systems, 6*(1), 140-173.

Shou, Y., Mamoulis, N., Cao, H., Papadias, D., & Cheung, D. W. (2003). Evaluation of Iceberg Distance Joins. *In Proceedings of the 8th International Symposium on Spatial and Temporal Databases (SSTD 2003)*, Santorini Island, Greece, 24-27 July, 2003, Lecture Notes in Computer Science 2750, (pp. 270-288), Springer.

Shukla, A., Deshpande, P., & Naughton, J. F. (2000). Materialized view selection for multi-cube data models. In *Lecture notes in computer science: Vol. 1777. 7th International Conference on Extending Database Technology (EDBT 2000)* (pp. 269-284). Berlin, Germany: Springer.

Shvaiko, P., & Euzenat, J. (2005). A survey of schema-based matching approaches. *Journal of Data Semantics, IV,* 146-171.

Siau, K. (2004). *Advanced Topics in Database Research, 3.* Hershey, PA: Idea Group Publishing.

Siau, K., & Shen, Z. (2003). Mobile communications and mobile services. *International Journal of Mobile Communications, 1*(1/2), 3-14.

Siau, K., Lim, E., & Shen, Z. (2001). Mobile commerce: Promises, challenges, and research agenda. *Journal of Database Management, 12*(3), 4-13.

Sibanda, T., & Uzuner, Ö. (2006). Role of local context in automatic deidentification of ungrammatical, fragmented text. In *Proc. of HLT-NAACL, Human Language Technology Conf. of the NAACL* (pp. 65–73), New York, USA.

SICStus prolog. (n.d.). Retrieved from http://www.sics.se/sicstus/

Sidhu, A., Dillon, T., & Chang, E. (2006). Advances in Protein Ontology Project. *In Proceedings of 19th IEEE International Symposium on Computer-Based Medical Systems (CBMS 2006).*

Silberschatz, A., & Tuzhilin, A. (1996). What Makes Patterns Interesting in Knowledge Discovery Systems. *IEEE Transactions on Knowledge and Data Engineering, 8*(6), 970-974.

Silva, L. P., & Pinheiro, F. A. C. (2003). Eliciting requirements for identifying workflow categories. *Proceedings of the VI Workshop on Requirements Engineering (WER'03)* (pp 16-31).

Silvestri, F., Orlando, S., & Perego, R. (2004). WINGS: A Parallel Indexer for Web Contents, *Lecture Notes in Computer Science, 3036,* 263-270.

Simon, E. and Valduriez, P. (1986). Integrity control in distributed database systems. *Proceedings of the 19th Hawaii International Conference on System Sciences,* Hawaii, (pp. 622-632).

Simon, E., & Valduriez, P. (1987). Design and analysis of a relational integrity subsystem. MCC Technical Report DB-015-87.

Singh, S. et al. (2003). Context-Based Data Mining Using Ontologies. In I. Song et al. (Eds.), *Proceedings 22nd International Conference on Conceptual Modeling*, Lecture Notes in Computer Science, *2813,* 405-418. Springer.

Sinha, A. P., & Vessey, I. (1999). An empirical investigation of entity-based and object-oriented data modeling. Proceedings of the Twentieth International Conference on Information Systems. P. De & J. DeGross (Eds.), Charlotte, North Carolina, (pp. 229-244).

Sintek, M., & Decker, S. (2002). TRIPLE – A query, inference, and transformation language for the Semantic Web. In *Proceedings of the First International Semantic Web Conference – ISWC 2002* (LNCS 2342). Heidelberg: Springer-Verlag.

Sismanis, Y., Deligiannakis, A., Roussopoulos, N., & Kotidis, Y. (2002). Dwarf: Shrinking the petacube. *ACM SIGMOD International Conference on Management of Data (SIGMOD 2002)* (pp. 464-475).

Sivasankaran, R.M., Ramamritham, K., Stankovic, J.A., & Towsley, D.F. (1995). *Data placement, logging and recovery in real-time active database*. Proceedings of the 1st International Workshop on Active and Real-time Database Systems, 226–241.

Skarra, A. H., & Zdonik, S. B. (1986). The Management of Changing Types in an Object-Oriented Database. *Conference on Object-Oriented Programming Systems, Languages, and Applications* (OOPSLA'86), Portland, Oregon, USA.

Smith, J. R., Li, C. S., & Jhingran, A. (2004). A wavelet framework for adapting data cube views for OLAP. *IEEE Transactions on Knowledge and Data Engineering, 16*(5), 552-565.

Smith, M. K., Welty, C., & McGuinness, D. L., (eds.) (2004). *OWL Web Ontology Language Guide – W3C Recommendation 10 February 2004.* W3C (http://www.w3.org/TR/owl-guide/).

Smith, R.K., Hale, J.E., & Parish, A.S. (2001). An empirical study using task assignment patterns to improve the accuracy of software effort estimation. *IEEE Transactions on Software Engineering, 27*(3), 264–271.

Smith, S. W., & Safford, D. (2000). *Practical private information retrieval with secure coprocessors* (Research Rep. No. RC-21806). IBM T. J. Watson Research Center.

Smits, P. (Ed.). (2002). INSPIRE architecture and standards position paper. Architecture and Standards Working Group. Retrieved from http://inspire.jrc.it/documents/inspire_ast_pp_v4_3_en.pdf

Smolkin, M., & Ghosh, D. (2003). Cluster Stability Scores for Microarray Data in Cancer Studies. *BMC Bioinformatics, 4*, 36.

Snodgrass, R. T. (1987). The temporal query language Tquel,. *ACM Transactions on Database Systems, 12*(2), 247-298.

Snodgrass, R. T. (1995). *The TSQL2 temporal query language*. Kluwer 1995.

Snodgrass, R. T. (1999). Developing time oriented applications in SQL. Morgan Kaufmann.

SOAP (2007). Simple Object Access Protocol (SOAP) 1.2. www.w3.org/TR/soap12-part1/ and www.w3.org/TR/soap12-part2/

Son, S.H., & Agrawala, A.K. (1989). Distributed checkpointing for globally consistent states of databases. *IEEE Transactions on Software Engineering, 15*(10), 1157–1167.

Son, S.H., & Kouloumbis, S. (1993). A token-based synchronization scheme for distributed real-time databases. *Information Systems, 18*(6), 375–389.

Sondheim, M., et al. (1999). GIS interoperability. In P. Logley, M. Goodchild, D. Maguire, & D. Rhind (Eds.), *Geographical information systems principles and technical issues.* New York: John Wiley & Sons.

Sørensen, C.F. (2005). Adaptive mobile work processes in context-rich, heterogeneous environments [doctoral thesis]. Norwegian University for Science and Technology.

Sotnykova, A., Vangenot, C., Cullot, N., Bennacer, N., & Aufaure, M. (2005). Semantic mappings in description logics for spatio-temporal database schemas integration. *Journal of Data Semantic, 3,* pp. 143-167.

Spaccapietra, S., Cullot, N., Parent, C., & Vangenot, C. (2004). *On spatial ontologies.* Proceedings of the VI Brazilian Symposium on Geoinformatica (GEOINFO 2004), Campos do Jordão, Brazil.

Speičys, L., Jensen, C. S., & Kligys, A. (2003). Computational data modeling for network constrained moving objects. *Proceedings of 11th ACM International Symposium on Advances in Geographic Information Systems* (pp. 118-125).

Spyns, P., Meersman, R., & Jarrar, M. (2002). Data modeling versus ontology engineering. *SIGMOD Record Special Issue on Semantic Web, Database Management and Information Systems, 31.*

Srikant, R., & Agrawal, R. (1996). Mining Sequential Patterns: Generalizations and Performance Improvements. *Proceedings of the 5th International Conference Extending Database Technology,* France, (pp. 3-17).

Sripanidkulchai, K., Maggs, B., & Zhang, H. (2003). Efficient content location using interest-based locality in peer-to-peer systems. *Proceedings of the 22nd Annual Joint Conference of the IEEE Computer and Communications Societies* (pp. 81-87).

Srivastava, A., Han, E., Kumar, V., & Singh, V. (1999). Parallel formulations of decision-tree classification

algorithms. *Data Mining and Knowledge Discovery, 3*(3), 237–261.

Srivastava, D., Al-Khalifa, S., Jagadish, H.V., Koudas, N., Patel, J.M., & Wu, Y. (2002). *Structural joins: A primitive for efficient XML query pattern matching.* Proceedings of the 18th International Conference on Data Engineering (ICDE 02), San Jose, California, 141.

Sschnaitter, K., Abiteboul, S., Milo, T., & Polyzotis, N. (2006). COLT: Continuous on-line tuning. *In Proceedings of the 2006 ACM SIGMOD International Conference on Management of data*, 793-795).

St. Amant, K. (2003). Designing web sites for international audiences, *Intercom*, 15-18. The flies swarm in. (2000). *The Economists.* Retrieved April 3, 2008, from http://www.economist.com/world/displaystory.cfm?story_id=E1_GPPG

St.Amant, K. (2007). Outsourcing: Perspectives, practices, and projections, *IEEE Transactions on Professional Communication, 50*, 81-84.

Staab, S., & Studer, R. (2004). *Handbook on ontologies* (International handbooks on information systems). Springer.

Stanoi, I., Agrawal, D., El Abbadi, A., Phatak, S. H., & Badrinath, B. R. (1999). Data Warehousing Alternatives for Mobile Environments. *Proc. of the ACM Int. Work. on Data Engineering for Wireless and Mobile Access*, (pp. 110-115).

Stauffer, D., & Aharony, A (1992). *Introduction to percolation theory* (2nd ed.). London, UK: Taylor and Francis.

Stefanovic, N., Han, J., & Koperski, K. (2000). Object-based selective materialization for efficient implementation of spatial data cubes. *Transactions on Knowledge and Data Engineering, 12*(6), 938–958.

Stephens, L., Gangam, A., & Huhns, M. (2004). Constructing consensus ontologies for the Semantic Web: A conceptual approach. *In World Wide Web: Internet and Web Information Systems, 7*, 421-442. Kluwer Academic Publishers.

Stinson, D. R. (2006). *Cryptography theory and practice* (3rd ed.). Chapman & Hall/CRC.

Stohr, E. A., & Zhao, J. L. (2001). Workflow automation: Overview and research issues. *Information Systems Frontiers, 3*(3), 281-296.

Stoica, I., Morris, R., Karger, D., Frans Kaashoek, M., & Balakrishnan, H. (2001). Chord: A scalable peer-to-peer lookup service for Internet applications. *Proceedings of the 2001 ACM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications* (pp. 149-160).

Stoimenov L., & Đorđević-Kajan, S. (2003). Realization of GIS semantic interoperability in local community environment. *6th AGILE Conference on Geographic Information Science: The Science Behind the Infrastructure* (pp. 73-80).

Stoimenov, L., & Djordjevic-Kajan, S. (2006). *Discovering mappings between ontologies in semantic integration process.* Proceedings of the AGILE Conference on Geographic Information Science, Visegrad, Hungary, 213–219.

Stoimenov, L., & Đorđević-Kajan, S. (2002). Framework for semantic GIS interoperability. *FACTA Universitatis, Series Mathematics and Informatics, 17,* 107-125.

Stoimenov, L., & Đorđević-Kajan, S. (2005). An architecture for interoperable GIS use in a local community environment. *Computers & Geoscience, 31*(2), 211-220.

Stoimenov, L., Đorđević-Kajan, S., & Stojanovic, D. (2000). Integration of GIS data sources over the Internet using mediator and wrapper technology. *Proceedings of MELECON 2000, 10th Mediterranean Electrotechnical Conference, 1,* 334-336.

Stoimenov, L., Mitrovic, A, Đorđević-Kajan, S., & Mitrovic, D. (1999). Bridging objects and relations: A mediator for an OO front-end to RDBMSs. *Information and Software Technology, 41*(2), 59-68.

Stoimenov, L., Stanimirovic, A., & Djordjevic-Kajan, S. (2006). *Discovering mappings between ontologies in semantic integration process.* Proceedings of the AGILE'06: 9th Conference on Geographic Information Science, Visegrd, Hungary, 213–219.

Stoimenov, L., Stanimirović, A., & Đorđević-Kajan, S. (2004). Realization of component-based GIS application framework. *7th AGILE Conference on Geographic Information Science* (pp. 113-120).

Stojanovic, N., Stojanovic, L., & Volz, R. (2002). A reverse engineering approach for migrating data-intensive Web sites to the semantic Web. *Proceedings of the Conference on Intelligent Information Processing.*

Stollnitz, E. J., Derose, T. D., & Salesin, D. H. (1996). *Wavelets for computer graphics.* Morgan Kauffmann.

Strehl, A., & Ghosh, J. (2002). Cluster ensembles - a knowledge reuse framework for combining multiple partitions. *Machine Learning Research, 3,* 583-417.

Stuckenschmidt, H., Wache, H., Vogele, T., & Vissar, H. (2000). Enabling technologies for interoperability. *Workshop on the 14th International Symposium of Computer Science for Environmental Protection* (pp. 35-46).

Stumme G., Rafik T., Bastide Y., Pasquier N., & Lakhal L. (2002). Computing Iceberg Concept Lattices with Titanic. Journal on Knowledge and Data Engineering, 42(2), 189-222.

Stumme, G., (1999), Conceptual Knowledge Discovery with Frequent Concept Lattices, FB4-Preprint 2043, TU Darmstadt

Subrahmanian, V. S. (1994). Amalgamating knowledge bases. *ACM-TODS, 19*(2), 291-331.

Suzuki, E., & Zytkow, J. M. (2005). Unified Algorithm for Undirected Discovery of Exception Rules. *International Journal of Intelligent Systems, 20*(7), 673-691.

Swagerman, D. M., Dogger, N., & Maatman, S. (2000). Electronic markets from a semiotic perspective. Electronic Journal of Organizational Virtualness, 2(2), 22-42.

SWIFT (2007). Society for Worldwide Interbank Financial Telecommunication (SWIFT). www.swift.com

Sybase (2004). *Adaptive Server Anywhere Datenbankadministration,* Bestel lnummer: DC03928-01-0902-01. iAnywhere, A Sybase Company.

Syrjanen, T., & Niemela, I. (2001). The Smodels system. *International Conference on Logic Programming and Nonmonotonic Reasoning* (pp. 434-438).

Syukur, E., & Loke, S. (2006). Implementing context-aware regulation of context-aware mobile services in pervasive computing environments. *International Journal of Web and Grid Services, 2*(3), 260-305.

Tagarelli, A. & Greco, S. (2006). Toward semantic XML clustering. *Proceedings of the SIAM International Conference on Data Mining* (pp. 188-199).

Tagarelli, A., & Greco, S. (2004). Clustering transactional XML data with semantically-enriched content and structural features. *Proceedings of the International Conference on Web Information Systems Engineering* (pp. 266-278).

Tait, C., Lei, H., Acharya, S., & Chang, H. (1995). Intelligent File Hoarding for Mobile Computers. *Proc. of the 1st ACM Int. Conf. on Mobile Computing and Networking,* (pp. 119-125).

Takahashi, J. (1990). Hybrid relations for database schema evolution. *14th Annual International Computer Software and Applications Conference*, Chicago, Ilinois, USA.

Tanasescu, V., et al. (2007). Geospatial data integration with semantic Web services: The eMerges approach. In A. Scharl, & K. Tochtermann (Eds.), *The geospatial Web***. London:** Springer.

Tang, C., Xu, Z., & Dwarkadas, S. (2003). Peer-to-peer information retrieval using self-organizing semantic overlay networks. *Proceedings of the 2003 ACM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications* (pp. 175-186).

Tang, J., Chen, Z., Fu, A., & Cheung, D. W. (2002). Enhancing effectiveness of outlier detections for low density patterns. *Proceedings of the 6th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD'02)* (pp. 535-548).

Taniar, D., & Goel, S. (2007). Concurrency control issues in grid databases. *Future Generation Computer Systems, 23*(1), 154-162.

Tansel, A. U (1997). Temporal relational data model. *IEEE Transactions on Knowledge and Database Engineering, 9*(3), 464-479.

Tansel, A. U (2004). On handling time-varying data in the relational databases. *Journal of Information and Software Technology, 46*(2), 119-126.

Tansel, A. U et al. (1993). (Ed.). *Temporal databases: Theory, design and implementation.* Benjamin/Cummings.

Tansel, A. U, & Eren-Atay, C., (2006). Nested bitemporal relational Algebra. *ISCIS 2006,* 622-633.

Tansel, A. U., & Garnett, L. (1989). Nested temporal relations. I*n Proceedings of ACM feshi SIGMOD International Conference on Management of Data,* 284-293.

Tansel, A. U., & Tin, E. Expressive power of temporal relational query languages. *IEEE Transactions on Knowledge and Data Engineering, 9*(1), 120-134.

Tansel, A. U., Arkun, M. E., & Ozsoyoglu, G. (1989). Time-by-Example query language for historical databases. *IEEE Transactions on Software Engineering, 15*(4), 464-478.

Tansel, A.U (1986). Adding time dimension to relational model and extending relational algebra. *Information Systems 11*(4), 343-355.

Tao, Y., & Papadias, D. (2001a). Efficient historical Rtrees. *Proc. of the International Conference on Scientific and Statistical Database Management (SSDBM'2001),* 223-232. George Mason University, Fairfax, Virginia.

Tao, Y., & Papadias, D. (2001b). MV3RTree: A spatio-temporal access method for timestamp and interval queries. *Proc. of the 27th International Conference on Very Large Data Bases, (VLDB'2001),* 431-440. Roma, Italy.

Tao, Y., Faloutsos, C., Papadias, D., & Liu, B. (2004). Prediction and indexing of moving objects with unknown motion patterns. *Proc. of the ACM Conference on the Management of Data (SIGMOD'2004).* Paris, France, to appear.

Tao, Y., Papadias, D., & Sun, J. (2003). The TPR*Tree: An optimized spatiotemporal access method for predictive queries. *Proc. of the 29th International Conference on Very Large Data Bases, (VLDB'2003),* 790-801. Berlin, Germany.

Tatarinov, I., & Halevy., A. (2004). Efficient query reformulation in peer data management systems. *SIGMOD* (pp. 539-550).

Tayeb, J., Ulusoy, O., & Wolfson, O. (1998). A quadtree based dynamic attribute indexing method. *The Computer Journal, 41*(3), 185-200.

Teorey, T. J., Yang, D., & Fry, J. P. (1986). A logical design methodology for relational databases using the extend entity-relationship model. *Computing Surveys, 18*(2).

Terra, J., & Gordon, C. (2003). *Realizing the promise of corporate portals.* Boston, MA: Butterworth-Heinemann.

The new geography of the IT industry. (2003, 17 July). *The Economist,* Retrieved December 20, 2003, from http://www.economist.com/displaystory.cfm?story_id=S%27%29HH%2EQA%5B%21%23%40%21D%0A

Theodoratos, D., & Bouzeghoub, M. (2000). A general framework for the view selection problem for data warehouse design and evolution. *3rd ACM International Workshop on Data Warehousing and OLAP (DOLAP 2000)* (pp. 1-8).

Theodoridis, S., & Koutroumbas, K. (2006). *Pattern Recognition*, Third Edition: Academic Press, Inc.

Theodoridis, Y., Stefanakis, E., & Sellis, T. (2000). Efficient Cost Models for Spatial Queries Using R-Trees. *IEEE Transactions Knowledge Data Engineering, 12*(1), 19-32.

Theodoridis, Y., Vazirgiannis, M., & Sellis, T. (1996). Spatio-temporal indexing for large multimedia applications. *Proc. of the 3rd IEEE International Conference on Multimedia Computing and Systems (ICMCS'1996),* 441-448. Hiroshima, Japan.

Thistlewaite, P. (1997). Automatic Construction and Management of Large Open Webs. *Information Processing and Management, 33*(2), 161-173.

Thomas, R. H. (1979). A Majority Consensus Approach to Concurrency Control for Multiple Copy Databases. *ACM Transactions on Database Systems, 4*(2), 180-209.

Thomason, R. H. (2007). Conditional and Action Logics. *In Logical Formalizations of Commonsense Reasoning: Papers from the AAAI Spring Symposium.* (AAAI Technical Report No SS-07-05), Menlo Park, California, USA: AAAI Press.

Thome B., Gawlick D., & Pratt M. (2005). Event Processing with an Oracle Database. *In Proceedings of SIGMOD* (pp.863-867), Baltimore, Maryland.

Thompson, H. S., Beech, D., Maloney, M., & Mendelsohn, N. (2004). *XML Schema Part 1: Structures (Second Edition).* W3C Recommendation. www.w3.org/TR/xmlschema-1/.

Thompson, H. S., Beech, D., Maloney, M., & Mendelsohn, N., eds. (2001). *XML Schema Part 1: Structures – W3C Recommendation 02 May 2001.* W3C (http://www.w3.org/TR/xmlschema-1/).

Tiakas, E., Papadopoulos, A.N., Nanopoulos, A., Manolopoulos, Y., Stojanovic, D., & Djordjevic-Kajan, S.

(2006). *Trajectory similarity search in spatial networks.* Proceedings of the IDEAS 2006, 185–192.

Tiako, P., & Derniame, J.-C. (1999). *Toward process components mobility in federated PSEES*. Proceedings of the 5th International Conference on Information Systems, Analysis and Synthesis: ISAS'99, Orlando, Florida.

Tiako, P.F. (2005). *Collaborative approach for modeling and performing mobile software process components.* Proceedings of the 2005 International Symposium on Collaborative Technologies and Systems (CTS 2005), Austin, Texas.

Tikk, D., Biró, Gy., & Törcsvári, A. (2007). A hierarchical online classifier for patent categorization. In H. A. do Prado & E. Ferneda (Eds.), *Emerging Technologies of Text Mining: Techniques and Applications.* Idea Group Inc. (in press).

Tilley, S. R., & Smith, D. B. (1995). *Perspectives on legacy system reengineering* (Tech. Rep.). Carnegie Mellon University, Software Engineering Institute.

Tomai, E., & Prastacos, P. (2006). *A framework for intensional and extensional integration of geographic ontologies.* Proceedings of the AGILE Conference on Geographic Information Science, Visegrad, Hungary, 220–227.

Tomasic, A., Kapitskaia, O., Naacke, H., Bonnet, P., Raschid, L., & Amouroux, R. (1997). *The distributed information search component (disco) and the World Wide Web.* Proceedings of the ACM SIGMOD Conference on Management of Data.

Tomic, B., Jovanovic, J., & Devedzic, V. (2006). JavaDON: An Open-Source Expert System Shell. *Expert Systems With Applications, 31*(3), 595-606

Torlone, R. (2003). Conceptual multidimensional models. In M. Rafanelli (Ed.), *Multidimensional databases: Problems and solutions* (pp. 91–115). Hershey, PA: Idea Group Publishing.

Touati, M., & Diday, E. Sodas Home Page. http://www.ceremade.dauphine.fr/~touati/sodas-pagegarde.htm. Last access on June 2008.

Tousidou, E., Bozanis, P. and Manolopoulos, Y. (2002). Signature-based structures for objects with set-values attributes. *Information Systems,* 27(2):93-121.

Touting, T., & Gray, L. (2000). State - of - the - art of elevation extraction from satellite SAR data. *Photogrammetry and Remote Sensing, 55*, 13-33.

Truong, H., & Fahringer, T. (2004). SCALEA-G: A unified monitoring and performance analysis system for the grid. *European Across Grids Conference* (pp. 202-211).

Tryfona, N., Busborg, F., & Borch, J. (1999). *StarER: A conceptual model for data warehouse design*. Proceedings of the 2nd ACM International Workshop on Data Warehousing and OLAP, 3–8.

Tseng, V. S., & Kao, C.-P. (2005). Efficiently mining gene expression data via a novel parameterless clustering method. *IEEE/ACM Transactions on Computational Biology and Bioinformatics, 2*(4), 355-365.

Tsois, A., Karayannidis, N., & Sellis, T. (2001). *MAC: Conceptual data modeling for OLAP*. Proceedings of the 3rd International Workshop on Design and Management of Data Warehouses, 5.

Tsoumakos, D., & Roussopoulos, N. (2003a). Adaptive probabilistic search for peer-to-peer networks. *Proceedings of the 3rd IEEE International Conference on Peer-to-Peer Computing* (pp. 102-109).

Tsoumakos, D., & Roussopoulos, N. (2003b). A comparison of peer-to-peer search methods. *Proceedings of the 2003 ACM International Workshop on Web and Databases, in conjunction with the 2003 ACM International Conference on Management of Data* (pp. 61-66).

Tung, A. K., Zhang, R., Koudas, N., & Ooi, B. C. (2006). Similarity search: A matching-based approach. *Proceedings of VLDB*, 631-642. Seoul, Korea.

Turel, O. (2006). Contextual effects on the usability dimensions of mobile value-added services: A conceptual framework. *International Journal of Mobile Communications, 4*(3), 309-332.

Türker, C., & Gertz, M. (2000). *Semantic integrity support in SQL-99 and commercial (object) relational database management systems.* UC Davis Computer Science Technical Report CSE-2000-11, University of California.

Türker, C., & Gertz, M. (2001). Semantic integrity support in SQL:1999 and commercial (object-) relational database management systems. *The VLDB Journal, 10*, 241-269.

Turner, H. L., Bailey, T.C., Krzanowski, W. J., & Hemingway, C. A. (2005). Biclustering models for structured microarray data. *IEEE/ACM Transactions on Computational Biology and Bioinformatics, 2*(4), 316-329.

Tveit, A. (2001). A Survey of Agent-Oriented Software Engineering. *Proceedings of the First NTNU Computer Science Graduate Student Conference.*

Tversky, A. (1977). Features of similarity. *Psychological Review, 84,* 327-352.

Tversky, A., & Gati, I. (1982). Similarity, separability, and the triangle inequality. *Psychological Review, 89,* 123-154.

Tyagi, B. K., Sharfuddin, A., Dutta, R. N., & Tayal, D.K. (2005). A complete axiomatization of fuzzy functional dependencies using fuzzy function. *Fuzzy Sets and Systems, 151*(2), 363-379.

Tzanis, G., & Vlahavas, I. (2007). Mining High Quality Clusters of SAGE Data. *Proceedings of the 2nd VLDB Workshop on Data Mining in Bioinformatics,* Vienna, Austria.

Tzanis, G., Berberidis, C., & Vlahavas, I. (2007). MANTIS: A Data Mining Methodology for Effective Translation Initiation Site Prediction. *Proceedings of the 29th Annual International Conference of the IEEE Engineering in Medicine and Biology Society,* IEEE, Lyon, France.

Tzouramanis, T., Vassilakopoulos, M., & Manolopoulos, Y. (2003). Overlapping linear quadtrees and spatio-temporal query processing. *The Computer Journal, 43*(4), 325-343.

Tzouramanis, T., Vassilakopoulos, M., & Manolopoulos, Y. (2004). Benchmarking access methods for time-evolving regional data. *Data & Knowledge Engineering, 49* (3), 243-286.

Tzvetkov, P. et al. (2003). TSP: Mining Top-K Closed Sequential Patterns. *Proceedings of the 2003 International Conference on Data Mining,* USA, (pp. 347-354).

U.S. Congress (2002, July 30) Public Law 107-204: Sarbanes-Oxley Act of 2002 Retrieved July 16th 2004 from http://frwebgate.access.gpo.gov/cgi-bin/getdoc.cgi?dbname=107_cong_public_laws&docid=f:publ204.107.pdf

Ubeda, T., & Servigne, S. (1996). Geometric and Topological Consistency of Spatial Data. *Proceedings of the 1st International Conference on Geocomputation, ,* 1, 830-842, Leeds, United Kingdom, September 17-19.

Uchiyama, H., Runapongsa, K., & Teorey, T. J. (1999). A progressive view materialization algorithm. *2nd International Workshop on Data Warehousing and OLAP (DOLAP 1999)* (pp. 36-41).

Ulijn, J. M. & St.Amant K. (2000). Mutual intercultural perception: How does it affect technical communication–some data from China, the Netherlands, Germany, France, and Italy. *Technical Communication, 47*(2), 220-37.

Ulijn, J. M. (1996). Translating the culture of technical documents: Some experimental evidence.

Ulijn, J. M., & Strother, J. B. (1995). *Communicating in business and technology: From psycholinguistic theory to international practice*. Frankfurt, Germany: Peter Lang.

Ullman J.D (1988). Principles of Database and Knowledge-base Systems , Volume 1. Computer Science Press.

Ulrich, S., & Torben, S. (2008), Are international co-publications an indicator for quality of scientific research? *Scientometrics, 74*(3), 361-377.

Ultsch, A., & Panda, P. G. (1991). Die Kopplung konnektionistischer Modelle mit wissensbasierten Systemen. In *Tagungsband Experteny stemtage Dortmund* (pp. 74-94). VDI Verlag.

Ultsch, A., & Siemon, H. P. (1990). Kohonen's self organizing feature maps for exploratory data analysis. In *Proc. Intern. Neural Networks* (pp. 305-308). Kluwer Academic Press.

Ulusoy, O. (1992). Concurrency control in real-time database systems. *PhD Thesis, Department of Computer Science, University of Illinois, Urbana-Champaign.*

Ulusoy. (1993). Lock-based concurrency control in distributed real-time database systems. *Journal of Database Management, 4*(2), 3–16.

Ulusoy. (1994). Processing real-time transactions in a replicated database system. *Distributed and Parallel Databases, 2*(4), 405–436.

Uren, V., Cimiano, P., Iria, J., Handschuh, S., Vargas-Vera, M., Motta, E., & Ciravegna, F. (2006). Semantic annotation for knowledge management: Requirements and a survey of the state of the art. *Web Semantics: Science, Services and Agents on the World Wide Web, 4*(1), 14-28.

Urhan, T., & Franklin, M.J. (2000). Xjoin: A reactively-scheduled pipelined join operator. *IEEE Data Engineering Bulletin, 23*, 27–33.

Urhan, T., & Franklin, M.J. (2001). *Dynamic pipeline scheduling for improving interactive performance of online queries.* Proceedings of the International Conference on Very Large Databases VLDB'01.

Urhan, T., Franklin, M.J., & Amsaleg, L. (1998). *Cost based query scrambling for initial delays.* Proceedings of the 1998 ACM SIGMOD International Conference on Management of Data, 130–141.

Using Rose Data Modeler (2001). Retrieved October 6, 2007, from ftp://ftp.software.ibm.com/software/rational/docs/v2002/Rose_dm.pdf.

v. d. Meyden, R. (1997). The complexity of querying indefinite data about linearly ordered domains. *Journal of Computer and System Sciences, 54*, 113–135.

Vadaparty K.V. and Naqvi S.A.(1995). Using Constraints for Efficient Query Processing in Nondeterministic Databases, IEEE Trans. Knowl. Data Eng., 7(9):860-864.

Vadaparty, K. (1999). ODBMS - *Bridging the gap between objects and tables: Object and data models, 12*(2).

Valduriez P. (1993). Parallel Database Systems: Open Problems and New Issues. *Distributed and Parallel Databases, 1*(2), 137–165.

Valentin, G., Zuliani, M., Zilio, D., Lohman, G., & Skelley, A. (2000). DB2 advisor: An optimizer smart enough to recommend its own indexes. *16th International Conference on Data Engineering, (ICDE 2000)* (pp. 101-110).

Valluri, S. R., Vadapalli, S., & Karlapalem, K. (2002). View relevance driven materialized view selection in data warehousing environment. *13th Australasian Database Conference (ADC 2002)* (pp. 187-196).

van den Akker, J., & Siebes, A. (1997). Enriching Active Databases with Agent Technology. *Proceedings of CIA 1997*, (pp. 116-125).

van der Aalst, W., ter Hofstede, A. H. M., Kiepuszewski, B., & Barros, A. P. (2003). Workflow patterns. *Distributed and Parallel Databases, 14*, 5-51.

Van der Wel, F., Peridigao, A., Pawel, M., Barszczynska, M., & Kubacka, D. (2004, June). COST 719: Interoperability and integration issues of GIS data in climatology and meteorology. *Proceedings of the 10th EC GI & GIS Workshop: ESDI State of the Art*, Warsaw, Poland.

Van Hentenryck, P. (1999). *The OPL optimization programming language.* MIT Press.

Van Hentenryck, P., Michel, L., Perron, L., & Regin, J. C. (1999). Constraint programming in OPL. *International Conference on Principles and Practice of Declarative Programming* (pp. 98-116).

van Onselen, P., & Errington, W. (2004). Electoral Databases: Big Brother or Democracy Unbound? *Australian Journal of Political Science, 39*(2), 349-366.

Van Roy, P. (1999). Logic programming in Oz with Mozart. *International Conference on Logic Programming* (pp. 38-51).

Vara, J. M., Vela, B., Cavero, J. M., & Esperanza, M. (2007). Model transformation for object-relational database development. In *Proceedings of the 2007 ACM symposium on Applied computing* (pp. 1012-1119). Seoul: ACM Press.

Vazirgiannis, M., & Wolfson (2001). *A spatiotemporal model and language for moving objects on road networks.* Proceedings of the SSTD 2001, 20–35.

Vazirgiannis, M., & Wolfson, O. (2001). A spatio temporal model and language for moving objects on road networks. *Proceedings of 7th International Symposium on Advances in Spatial and Temporal Databases* (pp. 20-35).

Vckovsky, A. (1998). *International Journal of Geographic Information Science, 12*(4).

Velculescu, V. E., Zhang, L., Vogelstein, B., & Kinzler, K. W. (1995). Serial Analysis of Gene Expression. *Science, 270*(5235), 484-487.

Verelst, J. (2004). The Influence of the level of Abstraction on the Evolvability of conceptual models of information systems. *International Symposium on Empirical Software Engineering (ISESE 2004)*, Redondo Beach, CA, USA, IEEE Computer Society

Vessey, I., & Conger, S. A. (1994). Requirements specification: Learning object, process, and data methodologies. Communications of the ACM, 37(5), 102-113.

Visser, P., Jones, D., Bench-Capon, T., & Shave, M. (1997). *An analysis of ontology mismatches; heterogeneity versus interoperability.* Proceedings of the AAAI 1997 Spring Symposium on Ontological Engineering.

Visser, U. (2004). *Intelligent information integration for the semantic Web.* Volume 3159. Lecture Notes in Computer Science. Berlin-Heidelberg: Springer.

Visser, U., & Stuckenschmidt, H. (2002). Interoperability in GIS: Enabling technologies. *5th AGILE Conference on Geographic Information Science* (pp. 291-297).

Viswanath N. and Sunderraman R.(2007). Query Processing in Paraconsistent Databases in the Presence of Integrity Constraints, In the Proceedings of the Nineteenth International Conference on Software Engineering and Knowledge Engineering SEKE 2007, Boston, USA.

Vitali, F. (1999). Versioning hypermedia. *ACM Computing Surveys, 31*(4es), 24.

Vitter, J. (2001). External memory algorithms and data structure: Dealing with massive data. *ACM Computing Surveys, 33*(2), 209-271.

Vitter, J. S. (1985). Random sampling with a reservoir. *ACM Transactions on Mathematical Software, 11*(1), 37-57.

Vitter, J. S., Wang, M., & Iyer, B. (1998). Data cube approximation and histograms via wavelets. *Proceedings of the 7th ACM International Conference on Information and Knowledge Management* (pp. 96-104).

Vitter, J.S., Wang, M., & Iyer, B. (1998). Data Cube Approximation and Histograms via Wavelets. *Proc. of the 7th ACM Int. Conf. on Information and Knowledge Management,* (pp. 96-104).

Vlachos, M., Hatjieleftheriou, M., Gunopoulos, D., & Keogh, E. (2003). Indexing multidimensional time series with support for multiple distance measures. *Proceedings of ACM KDD*, 216-225. Washington, DC, USA.

Vlachos, M., Kollios, G., & Gunopoulos, D. (2002). Discovering similar multidimensional trajectories. *Proceedings of IEEE ICDE*, 673-684. San Jose, California, USA.

Voisard, A., & Juergens, M. (1998). Geographic information extraction: Querying or quarrying? In M. Goodchild, M. Egenhofer, R. Fegeas, & C. Kottman (Eds.), *Interoperating geographic information systems.* New York: Kluwer Academic Publishers.

Vretanos, P.A. (2005). Web feature service implementation specification, Version 1.1.0. Open Geospatial Consortium. Retrieved from http://www.opengeospatial.org/standards/wfs

W3C. (2004). Extensible markup language (XML). http://www.w3.org/XML/.

Wache, H., et al. (2001). *Ontology-based integration of information—A survey of existing approaches.* Proceedings of the IJCAI-01 Workshop: Ontologies and Information Sharing, Seattle, Washington, 108–117.

Wache, H., Vögele, T., Visser, U., Stuckenschmidt, H., Schuster, G., Neumann, H., et al. (2001). Ontology-based integration of information: A survey of existing approaches. *IJCAI-01 Workshop: Ontologies and Information Sharing* (pp. 108-117).

Wagstaff, K., & Cardie, C. (2000). *Clustering with Instance-level Constraints.* Paper presented at ICML'00: the seventeenth International Conference on Machine Learning, Morgan Kaufmann Publishers Inc., 2000, 1103-1110.

Wakayama, T., Kannapan, S., Khoong, C. M., Navathe, S., & Yates, J. (1998). Information and Process Integration in Enterprises: Rethinking Documents. Norwell, MA: Kluwer Academic Publishers.

Wallace, M., & Schimpf, J. (1999). ECLiPSe: Declarative specification and scaleable implementation. *International Workshop on Practical Aspects of Declarative Languages* (pp. 365-366).

Wang, F., Zhou, X., & Zaniolo, C. (2006). Bridging relational database history and the web: the XML approach. *In Eight ACM International Workshop on Web Information and Data Management (WIDM'06)* (pp. 3-10). Arlington, Virginia, USA: ACM.

Wang, H., & Meng, X. (2005, April). On the sequencing of tree structures for XML indexing In *Proc. Conf. Data Engineering*, 372-385. Tokyo, Japan.

Wang, H., Park, S., Fan, W., & Yu, P. S. (2003, June). ViST: A dynamic index method for querying XML data

by tree structures. *SIGMOD Int. Conf. on Management of Data*, San Diego, CA..

Wang, J., & Lochovsky, F. (2003). Data extraction and label assignment for web databases. *12th International Conference on* World Wide Web *(WWW'03),* 187-196.

Wang, K., Zhou, S., Yang, Q., & Yeung, J.M.S. (2005). Mining Customer Value: From Association Rules to Direct Marketing. *Data Mining and Knowledge Discovery, 11,* 5779.

Wang, L., Jajodia, S., & Wijesekera, D. (2004). Securing OLAP data cubes against privacy breaches. *Proceedings of 2004 IEEE Symposium on Security and Privacy* (pp. 161-175).

Wang, S. L., Tsai, J. S., & Hong, T. P. (2000). Mining functional dependencies from fuzzy relational databases. *Proceedings of the 2000 ACM symposium on Applied computing, 1,* 490-493.

Wang, W., Yang, J., & Muntz, R. R. (1997). *STING: A Statistical Information Grid Approach to Spatial Data Mining.* Paper presented at VLDB'97: the 23rd International Conference on Very Large Data Bases, August 25-29, 1997, Athens, Greece.

Wang, W., Zhang, J., & Wang, H. (2005). Grid-ODF: Detecting outliers effectively and efficiently in large multi-dimensional databases. *Proceedings of the 2005 International Conference on Computational Intelligence and Security (CIS'05)* (pp. 765-770).

Wang, Y.M., & Fuchs, W.K. (1993). *Lazy checkpoint coordination for bounding rollback propagation.* Proceedings of the 12th Symposium on Reliable Distributed Systems, 78–85.

Ward, M. (1992). *The syntax and semantics of the wide spectrum language* (Tech. Rep.). England: Durham University.

Warner, S. L. (1965). Randomized response: A survey technique for eliminating evasive answer bias. *Journal of the American Statistical Association, 60*(309), 63-69.

Warren, D. E., Dunfee, T. W., & Li, N. (2004). Social Exchange in China: The Double-Edged Sword of Guanxi. *Journal of Business Ethics, 55,* 355-372.

Wasserman, S., & Faust, K. (1994). *Social Network Analysis: Methods and Applications.* Cambridge, England, and New York: Cambridge University Press.

Watson, E. E., & Schneider, H. (1999). Using ERP systems in education. Communications of the Association for Information Systems, 1(9).

Watts, D. J., Dodds, P. S., & Newman, M. E. J. (2002). Identity and search in social networks. *Science, 296,* 1302-1305.

Weber, R. (1986). Data models research in accounting: An evaluation of wholesale distribution software. The Accounting Review, 61(3), 498-518.

Weber, R. (1996). Are attributes entities? A study of database designers' memory structures. Information Systems Research, 7(2), 137-162.

Weber, R., Schek, H.-J., & Blott, S. (1998). A quantitative analysis and performance study for similarity-search methods in high-dimensional spaces. *Proceedings of VLDB,* 194-205. New York City, USA.

Webre, N. W. (1981). An extended entity-relationship model. *Proceedings of the Second International Conference on the Entity-Relationship Approach to Information Modeling and Analysis.*

Wedemeijer, L. (2000). Defining Metrics for Conceptual Schema Evolution. *9th International Workshop on Foundations of Models and Languages for Data and Objects, FoMLaDO/DEMM 2000,* Dagstuhl Castle, Germany, Springer.

Wei, A. (1982). The dependent origination-the core of Buddhism. *Dharmaghosa (The Voice of Dharma), 1982*(01), 31-33.

Weiderhold, G. (1995). Modelling and system maintenance. *Proceedings of the International Conference on Object-Orientation and Entity-Relationship Modelling.*

Weingart, P. (2005). Impact of bibliometrics upon the science system: Inadvertent consequences? *Scientometrics, 62,* 117-131.

Weiser, M. (1991). The computer of the 21st century. *Scientific American, 265*(3), 66–75.

Welsh, M. (2004). Exposing Resource Tradeoffs in Region-Based Communication Abstractions for Sensor Networks. In SIGCOMM, 34(1).

WfMC. (2006). Interoperability abstract specification. WfMC Document Number TC-1012, Version 2.0b.

Retrieved December 2006, from http://www.wfmc.org/standards/docs/TC-1012_Nov_99.pdf

WfMC. (2008). *Workflow management coalition*. Retrieved July 16, 2008, from www.wfmc.org.

Whishart, D.S. (2002). Tools for Protein Technologies. In C. W. Sensen, (Ed.), *Biotechnology, 5b, Genomics and Bioinformatics*, 325-344. Wiley-VCH.

Widom, J. (1999). Data Management for XML – Research Directions. *IEEE Data Engineering Bulletin. Special Issue on XML, 22*(3), 44-52.

Widom, J. (1999). Data management for XML: Research directions. *IEEE Data Engineering Bulletin, 22*(3), 44-52.

Wiederhold, G. (1992). Mediators in the architecture of future information systems. *IEEE Computer, 25*(3), 38–49.

Wiederhold, G. (1998). Weaving data into information. *Database Programming & Design, 11*(9), 22-29.

Wienberg, A., Ernst, M. et al. (2002). Content Schema Evolution in the CoreMedia Content Application Platform CAP. Advances in Database Technology - EDBT 2002. *8th International Conference on Extending Database Technology*, Prague, Czech Republic, March 25-27, Proceedings, Prague, Czech Republic, Springer.

Wiesmann, M., & Schiper, A. (2005). Comparison of Database Replication Techniques Based on Total Order Broadcast. *IEEE Transactions on Knowledge and Data Engineering, 17(4),* 206-215.

Wiesmann, M., Pedone, F., Schiper, A., Kemme, B., & Alonso, G. (2000). Database Replication Techniques: A Three-Parameter Classification. *Symposium on Reliable Distributed Systems,* (pp. 206-215).

Wilkinson, R. Arnold-Moore, T., Fuller, M., Sacks-Davis, R., Thom J., & Zobel, J. (1998). *Document computing: technologies for managing electronic document collections.* Boston, MA: Kluwer Academic Publishers.

Willenborg, L., & de Waal, T. (2001). Elements of statistical disclosure control. In *Lecture notes in statistics* (Vol. 155). Springer-Verlag.

Winter, R., & Strauch, B. (2003). Demand-driven information requirements analysis in data warehousing. *Proc. of the 36th Hawaii International Conference on System Sciences,* 1359-1365.

Wired China (2000, July 22). *The Economist,* pp. 24-28.

Witten, I., & Frank, E. (2005). *Data mining: Practical machine learning tools and techniques.* 2nd Edition. Morgan Kaufmann.

Woehrer, A., et al. (2005). Towards semantic data integration for advanced data analysis of grid data repositories. Proceedings of the First Austrian Grid Symposium, Schloss Hagenberg, Austria.

Woelk, D., Cannata, P., Huhns, M., Shen, W., & Tomlinson, C. (1993). Using Carnot for enterprise information integration. *Second International Conf. Parallel and Distributed Information Systems,* 133-136.

Wojciechowski, M. (1999). Mining various patterns in sequential data in an SQL-like manner. Proceedings of the Advances in Database and Information Systems Third East European Conference, ADBIS'99, Maribor, Slovenia, 131–138.

Wolfson, O., et al. (1999). *Tracking moving objects using database technology in DOMINO.* Proceedings of the NGITS 1999, 112–119.

Wolski, R., Spring, N. T., & Hayes, J. (1999). The network weather service: A distributed resource performance forecasting service for metacomputing. *Future Generation Computer Systems, 15*(5-6), 757-768.

Wong, K., & Sun, D. (2006). On evaluating the layout of UML diagrams for program comprehension. *Software Quality Journal, 14*(3), 233-259.

Woo, S.K., Kim, M.H., & Lee, Y.J. (1997). A recovery method based on shadow updating and fuzzy checkpointing in main memory database systems. *Korea Information Science Society Journal, 24*(3), 266–278.

Wooldridge, M., & Ciancarini, P. (2001). Agent-Oriented Software Engineering: The State of the Art. *LNCS, 1957,* 1-28.

Woolever, K. R. (2001). Doing global business in the information age: Rhetorical contrasts in the business and technical professions. In C. G. Paneta (Ed.), *Contrastive Rhetoric Revisited and Redefined* (pp. 47-64). Mahwah, NJ: Lawrence Erlbaum Associates.

World Wide Web Consortium. (2001). *XQuery 1.0: An XML Query Language, W3C Recommendation, Version 1.0*, Dec. 2001. See http://www.w3.org/TR/xquery.

World Wide Web Consortium. (2007). *XML path language (XPath). W3C Recommendation*, 2007. See http://www.w3.org/TR/xpath20.

Wu, B., Lawless, D., Bisbal, J., Richardson, R., Grimson, J., Wade, V., et al. (1997). The butterfly methodology: A gateway-free approach for migrating legacy information system. In *ICECOS* (pp. 200-205). Los Alamos, CA: IEEE Computer Society Press.

Wu, G. (1996). Commentary to the dependent arising and emptiness. *Wuhan University Journal (Philosophy & Social Science Edition), 1996*(02), 35-38.

Wu, H., Wang, Q., Xu Yu, J., Zhou, A., & Zhou, S. (2003). *UD(k and l)-index: An efficient approximate index for XML data*. Proceedings of the 4th International Conference on Advances in Web-Age Information Management (WAIM 03), Chengdu, China, 2762, 68–79.

Wu, P., Wen, J.-R., Liu, H., & Ma, W.-Y. (2006). Query selection techniques for efficient crawling of structured web sources. 22nd International Conference on Data Engineering (ICDE'06), 47.

Wu, X. (1999). The reconstruction of dependent arising and emptiness contributed by the mind-only theory. *Fudan Journal (Social Sciences), 1999*(03), 67-88.

Wu, X., Ye, Y., & Zhang, L. (2003). Graphical modeling based gene interaction analysis for microarray data. *SIGKDD Explorations Newsletter, 5*(2), 91-100.

Wyse, J. E. (2006). Location-aware query resolution for location-based mobile commerce: performance evaluation and optimization. *Journal of Database Management, 17*(3), 41-65.

Wyse, J. E. (2007). Applying location-aware linkcell-based data management to context-aware mobile business services. *Proceedings of the Sixth International Conference on Mobile Business*, Toronto, Ontario, Canada, (July 9-11): 8 pages.

Wyse, J. E. (2008). Optimizing server performance for location-aware applications in mobile commerce: The repository manager's formula. *International Journal of Wireless and Mobile Computing.* (forthcoming).

Wyse, J. E. (2003). Supporting m-commerce transactions incorporating locational attributes: An evaluation of the comparative performance of a location-aware method of locations repository management. *International Journal of Mobile Communications, 1*(1/2), 119-147.

Wyse, J. E. (2004). *System and Method for Retrieving Location-Qualified Site Data.* Patent Number 6,792,421, United States Patent and Trademark Office. Patent Issued: September 14.

X-Hive Corporation. (2007). The fastest and most scalable XML database powered by open standards. Retrieved February 26, 2007, from http://www.x-hive.com/products/db/

Xi, Y., Martin, P., & Powley, W. (2001). An analytical model for buffer hit rate prediction. *In Proceedings Of the 2001 Conference of the Centre for Advanced Studies on Collaborative Research*, 18-29.

Xia, T., Zhang, D., Kanoulas, E., & Du, Y. (2005). On Computing Top-t Most Influential Spatial Sites. *In Proceedings of the Very Large Data Bases Conference (VLDB 2005)*, Trondheim, Norway, August 30 - September 2, 2005, (pp. 946-957). Morgan Kaufmann, San Francisco.

Xiao, Y.Y., Liu, Y.S., Deng, H.F., & Liao, G.Q. (2006). One-phase real-time commitment for distributed real-time transactions. *Journal of Huazhong University of Science and Technology, 34*(4), 1–4.

Xiao-ling, W., Jin-feng, L., & Yi-sheng. D. (2003). An Adaptable and Adjustable Mapping from XML Data to Tables in RDB. In *Proceedings of the VLDB 2002 Workshop EEXTT and CAiSE 2002 Workshop DTWeb*, (pp. 117–130), London, UK. Springer-Verlag.

Xing, E. P., Jordan, M. I., & Karp, R. M. (2001). Feature selection for high-dimensional genomic microarray data. *Proceedings of the 18th International Conference on Machine Learning*, (pp. 601-608).

Xu, J., & Callan, J. (1998). Effective retrieval with distributed collections. *Proceedings of the 21st ACM International Conference on Research and Development in Information Retrieval* (pp. 112-120).

Xu, R., & Wunsch, D., II. (2005). Survey of clustering algorithms. *IEEE Transactions on Neural Networks, 16*(3), 645-678.

Xu, X., Han, J., & Lu, W. (1990). RTTree: An improved RTree indexing structure for temporal spatial databases. *Proc. of the International Symposium on Spatial Data Handling (SDH'1990)*, 1040-1049. Zurich, Switzerland.

Yan, L. L., & Ozsu, M. T. (1999). Conflict tolerant queries in Aurora Coopis. (pp. 279-290).

Yan, X. et al. (2003). CloSpan: Mining Closed Sequential Patterns in Large Datasets, *Proceedings of the SIAM International Conference on Data Mining*, USA, (pp. 166-177).

Yang, B., & Garcia-Molina, H. (2002). Efficient search in peer-to-peer networks. *Proceedings of the 22nd IEEE International Conference on Distributed Computing Systems* (pp. 5-14).

Yang, J. W., & Chen, X. O. (2002). A semi-structured document model for text mining. *Journal of Computer Science and Technology, 17*(5), 603-610.

Yao, A. C. (1986). How to generate and exchange secrets. *Proceedings of the 27th Annual Symposium on Foundations of Computer Science (FOCS'86)* (pp. 162-167).

Yao, Y., & Gehrke, J. E. (2002). The Cougar Approach to In-Network Query Processing in Sensor Networks. *Sigmod Record, 31*(3).

Yen, S. -J. (2005). Mining Interesting Sequential Patterns for Intelligent Systems. *International Journal of Intelligent Systems, 20*(1), 73-87.

Yeung, K. Y., Haynor, D. R., & Ruzzo, W. L. (2001). Validating clustering for gene expression data. *Bioinformatics, 17,* 309-318.

Yeung, Y. K., Medvedovic, M., & Bumgarner, R. E. (2003). Clustering Gene-Expression Data with Repeated Measurements. *Genome Biology, 4*(5), R34.

Yi, B.-K., & Faloutsos, C. (2000). Fast time sequence indexing for arbitrary Lp norms. *Proceedings of VLDB,* 385-394. Cairo, Egypt.

Yi, B.-K., Jagadish, H. V., & Faloutsos, C. (1998). Efficient retrieval of similar time sequences under time wrapping. *Proceedings of IEEE ICDE,* 201-208. Orlando, FL, USA.

Yi, H., Ao, X., & Ho, Y. (2008). Use of citation per publication as an indicator to evaluate pentachlorophenol research. *Scientometrics, 75,* 67-80.

Yiu, M. L., Dai, X., Mamoulis, N., & Vaitis, M. (2007). Top-k Spatial Preference Queries. *In Proceedings of the 23rd IEEE International Conference on Data Engineering (ICDE 2007),* Istanbul, Turkey, 15-20 April, 2007, (pp. 1076-1085). IEEE Computer Society Press.

Yiu, M.L., Mamoulis, N., & Papadias, D. (2005). Aggregate nearest neighbor queries in road networks. *IEEE Trans. Knowl. Data Eng., 17*(6), 820–833.

Yiu, M.L., Papadias, D., Mamoulis, N., & Tao, Y. (2006). Reverse nearest neighbors in large graphs. *IEEE Trans. Knowl. Data Eng., 18*(4), 540–553.

Yong, H.S., Lee, S. and Kim, H.J. (1994). Applying Signatures for Forward Traversal Query Processing in Object-Oriented Databases. *Proc. of 10th Int. Conf. on Data Engineering,* Houston, Texas, 518-525.

Yoo, J. S., & Shekhar, S. (2005). In-route nearest neighbor queries. *GeoInformatica, 9*(2), 117-137.

Yoon, J., Raghavan, V., Chakilam, V., & Kerschberg, L. (2001). BitCube: A three-dimensional bitmap indexing for XML documents. *Journal of Intelligent Information Systems, 17*(1), 241-252.

Young, F., & Thurstone, L. (1996). *A Cognitive Model of Data Analysis and its Implementation as a Human-Computer Interface*. Psychometric Laboratory, University of North Carolina, Chapel Hill, North Carolina.

Yu, C., & Popa, L. (2005). Semantic Adaptation of Schema Mappings when Schemas Evolve. *Proceedings of the 31st International Conference on Very Large Data Bases*, Trondheim, Norway, ACM.

Yuan, Y., & Zhang, J. J. (2003). Towards an appropriate business model for m-commerce. *International Journal of Mobile Communications, 1*(1/2), 35-56.

Yui, M. L., Mamoulis, N., & Papadias, D. (2005). Aggregate Nearest Neighbor Queries in Road Networks. *IEEE Transactions Knowledge Data Engineering, 17*(6): 820-833.

Yunker, J. (2003). *Beyond borders: Web globalization strategies.* Boston: New Riders.

Zait, M., & Messatfa, H. (1997). A comparative study of clustering methods. *Future Generation Computer Systems, 13*(2-3), 149-159.

Zaki M. J. & Hsiao C.–J. (1999). Charm: An Efficient Algorithm for Closed Association Rule Mining. Computer Science Dept., Rensselaer Polytechnic Institute. Technical Report 99–10.

Zaki M. J. & Hsiao C.–J. (2005). Efficient Algorithms for Mining Closed Itemsets and their Lattice Structure. IEEE Transactions on Knowledge and Data Engineering, 17(4), 462-478.

Zaki M. J. (2000). Generating Non-Redundant Association Rules. 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 34-43.

Zaki, M. J. (2001). SPADE: An Efficient Algorithm for Mining Frequent Sequences, *Machine Learning, 42*(1/2), 31-60.

Zambonelli, F., & Omicini, A. (2004). Challenges and Research Directions in Agent-Oriented Software Engineering. *Autonomous Agents and Multi-Agent Systems, 9*(3), 253-283.

Zamboulis, L., & Poulovassilis, A. (2006). Information sharing for the Semantic Web — A schema transformation approach. *In Proceedings of* DISWeb06, *CAiSE06 Workshop Proceedings,* 275-289.

Zamir, O., & Etzioni, O. (1998). *Web document clustering: a feasibility demonstration*. Paper presented at SIGIR '98: the 21st annual international ACM SIGIR conference on Research and development in information retrieval, ACM, 1998, 46-54.

Zamir, O., Etzioni, O., Madani, O., & Karp, R. M. (1997). Fast and intuitive clustering of web documents. In *Proc. of SIGKDD-97, 3rd Int. Conf. on Knowledge Discovery and Data Mining* (pp. 287–290), Newport Beach, USA.

Zarri, G. P. (2005). An *n*-ary language for representing narrative information on the Web. In *SWAP 2005, Semantic Web Applications and Perspectives – Proceedings of the 2nd Italian Semantic Web Workshop* (CEUR-WS.org/Vol-166), Boquet, P., and Tummarello, G., eds. Aachen: Sun SITE Central Europe (http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-166/63.pdf).

Zeinalipour-Yazti, D., Kalogeraki, V., & Gunopulos, D. (2004). Information retrieval techniques for peer-to-peer networks. *IEEE CiSE Magazine,* 12-20.

Zeinalipour-Yazti, D., Kalogeraki, V., & Gunopulos, D. (2005). Exploiting locality for scalable information retrieval in peer-to-peer systems. *Information Systems, 30*(4), 277-298.

Zelasco, J. F. (2002). Contrôle de qualité des modèles numériques des bases de données géographiques. *Journal XYZ 90,* Association Française de Topographie, 50-55.

Zelasco, J. F. (2002). Geographical database integrity. International Conference on Computer Science, Software Engineering, Information Technology. *e-Business and Applications (CSITeA-02).* Foz do Iguazu, Brazil.

Zelasco, J. F. (2005). SAR Interferometry: DEM Quality Control Method. Assestement and Applications. *Congreso 6ª Semana Geomática Barcelona del 8 al 11 de febrero* 2005.

Zelasco, J. F., & Donayo, J. (2007). *Control de Calidad Geométrica de MNT obtenidos por IFSAR por medio del PDEM*. Contribuciones a la geodesia aplicada, in press, Instituto de Geodesia de la Facultad de Ingeniería de la Universidad de Buenos Aires.

Zelasco, J. F., & Ennis, K. (2000) *Solución de un problema en la estimación de variancias en un caso especial en el que no se pueden aplicar estimadores habituales.* XXVIII Coloquio Argentino de estadística, Posadas, Misiones, Argentina.

Zendulka, J. (2006). Object-Relational Modeling in UML. In L. C. Rivero, J. H. Doorn, & V. E. *Ferragine (Eds.). Encyclopedia of database technologies and applications* (pp. 421-426). Idea Group Reference.
Zhang Y. et al. (2007). ICEDB: Intermittently-Connected Continuous Query Processing. *In Proceedings of ICDE.*

Zhang, D., Du, Y., Xia, T., & Tao, Y (2006). Progressive Computation of the Min-Dist Optimal-Location Query. *In Proceedings of the Very Large Data Bases Conference (VLDB 2006),* Seoul, Korea, 12-15 September, 2006, (pp. 643-654). Morgan Kaufmann, San Francisco.

Zhang, J., & Wang, H. (2006). Detecting outlying subspaces for high-dimensional data: The new task, algorithms and performance. *Knowledge and Information Systems: An International Journal (KAIS).*

Zhang, J., Hsu, W., & Lee, M. L. (2005). Clustering in dynamic spatial databases. *Journal of Intelligent Information Systems, 24*(1), 5-27.

Zhang, J., Lou, M., Ling, T. W., & Wang, H. (2004). HOS-miner: A system for detecting outlying subspaces of high-dimensional data. *Proceedings of the 30th International Conference on Very Large Data Bases (VLDB'04)* (pp. 1265-1268).

Zhang, J., Mamoulis, N., Papadias, D., & Tao, Y. (2004). All-Nearest-Neighbors Queries in Spatial Databases. *In Proceedings of the 16th Scientific and Statistical Database Management Conference (SSDBM 2004),* Santorini Island, Greece, 21-23 June, 2004, (pp. 297-306), IEEE Computer Society Press.

Zhang, K., & Shasha, D. (1989). Simple fast algorithms for the editing distance between trees and related problems. *SIAM Journal on Computing, 18*(6), 1245-1262.

Zhang, K., Statman, R., & Shasha, D. (1992). On the editing distance between unordered labeled trees. *Information Processing Letters, 42*(3), 133-139.

Zhang, T., Ramakrishnan, R., & Livny, M. (1996). *BIRCH: an efficient data clustering method for very large databases.* Paper presented at SIGMOD'96: the 1996 ACM SIGMOD international conference on Management of data, New York, NY, USA.

Zhang, X., Berry, M. W., & Raghavan, P. (2001). Level search schemes for information filtering and retrieval. *Information Processing and Management, 37*(2), 313-334.

Zhang, Z. (2005). *Ontology query languages for the semantic Web: A performance evaluation* [master's thesis]. Athens, GA: University of Georgia, Athens.

Zhao, Y. B., Kubiatowicz, J., & Joseph, A. (2001). *Tapestry: An infrastructure for fault-tolerant wide-area location and routing* (Tech. Rep. No. UCB/CSD-01-1141). University of California, Berkeley, Computer Science Division.

Zhao, Y., & Song, J. (2003). *AGRID: An Efficient Algorithm for Clustering Large High-Dimensional Datasets.* Paper presented at the PAKDD'03: 7th Pacific-Asia Conference on Knowledge Discovery and Data Mining, Seoul, Korea.

Zheng, S., Wen, J., & Lu, H. (2003). Cost-Driven Storage Schema Selection for XML. In *DASFAA'03: Proceedings of the 8th International Conference on Database Systems for Advanced Applications*, (pp. 337–344), Kyoto, Japan. IEEE Computer Society.

Zhong, S., & Ghosh, J. (2003). A unified framework for model-based clustering. *Journal of Machine Learning Research, 4*, 1001-1037. MIT Press.

Zhong, S., & Ghosh, J. (2005). Generative model-based document clustering: a comparative study. *Knowledge and Information Systems, 8*(3), 374-384. Aggarwal, C. C., Han, J., Wang, J., & Yu, P. S. (2003). *A framework for clustering evolving data streams.* Paper presented at the 29th international conference on Very Large Data Bases, Berlin, Germany.

Zhong, S., Yang, Z., & Wright, R. N. (2005). Privacy-enhancing *k*-anonymization of customer data. *Proceedings of the 24th ACM Symposium on Principles of Database Systems (PODS'05)* (pp. 139-147).

Zhou, X., Geller, J., Perl, Y., & Halper, M. (2006). An application intersection marketing ontology. Theoretical computer science: Essays in memory of Shimon Even. *Lecture Notes in Computer Science, 3895,* 143-153. Berlin: Springer-Verlag.

Zhou, Y., & Kontogiannis, K. (2003). *Incremental transformation of procedural systems to object-oriented platform.* Proceedings of COMPSAC, Dallas, TX.

Zhu, C., Kitagawa, H., & Faloutsos, C. (2005). Example-based robust outlier detection in high dimensional datasets. *Proceedings of the 2005 IEEE International Conference on Data Mining (ICDM'05)* (pp. 829-832).

Zhu, C., Kitagawa, H., Papadimitriou, S., & Faloutsos, C. (2004). OBE: Outlier by example. *Proceedings of the 8th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD'04)* (pp. 222-234).

Zhu, X., Cao, H., & Yu, Y. (2006). SDQE: Towards automatic semantic query optimization in P2P systems. *Information Processing and Management, 42*(1), 222-236.

Zhu, Y., & Shasha, D. (2002). StatStream: Statistical monitoring of thousands of data streams in real time. *Proceedings of the 28th International Conference on Very Large Databases* (pp. 358-369).

Zien, A., Ratsch, G., Mika, S., Scholkopf, B., Lengauer, T., & Muller, R.-K. (2000). Engineering Support Vector Machine Kernels that Recognize Translation Initiation Sites. *Bioinformatics, 16*(9), 799-807.

Zilio, D. C., Rao, J., Lightstone, S., Lohman, G. M., Storm, A., Garcia-Arellano, C., et al. (2004). DB2 design advisor: Integrated automatic physical database design. *30th International Conference on Very Large Data Bases (VLDB 2004)* (pp. 1087-1097).

Zobel, J., & Moffat, A. (2006). Inverted Files for Text Search Engines. *ACM Computing Surveys, 38*(2), Article 6.

Zoumboulakis, M., Roussos, G., & Poulovassilis, A. (2004). Asene: Active Sensor Networks. *International Workshop on Data Management for Sensor Networks VLDB 2004,* Toronto, Canada, 30 August.

Zuikeviciute, V., & Pedone, F. (2008). Conflict-Aware Load-Balancing Techniques for Database Replication. *ACM Symposium on Applied Computing*, (pp. 2169-2173).

Zumpano, E., Greco, S., Trubitsyna, I., & Veltri, P. (2004). On the semantics and expressive power of datalog-like languages for NP search and optimization problems. *ACM Symposium on Applied Computing* (pp. 692-697).

Zurek, T., & Sinnwell, M. (1999). *Data warehousing has more colours than just black and white*. Proceedings of the VLDB'99, Edinburgh, Scotland, 726–728.

Index

D

E

F

G

H

I

K

L

M

N

O

P

Q

R

S

T

U

V

W

X