Integrity Constraints over Association Rules*

Artur Bykowski[1], Thomas Daurel[1,2], Nicolas Méger[1], and Christophe Rigotti[1]

[1] Laboratoire d'Informatique de Recherche
en Image et Systèmes d'information (LIRIS)
Bâtiment Blaise Pascal, INSA Lyon, 69621 Villeurbanne Cedex, France
[2] Etudes et Productions Schlumberger 1, rue Henri Becquerel,
92142 Clamart Cedex, France

Abstract. In this paper, we propose to investigate the notion of integrity constraints in inductive databases. We advocate that integrity constraints can be used in this context as an abstract concept to encompass common data mining tasks such as the detection of corrupted data or of patterns that contradict the expert beliefs. To illustrate this possibility we propose a form of constraints called *association map constraints* to specify authorized confidence variations among the association rules. These constraints are easy to read and thus can be used to write clear specifications. We also present experiments showing that their satisfaction can be tested in practice.

1 Introduction

Integrity constraints are a central notion in databases used primarily to ensure data consistency. It has shown to be a fruitful and useful concept, with important additional benefits to guide very different aspects such as design, implementation and also query optimization (see [21,1] for an overview).

Basically, integrity constraints are an abstract specification of the possible contents of the database with respect to our current knowledge of the data domain. They have been deeply investigated in the context of relational databases as well as object-oriented databases, according to various objectives (e.g., specifications, efficient checking).

Recently, the concept of inductive database (IDB) has emerged [13,15,8], promoting the vision that a database dedicated to data mining contains not only data (e.g., customer transactions) but also all patterns that hold in the data (e.g., association rules [2]). Ideally, the user of an IDB can query data and patterns within a single language and can also express operations involving both data and patterns. The collection of patterns may be several orders of magnitude larger than the set of data itself, and thus cannot be materialized in general in the IDB. However from the user point of view, each pattern that holds should be considered as available for querying.

* This research is partially funded by the European Commission IST Programme - Accompanying Measures, AEGIS project (IST-2000-26450).

R. Meo et al. (Eds.): Database Support for Data Mining Applications, LNAI 2682, pp. 306–323, 2004.

We advocate in this paper that integrity constraints are also a very promising concept in IDB. Like in the classical database frameworks, they can be used for specifying data consistency and for rejecting inconsistent updates. However, in the context of IDB, integrity constraints raise new interesting and challenging issues, if we consider that they can also be applied to the patterns that hold in the data. Regarding this view, they can be used to specify what knowledge the designer (or an expert) considers to be reasonable to find in the data. Then, the violation of a *pattern integrity constraint* may be seen as an evidence of various phenomena. For example, if we have an IDB containing alarm logs with a daily insertion of a batch of new logs, then if after such an update one of the pattern integrity constraints is no longer satisfied this may highlight that this set of log records has not been properly cleaned. In this case the IDB engine can abort and undo the insertion, and let the IDB administrator (or a user) check these new logs. Another useful possibility is to consider that the designer/expert specifies intensionally with pattern integrity constraints the set of patterns that in her/his opinion could be found in the data. This provides a way to delimit the acceptable laws that could hold with respect to the knowledge that the designer/expert has about the domain. In this case an integrity violation can be assimilated to the occurrence of an unexpected phenomenon and the patterns violating that constraint can be considered as subjectively interesting piece of information for the designer/expert. Obviously, in the context of a multi-user IDB, such integrity constraints on patterns can be customized by each user, so that she/he can add more specific constraints than the ones set by the designer, to reflect her/his own belief and background knowledge.

The detection of corrupted data and the identification of new interesting knowledge among the extracted patterns are common tasks in data mining (see for example the classification of actions proposed by [19] when beliefs are contradicted). The idea, that we want to point out in this paper, is that large parts of these processes can be incorporated nicely in the IDB framework by means of integrity constraint specification and checking.

Of course, most forms of integrity constraints proposed previously in the database domain can be reused to specify the contents of an IDB in terms of tuples or objects (e.g., functional dependency, class inclusion hierarchy). And these constraints can be used directly to specify the data that are admissible in an IDB but also to specify the admissible patterns themselves, when these patterns are encoded as tuples or objects. So, at first sight, we can imagine to choose one of the very expressive languages already proposed in the literature (e.g., using a data manipulation language itself [21]) and use it to specify a large class of constraints over the patterns. The drawback of this approach is that it does not take into account the tradeoff between expressivity and computational complexity of constraint checking in the context of IDB.

For example using a Datalog like language with a polynomial evaluation complexity (w.r.t. the number of tuples in the database) to express constraints may be reasonable in a relational database, but will be in general not applicable in practice for IDB. The reason is that the number of patterns *stored* in an IDB

(in a materialized way or not) is for common families of patterns inherently exponential with respect to the pattern domain parameters. For instance, if we consider patterns called *frequent itemsets*, they are defined w.r.t. a set of binary attributes $\mathcal{A}$, and a frequent itemset may be any subset of $\mathcal{A}$, leading in the worst case to a collection of $2^{|\mathcal{A}|}$ patterns. Even if this number can remain reasonable in some practical cases (e.g., using a high frequency threshold on a sparse data set), when we set more difficult conditions (e.g., a lower frequency), all practitioners have had to deal with the problem of the exponential growth of the number of frequent itemsets extracted. The same problem can be illustrated on other commonly used patterns (e.g., association rules [2], frequent Datalog patterns [11]). So, we cannot expect to be able to apply a general integrity checking process (even one ensuring a polynomial evaluation complexity) on this set of patterns of exponential size.

The situation can be even worse since in most cases, in an IDB these patterns are not fully materialized, and thus some extra (in general non-polynomial) computation is needed to enumerate them.

In the context of IDB we propose to investigate the notion of integrity constraint for patterns, by taking advantage of the following observation. In IDB each pattern is an expression of a specific pattern domain with its own semantics and thus could come with its specific family of integrity constraints, offering an acceptable tradeoff between expressivity and evaluation cost.

In the rest of the paper we focus on a very common pattern called *association rule* [2] and we propose a dedicated form of integrity constraints called *association map constraints*.

Association rules were proposed to represent dependencies between the occurrences of items in customer transactions. Originally, the form of these rules was $A_1, A_2, A_3, \ldots \Rightarrow B$ where $A_1, A_2, A_3, \ldots$ and B denote items. The left hand side is called the *antecedent*, and the right hand side the *consequent*[1]. A confidence measure is defined for these rules. The value of the confidence could be considered as the conditional probability of having the consequent in a transaction when we have all items of the antecedent. Another quality measure, called relative support, is generally associated to the rules. A 10% relative support for a rule means that 10% of the observed transactions support the rule, i.e., the items (antecedent and consequent) could be observed together in 10% of the transactions. It should be noticed that mining association rules is not restricted to basket data analysis, and has been applied on many kinds of data sets after an appropriated encoding with Boolean variables (e.g., [20]). Association rules have received a lot of attention and several algorithms (e.g.,[18,3,12]) have been designed to extract them for given confidence and support thresholds.

An association map is an abstract specification of the set of association rules that could hold in the data according to our current knowledge of the data do-

[1] Consequents made of several items are also considered in the literature. The notion of association map constraint proposed in this paper can be adapted easily to this other form.

main. Association maps are a good candidate of dedicated forms of integrity constraints since they are very concise and readable in the following sense. Firstly, a small association map is sufficient to constrain a huge collection of association rules. Secondly, an association map has a strong hierarchical structure enabling quick intuitive browsing while its semantics remain very simple. And finally, another interesting property of association maps for their use as integrity constraints is that their satisfaction can be checked in a reasonably efficient way in practice.

The rest of this paper is organized as follows. In Section 2 we informally present the notion of association map constraint. More formal definitions and an algorithm to compute association maps are given in Section 3. In Section 4 we describe experiments showing that these constraints can be checked efficiently in practice even in difficult cases. We review related work and conclude with a summary in Section 5.

2 Informal Presentation

In this section we introduce in an informal way the notion of integrity constraint based on association map for IDB.

The key idea behind association map is to represent what should be the confidence variation if a particular item is added to or removed from the antecedent of a rule. Let us take a toy example where each transaction in the data set of the IDB describes one person involved in a car crash (her/his characteristic, the context, the damages).

We suppose that the designer has some knowledge in the car crash domain and wants to use it as integrity constraints over the association rules she/he thinks that could reasonably hold in the IDB. We make the hypothesis that there is a wide variety of such knowledge that can be expressed as the variation of rule confidence w.r.t. the presence/absence of a particular attribute in the left-hand side of the rule. For example, consider that for car crashes the expert thinks that the use of an airbag reduces the probability of severe injury, except for persons that wear glasses. This opinion can be seen as a constraint (denoted IC_1 below) on the variation of the confidence of rules concluding on *severe injury*, w.r.t. a *variation criterion* which is the presence/absence of an airbag.

Consider the following association rules that hold (among others) in the current instance of our IDB. For each rule we indicate the corresponding confidence, and one can easily see that this set of rules does not contradict the integrity constraint IC_1 set by the designer.

$\emptyset \Rightarrow$	$severe\,injury$ 20%
$airbag \Rightarrow$	$severe\,injury$ 10%
$driver \Rightarrow$	$severe\,injury$ 18%
$driver, airbag \Rightarrow$	$severe\,injury$ 10%
$wear\,glasses \Rightarrow$	$severe\,injury$ 15%
$wear\,glasses, airbag \Rightarrow$	$severe\,injury$ 20%
$wear\,glasses, driver \Rightarrow$	$severe\,injury$ 20%
$wear\,glasses, driver, airbag \Rightarrow$	$severe\,injury$ 25%

An association map is simply an explicit synthetic representation of these confidence variations in terms of effects of the presence/absence of the variation criterion in rule antecedents. A map is defined for a given consequent (e.g., *severe injury*), for a particular item used as variation criterion (e.g., *airbag*) and for a given support threshold, but without any confidence threshold. It contains a set of *regions*, where each region is characterized by an homogeneous effect on rule confidence when we add the variation criterion to the rule antecedent. The effect is called *positive* (resp. *negative*) if the addition of the variation criterion results in an increase (resp. a decrease) of the rule confidence. A region is delimited by a lower bound (w.r.t. set inclusion) which is a rule antecedent (a set of items) called *base*. Upward, a region is delimited by a *border* composed of the rule antecedents that are the minimal supersets of the base where the effect changes, from positive to negative or from negative to positive (neutral effects are not considered as real changes). And finally, all elements in a border of a region can be themselves the bases of new regions. Additionally, it should be noticed that rules having a support lower than the given threshold are not represented by the map.

The constraint IC_1 can be expressed as the association map depicted on Figure 1. If we consider all possible rule antecedents (excluding items *severe injury* and *airbag* that represent the rule consequent and the variation criterion), these antecedents can be organized in a lattice (w.r.t. set inclusion). This lattice is depicted using dashed lines on Figure 1. The bases and the border elements are simply particular elements of this lattice that delimitate the regions of homogeneous effects. For constraint IC_1 we have two such regions. The first having for base the empty set and as border $\{wear\_glasses\}$, and in which the effect is negative. And the second, with base $\{wear\_glasses\}$ and border $\{wear\_glasses, driver\}$ where the effect is positive. On the graphical representation, the space of all supersets of a base is sketched by a conic shape. The map presented on Figure 1 can be read as follows: If we add *airbag* to the antecedent of $\emptyset \Rightarrow severe\ injury$ then the confidence decreases, and this holds for all rules excepted if the antecedent contains *wear_glasses* in which case the confidence increases.

Suppose that a new set of transactions representing data related to pregnant women is inserted in the IDB, and that now we have the additional rules[2]:

$pregnant \Rightarrow$	$severe\,injury$ 30%
$pregnant, airbag \Rightarrow$	$severe\,injury$ 25%
$driver, pregnant \Rightarrow$	$severe\,injury$ 30%
$driver, pregnant, airbag \Rightarrow$	$severe\,injury$ 40%
$driver, pregnant, less\_than\_3\_month\_pregnancy \Rightarrow$	$severe\,injury$ 19%
$driver, pregnant, less\_than\_3\_month\_pregnancy, airbag \Rightarrow$	$severe\,injury$ 12%
$wear\,glasses, pregnant \Rightarrow$	$severe\,injury$ 35%

[2] To simplify the example we suppose that the previous rules still hold with the same confidence.

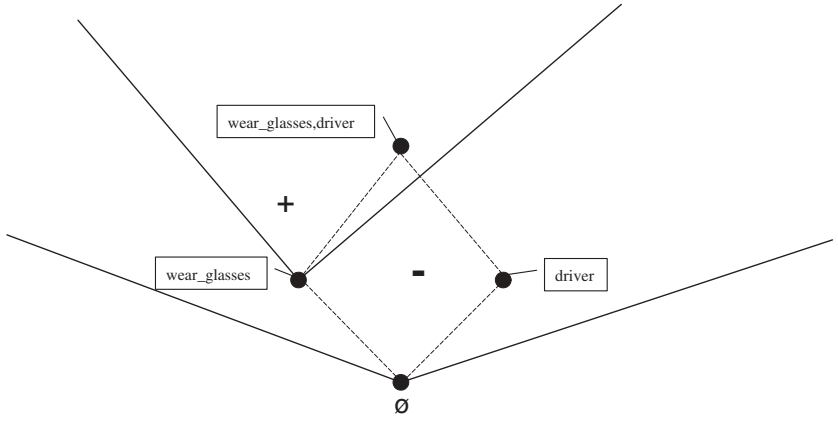

Fig. 1. Association map IC_1

$wear\ glasses, pregnant, airbag \Rightarrow$ $severe\ injury$ 40%
$wear\ glasses, driver, pregnant \Rightarrow$ $severe\ injury$ 35%
$wear\ glasses, driver, pregnant, airbag \Rightarrow$ $severe\ injury$ 42%
$wear\ glasses, driver, pregnant, less\_than\_3\_month\_pregnancy \Rightarrow$ $severe\ injury$ 23%
$wear\ glasses, driver, pregnant, airbag, less\_than\_3\_month\_pregnancy \Rightarrow$ $severe\ injury$ 28%

We recall that in the context of an IDB these rules are not necessarily extracted and materialized after the insertion of the new data, but from the user point of view they can be used/retrieved at any time.

If we have a close look at these rules, we notice that some of these patterns no longer satisfy IC_1. This can be seen more clearly by drawing the association map corresponding to the whole new set of association rules (for the consequent *severe injury*, the variation criterion *airbag*, and the same support threshold). This map is depicted on Figure 2, where, for readability reasons, the underlying lattice has not been represented. Testing if IC_1 is satisfied or not can then be performed by comparing this map to the map corresponding to IC_1. This is done on Figure 2, where the area of patterns that violate IC_1 is highlighted in grey.

In practice, the use of association maps as integrity constraints in an inductive database can be made as follows. First, the user or the database designer gives a collection of association maps to specify the authorized confidence variations in terms of known effects for specific variation criteria and rule consequents. After an update (or sequence of updates) of the data, the maps describing these effects are computed (from the data) by the inductive database system. Then, they are compared automatically to the ones that have been specified. If a difference is found, the system rejects the update(s) and presents this difference to the user (using eventually a graphical representation with highlighted areas as the one of Figure 2). The user can then assess whether the difference comes from a

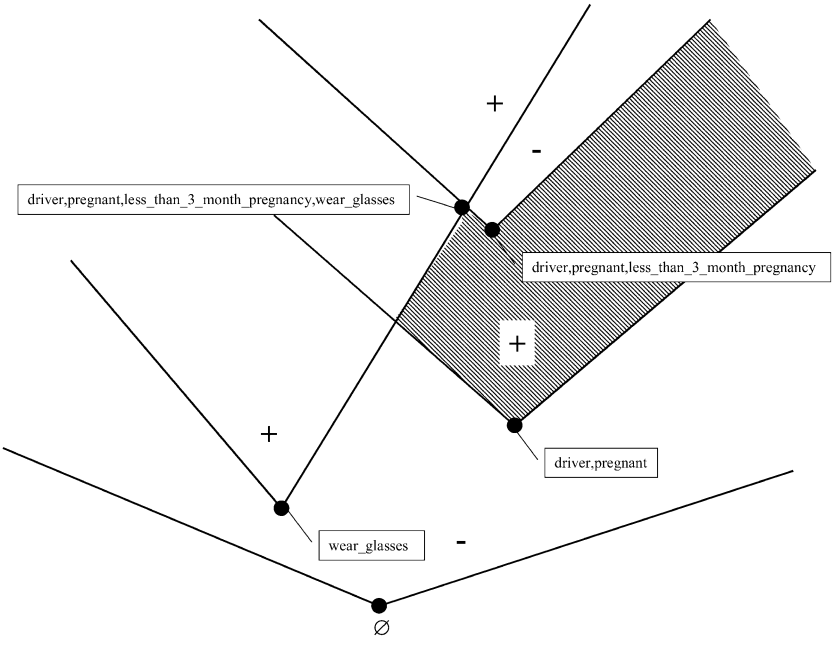

Fig. 2. Patterns that do not satisfy IC_1

corruption of the data or is due to effects that are not correctly specified by the integrity constraints. In the later case, this can leads to the modification of the integrity constraints and be a clue to find an unknown phenomenon.

2.1 Refinement of Maps

The notion of association map presented informally can lead on real data sets to many regions that are not appropriated. We introduce in this section two thresholds used to avoid such situations.

Discarding Extra Regions Using Strong Dependencies. Many exact or nearly exact association rules hold in real data sets and this phenomenon has been used recently to condense huge collections of itemsets [16,7].

Let us consider that we generate a map for consequent C and variation criterion H, and that we have between items A and B the association $A \Rightarrow B$ with a confidence of 100 % (such a rule can be due, for example, to a functional dependency holding in the data). Then $A \Rightarrow C$ and $A, B \Rightarrow C$ have the same confidence, and this is also true for rules $A, H \Rightarrow C$ and $A, B, H \Rightarrow C$. Thus, the effect of H on confidence is the same for antecedents $\{A\}$ and $\{A, B\}$. Moreover, the same holds for any pair of antecedents X and $X \cup \{B\}$, where X is a superset of $\{A\}$. This means that the portion of map generated for supersets of X is redundant with the part constructed for supersets of $X \cup \{B\}$.

Now, suppose that $\{B\}$ is the base of a region, and that the effect of H for $\{A, B\}$ is different than the effect of H on $\{B\}$. In this case, $\{A, B\}$ turns out to be in the border of the region based on $\{B\}$ and results in the generation of an extra region based on $\{A, B\}$. Since we know that the part of the map corresponding to supersets of $\{A, B\}$, is redundant with the one for supersets of $\{A\}$ we can avoid the construction of the extra region based on $\{A, B\}$. For any region based on a Y superset of $\{B\}$ we can make the same simplification when the region based on $Y \cup \{A\}$ has a different effect.

So, we can discard any base X if there exists $Y \subset X$ and $A \in X \setminus Y$ such that $Y \Rightarrow \{A\}$ with a 100% confidence (i.e., if there is an exact rule between items in X).

It should be noticed that exact rules are not likely to be found in noisy data sets or in presence of missing values. In these cases, they appear under the form of rules having a few number of exceptions. So, in the definitions given in Section 3.1 we discard a base X if there is a nearly exact rule between items in X. These nearly exact rules called δ-strong rules (rules with at-most δ exceptions) have been used previously in a different context [7] to condense collections of itemsets and mine frequent patterns more efficiently.

For association map extraction, δ will be a threshold called *freeness*.

Avoiding Regions Created as Artefacts

A confidence is the ratio s_1/s_2 of two integer support values. So, it cannot change in a continuous way, but only by discrete steps.

Let σ be the absolute support threshold used to generate the rules. The greatest discrete step variation due to a single row is encountered when confidence jumps from $(\sigma+1)/(\sigma+1)$ to $\sigma/(\sigma+1)$. Thus, a confidence variation lesser than $1 - \sigma/(\sigma+1)$ cannot be considered as really significative.

So, we use another threshold τ called *tolerance* to indicate what we consider as a clear confidence variation. When we add the item used as variation criterion to the antecedent of a rule, the variation of confidence must be strictly greater than τ (resp. strictly lower than $-\tau$) to be interpreted as a positive (resp. negative) effect. Otherwise the effect is said to be neutral.

The bases of regions are restricted to be such that their effects must be either positive or negative, except for the first base (the empty set) where the effect is also allowed to be neutral. Thus, in a region, when we encounter a neutral effect we consider that we are still in the same region, and it is only when we find a different and significative positive or negative effect that we generate a border element.

3 Computing Association Maps

In this section, we give more formal definitions and present a way to compute association maps.

3.1 Definitions

Preliminary

Definition 1 (Binary Database). Let R be a set of symbols called items. An *itemset* is a subset of R. A *binary database* r *over* R is a multiset of *rows*, where a row is an itemset. We use the notation $t \in r$ to denote that a particular row t belongs to r.

In this section, we assume that the data set is a binary database r over a set of items R.

Definition 2 (Itemset Support). We denote $\mathcal{M}(r, X) = \{t \in r | X \subseteq t\}$ the multiset of rows in r matched by the itemset X and $Sup(r, X) = |\mathcal{M}(r, X)|$ the *support* of X in r, i.e., the number of rows matched by X.

Definition 3 (Association Rule [2]). Let $Y \subseteq R$ be an itemset. An *association rule over* Y is an expression of the form $X \Rightarrow C$, where $C \in Y$ and $X \subseteq Y \setminus \{C\}$. The support of a rule in r is denoted $Sup(r, X \Rightarrow C)$ and is defined by $Sup(r, X \Rightarrow C) = Sup(r, X \cup \{C\})$. Its confidence is $Conf(r, X \Rightarrow C) = Sup(r, X \cup \{C\})/Sup(r, X)$.

We consider $\sigma \in (0, |r|]$ a support threshold. It should be noticed that it corresponds to an absolute number of rows. However to facilitate the reading of some examples we also use a relative support threshold, that simply corresponds to $\sigma/|r|$.

Definition 4 (Frequent Association Rules). A *frequent association rule* over R w.r.t. σ and r is an association rule $X \Rightarrow C$ over R, such that $Sup(r, X \Rightarrow C) \geq \sigma$. We denote $FreqRules(r, \sigma)$ the set of all frequent rules over R w.r.t. σ and r.

We also recall the definitions of δ-strong rules and $\delta - free$ sets, needed to define association maps. These two notions have been introduced in a different context[3] in [7,6].

Definition 5 (δ-Strong Rule). A *δ-strong rule*[4] in a binary database r is an association rule $X \Rightarrow C$ over R such that $Sup(r, X) - Sup(r, X \cup \{C\}) \leq \delta$, i.e., the rule is violated in no more than δ rows.

In this definition, δ is supposed to have a small value, so a δ-strong rule is intended to be a rule with very few exceptions.

Definition 6 (δ-Free Set). $X \subseteq R$ is a *δ-free set* w.r.t. r if and only if there is no δ-strong rule over X in r. The set of all δ-free sets w.r.t. r is noted $Free(r, \delta)$.

[3] Originally $\delta - free$ have been proposed as a condensed representation that can be extracted very efficiently and that can be used to closely approximate the support of all itemsets that are frequent w.r.t. a given support threshold.

[4] Stemming from the notion of *strong rule* of [17].

Since δ is supposed to be rather small, informally, a δ-free set is a set of items such that these items are not related by any very strong positive dependency.

Effect Regions and Association Maps. As presented in Section 2, an association map is defined w.r.t. two items (the consequent and the variation criterion) and three thresholds (support, tolerance and freeness). In this section, we denote respectively $C \in R$ the consequent and $H \in R$ the variation criterion, and we use $\sigma \in (0, |r|]$, $\tau \in [0, 1]$ and an integer δ to represent respectively the support, the tolerance and the freeness threshold.

Definition 7 (Local Effect). Let $X \subseteq R \setminus \{C, H\}$ be an itemset. The *local effect* of H on rule $X \Rightarrow C$, denoted $LocEffect(r, \tau, X, C, H)$ is defined as:

$$LocEffect(r, \tau, X, C, H) = \begin{cases} 1 & if \ \ Conf(r, X \cup \{H\} \Rightarrow C) - \\ & \qquad Conf(r, X \Rightarrow C) > \tau \\ -1 & if \ \ Conf(r, X \cup \{H\} \Rightarrow C) - \\ & \qquad Conf(r, X \Rightarrow C) < -\tau \\ 0 & otherwise \end{cases}$$

According to its value, the effect is respectively called *positive*, *negative* or *neutral*.

We now define the antecedents that are significative to generate the maps.

Definition 8 (Significant Antecedent). $SigAnte(r, \sigma, \tau, \delta, C, H)$ is the collection of significant antecedents and is defined by $SigAnte(r, \sigma, \tau, \delta, C, H) = \{X \subseteq R | (X \cup \{H\} \Rightarrow C) \in FreqRules(r, \sigma) \wedge X \in Free(r, \delta) \wedge LocEffect(r, \tau, X, C, H) \in \{-1, 1\}\}$.

These antecedents are itemsets that form with H the antecedent of a frequent rule. Moreover, they must be made of items that are not strongly dependent (i.e, they are δ-free) and where the local effect is clearly positive or negative (not neutral). Then, for an itemset X we define the border effect, which is the collection of the minimal supersets of X that are significant antecedents and where the local effect changes strongly (from positive to negative or from negative to positive).

Definition 9 (Effect Border). Let X be an itemset such that $X \subseteq R \setminus \{C, H\}$. The effect border for X is $Border(r, \sigma, \tau, \delta, X, C, H) = \{Y \in R | X \subset Y \wedge Y \in SigAnte(r, \sigma, \tau, \delta, C, H) \wedge LocEffect(r, \tau, X, C, H) \neq LocEffect(r, \tau, Y, C, H) \wedge (\forall Z, X \subset Z \subset Y \Rightarrow LocEffect(r, \tau, Z, C, H) \in \{0, LocEffect(r, \tau, X, C, H)\})\}$.

Now we consider regions of homogeneous effect, i.e., a set of rule antecedents having a common subset and a common local effect. We first define their lower bounds (w.r.t. set inclusion), called effect bases as follows. The empty set is an

effect base. A significant antecedent which is in the effect border of an effect base of smaller size is also an effect base. This notion is expressed more formally by the next definition.

Definition 10 (Effect Base). The collection of effect bases is defined inductively as follows.
$Base_0 = \{\emptyset\}$
$Base_i = \{X \subseteq R| \ |X| = i \wedge X \in SigAnte(r, \sigma, \tau, \delta, C, H) \wedge (\exists Y \in \bigcup_{j<i} Base_j, X \in Border(r, \sigma, \tau, \delta, Y, C, H))\}$

$Base(r, \sigma, \tau, \delta, C, H) = \bigcup_i Base_i$.

Then an association map is simply the collection of all effect bases together with their borders.

Definition 11 (Association Map). An association map for a binary database r w.r.t. items C, H and thresholds σ, τ, δ is defined by $AMap(r, \sigma, \tau, \delta, C, H) = \{\langle X, \mathcal{B}\rangle | X \in Base(r, \sigma, \tau, \delta, C, H) \wedge \mathcal{B} = Border(r, \sigma, \tau, \delta, X, C, H)\}$.

Each tuple in an association map corresponds to the lower and upper bounds (w.r.t. set inclusion) of a region where the local effect does not significatively change.

Note that two different effect regions may overlap. This overlapping may occur even when their respective effect bases have opposite local effects, in this case the itemsets that belong to both regions have a neutral local effect.

3.2 Algorithm

We present a generic algorithm called $GenMap$ to produce the map for a consequent C, a variation criterion H, and thresholds δ, σ and τ corresponding respectively to freeness, support and tolerance thresholds.

The algorithm calls three functions: $CandAnte, Signif$ and $Effect$.

The algorithm is presented using in its input a set $S = FreqRules(r, \sigma)$ of all frequent association rules along with their supports.

Algorithm 1 (*GenMap*)

Input: *C, H items, n the size of the largest candidate antecedent, set S, thresholds δ, σ and τ.*

Used subprograms: *$CandAnte(S, i, C, H)$ establishes the set of itemsets of size i, not containing C or H, that are candidates for being significant antecedents. $Signif(S, X, C, H, \tau, \delta, \sigma)$, which finds out whether X is a significant antecedent or not. And the function $Effect(S, X, C, H)$ is used to compute the local effect of H for X.*

Output: *a set of tuples containing all effect bases, and their corresponding effects and borders.*

1. **let** $E_\emptyset := Effect(S, \emptyset, C, H), Map := \{\langle \emptyset, E_\emptyset, \emptyset \rangle\}$;
2. **for all** $i \in \{1, \ldots, n\}$ **do**
3. **for all** $X \in CandAnte(S, i, C, H)$ **do**
4. **if** $Signif(S, X, C, H, \tau, \delta, \sigma)$ **then**
5. **let** $E_X := Effect(S, X, C, H)$;
6. **let** $MaxSubBases_X := \{\langle Y, E_Y, B_Y \rangle \in Map |$ $Y \subset X \wedge E_Y \neq E_X \wedge \forall W \in B_Y, W \not\subset X\}$;
7. **if** $MaxSubBases_X \neq \emptyset$ **then**
8. **let** $Map := Map \cup \{\langle X, E_X, \emptyset \rangle\}$;
9. **for all** $\langle Z, E_Z, B_Z \rangle \in MaxSubBases_X$ **do**
10. **let** $Map := (Map \setminus \{\langle Z, E_Z, B_Z \rangle\}) \cup$ $\{\langle Z, E_Z, B_Z \cup \{X\} \rangle\}$;
11. **od**
12. **fi**
13. **fi**
14. **od**
15. **od**
16. **output** Map

In line 1, $GenMap$ considers the empty itemset, which is always an effect base according to Definition 10. In line 2, the algorithm enters a loop corresponding to increasing sizes of candidate antecedents.

For each candidate antecedent X the algorithm checks if it is significant (line 4), and if so, $GenMap$ computes in line 6 the set $MaxSubBases_X$ of all bases of regions that contain X in their border.

If at least one of such region exists (line 7), then X is also an effect base, and the corresponding tuple is created in line 8. X is then stored in the borders of all regions having their bases in $MaxSubBases_X$ (lines 9–11).

Theorem 1 (Correctness of $GenMap$). *The algorithm GenMap outputs the effect bases (along with the corresponding border elements and effects) of the association map defined for a consequent C, a variation criterion H, and thresholds δ, σ and τ.*

Proof. *The proof is made by induction on the size of the bases. Note that the effect base $\emptyset$ is included in Map by the first line of the algorithm.*

Hypothesis. *Suppose that for every effect base X of size less or equal to i the algorithm GenMap correctly reported X as a base and as border element in Map.*

Consider an effect base $X \neq \emptyset$ of size $i + 1$. We are going to show that X is correctly reported as a base and as border element in Map.

X is an effect base implies that X is returned by $CandAnte(S, i, C, H)$ (line 3) and not filtered out by $Signif(S, X, C, H, \tau, \delta, \sigma)$ (line 4). Therefore, it will be considered in lines 5–12. By Definition 10, there is at least one effect base $Y \subset X$ such that X is in the border of the region of base Y. Assuming that the induction hypothesis holds, we find all such bases in line 6. Then, X is added as a base to Map in line 8, and correctly reported as border element in lines 9–11.

So, the algorithm correctly reported all effect bases and border elements in Map. The soundness of every update of Map is immediate. ◇

3.3 Computing Association Maps from Association Rules

GenMap can compute the association maps using as input S, the collection of all frequent rule, and running the functions $CandAnte(S, i, C, H)$, $Signif(S, X, C, H, \tau, \delta, \sigma)$ and $Effect(S, X, C, H)$ defined in the following manner.

$CandAnte(S, i, C, H)$ selects from S the rules having an antecedent of size i and consequent C, but skips the rules containing H in their antecedents. Then, it returns the collection of all antecedents of these rules.

$Signif(S, X, C, H, \tau, \delta, \sigma)$ checks if $X \cup \{H\} \Rightarrow C$ is in S (i.e., if the rule is frequent), and if it is the case it tests the local effect of H. To do so, it finds in S the rule $X \Rightarrow C$, and compares the confidences of the two rules. If the absolute value of their difference is less or equal to τ, the function exits returning $false$ (the local effect is neutral). Otherwise the δ-freeness of X is tested by simply checking that for every $A \in X$ the difference between the support of $X \setminus \{A\}$ and X is strictly greater than δ. It should be noticed that the supports of $X \setminus \{A\}$ and X can be obtained using S as follows. Let us consider that we need the support of a frequent itemset Z. Let B be any item such that $B \in Z$, then the rule $Z \setminus \{B\} \Rightarrow B$ is frequent and is in S. By definition 3 the support of Z is equal to the support of this rule.

If X is δ-free, $Signif(S, X, C, H, \tau, \delta, \sigma)$ returns $true$, and $false$ otherwise.

$Effect(S, X, C, H)$ finds the confidences of the rules $X \Rightarrow C$ and $X \cup \{H\} \Rightarrow C$ in S, and then returns the local effect of H according to the difference between the confidences of the two rules.

One can generate association maps using the generic algorithm and the collection of all frequent association rules. Unfortunately, this input collection may be very large. Moreover, for some data sets (e.g. highly correlated census-like data sets), it is an intractable process to mine all frequent association rules at interesting support thresholds.

In the next section, we show that one can avoid extracting all frequent rules, by using more elaborated input collections.

3.4 Computing Association Maps Directly

Let us now consider that S consists of all tuples $\langle Z \setminus \{H, C\}, Z \setminus \{H\}, Z \setminus \{C\}, Z \rangle$ such that Z is a frequent itemset (w.r.t. threshold σ) containing both C and H, and such that $Z \setminus \{H, C\}$ is δ-free. We also consider that we have at hand the supports of the itemsets in the tuples in S.

The main practical advantage of this new input S is that it remains in general many much more smaller than the set of all frequent association rules.

S can be used to generate the association map for consequent C, variation criterion H, thresholds δ, σ, τ, using algorithm $GenMap$ when the functions $CandAnte, Signif$ and $Effect$ are defined as follows.

$CandAnte(S, i, C, H)$ selects from all tuples in S the ones having a first element of size i and outputs these first elements. By grouping the tuples in S_i

according to the size of the first element at the time we construct S, we can compute the result of $CandAnte(S, i, C, H)$ in a very efficient manner.

$Signif(S, X, C, H, \tau, \delta, \sigma)$ looks for in S $\langle X, X \cup \{C\}, X \cup \{H\}, X \cup \{H, C\}\rangle$. If such a tuple exists in S then, by construction of S, X is δ-free and $X \cup \{H, C\}$ is frequent. Then, to verify that the local effect is not neutral for X, it compares the absolute value of the difference between $Sup(X \cup \{C\})/Sup(X)$ and $Sup(X \cup \{H, C\})/Sup(X \cup \{H\})$ to the tolerance threshold τ.

$Effect(S, X, C, H)$ used S to compute the local effect for X in the same way as function $Signif$. In fact, in an implementation of algorithm $GenMap$ the value of $Effect(S, X, C, H)$ is simply obtained during the computation of $Signif(S, X, C, H, \tau, \delta, \sigma)$.

The generation of S itself can be made using the algorithms presented in [7,6] to mine δ-free sets. In our prototype we choose to generate S using the technique proposed in [9,10] to mine frequent patterns efficiently even in presence of difficult dense data sets. The prototype extracted first a representation called *disjunction-bordered condensation* using the algorithm VLinEx proposed in [9,10] and then generates S from this representation. Finally, it produces the map itself using $GenMap$.

4 Experiments

To check the satisfaction of association map constraints we propose to first extract the corresponding association maps from the data of the IDB, and then to compare these maps with the association map constraints given by the designer of the IDB. We consider that the association map constraints are rather small, thus we neglect the computing cost of the second step and take only the first one into account. In this section, we report experiments showing that the first step (computation of association maps over the IDB) can be done efficiently even in difficult cases.

Conditions of Experiments. We choose *Pumsb*, a very challenging census data set, containing 7117 items, 49046 rows, each with 74 items set to *true*. The particularity of the selected data set is that it is very dense and the combinatorial explosion of the number of frequent itemsets makes the mining of all association rules intractable for low support thresholds [5]. This data set has been preprocessed by researchers from IBM Almaden Research Center[5].

We run experiments on a 1 GHz PC with 512 Mb of RAM and Linux operating system.

To produce difficult conditions for the association map extraction, we choose a value of δ equal to 0 (avoiding only regions due to exact dependencies), and $\tau = 10^{-4}$ (a tolerance close to the minimal tolerance threshold defined in Section 2.1). We also used a heuristic to select ten *hard* pairs consequent/variation criterion, i.e., pairs such that regions in maps tend to be large or numerous.

[5] http://www.almaden.ibm.com/cs/quest/data/ long_patterns.bin.tar

We present this heuristic and the results of the experiments in the following sections.

Selecting Consequents and Variation Criteria. We define a function to associate a score to each pair of consequent C and variation criterion H.

This score is computed from a collection of itemsets denoted $\mathcal{E}_{\sigma,\delta}$ and containing all δ-free itemsets having a support exceeding σ. Let $\mathcal{F}_{C,H}$ be the collection of itemsets in $\mathcal{E}_{\sigma,\delta}$ containing items C and H. Let nb_{neg} (resp. nb_{pos}) be the number of elements in $\mathcal{E}_{\sigma,\delta}$ having a negative (resp. positive) local effect for C, H with tolerance $\tau = 0$. Let M_{neg} (resp. M_{pos}) be the mean value of the confidence variation for all negative (resp. positive) local effects for C,H and $\tau = 0$.

We defined $score(C, H) = T_{CH} * N_{CH} * P_{CH} * abs(M_{neg}) * M_{pos}$.

The factor T_{CH} is $|\mathcal{F}_{C,H}|/|\mathcal{E}_{\sigma,\delta}|$, and represents the ratio of itemsets in $\mathcal{E}_{\sigma,\delta}$ that are candidates to be a base of a region.

The factor N_{CH} is $nb_{neg}/|\mathcal{F}_{C,H}|$ and corresponds to the ratio of negative local effects among all possible local effects. P_{CH} is $nb_{pos}/|\mathcal{F}_{C,H}|$ and corresponds to the same ratio for positive effects. $N_{CH} * P_{CH}$ is maximal when $N_{CH} = P_{CH} = 1/2$, i.e., when the amount of significant antecedents with negative and positive local effects for a given C and H are the same and there is no neutral-effect antecedents. High values of $N_{CH} * P_{CH}$ indicate that the map is likely to contain many changes of effects and thus many regions.

Finally, the factor $abs(M_{neg}) * M_{pos}$ takes into account the amplitude of the changes of the confidence. A higher value implies potential effect bases with clear positive or negative effects, and thus an important number of bases even at high values of the tolerance threshold.

A pair C, H having a high $score(C, H)$ offers a good potentiality of generating maps containing large and numerous regions.

Results. Figure 3 summarizes the results. For various support thresholds, we report the highest (MAX), the lowest (MIN) and the mean (MEAN) extraction time over the ten pairs consequent/variation criterion having the highest $score(C, H)$ values.

The experiments show that on this difficult data set, for support thresholds of 80% to 100%, the extraction of the maps from the data and thus the test of satisfaction of the association map constraints can be done in practice online (i.e., during interactive data manipulation sessions). For lower thresholds, the integrity check can be performed reasonably off-line even at a 50% support threshold, which represents very hard conditions on this dense data set[6].

In practice, a large amount of the map extraction time is spent to compute the association rules (or in our prototype, to generate the intermediate representation as presented in Section 3.4). It should be noticed that the computation

[6] Such conditions can be considered as much more difficult than lower support thresholds (e.g., 1% or even less) on many sparse data sets (e.g., basket data, logs of alarms).

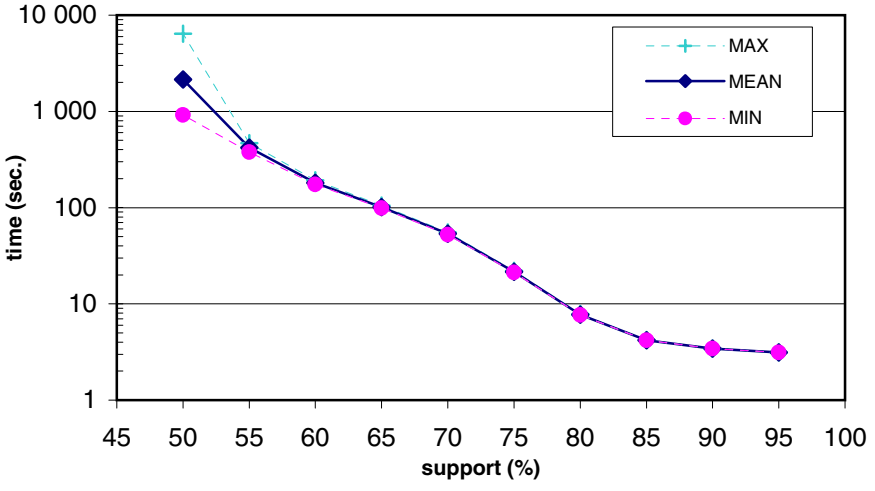

Fig. 3. Extraction times of association maps

of the association rules (or of the intermediate representation) is common to all maps for a given support threshold in a given data set. So, in most cases, when a map has been extracted for a pair consequent/variation, the maps for the other pairs involved in association map constraints can be obtained at a marginal extra cost.

5 Conclusion and Related Work

To our knowledge the notion of integrity constraint in IDB has not been previously explicitly investigated. In this paper we advocated that common data mining tasks such as the detection of corrupted data or of patterns that contradict the expert beliefs [19] can be integrated in a clean way under the concept of integrity constraints for IDB.

We illustrated this possibility by proposing a form of integrity constraints called association map constraints. Such a constraint is a specification of the sign of the variation of association rule confidences when a given attribute is added in the antecedent of the rule. These maps have a simple intuitive meaning and can concisely constrain all association rules. Thus, they allow to express clear and understandable specifications. Moreover, we have shown by means of experiments that the satisfaction of association map constraints can be checked in practice in a reasonably efficient way.

The use of confidence variation has been investigated previously in [4,14] to prune and summarize collection of association rules. As for association map these variations are considered w.r.t. a fixed rule consequent. [4] proposed to select rules $\alpha \Rightarrow C$ showing an increase (or eventually a limited decrease) of confidence with respect to all rules $\beta \Rightarrow C$ where $\beta \subset \alpha$ (i.e., more general rules). If we adapt this idea in the context of integrity constraints for IDB, it

leads to specify that some particular rules must have a confidence higher than any of their more general rules, and then to use the algorithm described in [4] to check if this specification is satisfied in the database. In [14] the authors proposed to select rules that are statistically more significant w.r.t. the more general rules and then to summarize this collection of selected rules. Similarly to [4] this approach can be adapted as integrity constraints for IDB.

Compared to these works, association maps are complementary. On one hand, they are more specific in the sense that an association map focuses on the effect of the absence/presence of a particular attribute H (the variation criterion) in the antecedent of the rules. However, it is possible to specify several association maps, each for a different attribute H. On the other hand, an association map is a cartography of all association rules (areas of decrease/increase of confidence w.r.t. the presence of H in the antecedent) and thus give a more general view than the approaches of [4] and [14] that concentrate on rules *better* (in some sense) than the more general ones.

With respect to the association map constraints proposed in this paper, an interesting issue to investigate, is to determine how and in which cases the check of the constraints can be performed incrementally with respect to the updates of the databases.

A more general direction of future work is to investigate how the concepts and techniques proposed previously in the data mining literature, can be adapted and used to specify and check the data and pattern consistency in the context of IDB.

Acknowledgments

We would like to thank the anonymous referees for their helpful comments and suggestions.

References

1. S. Abiteboul, R. Hull, and V. Vianu. *Foundations of Databases*. Addison-Wesley, 1995.
2. R. Agrawal, T. Imielinski, and A. N. Swami. Mining association rules between sets of items in large databases. In *Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data*, pages 207–216, Washington, D.C., USA, May 1993. ACM Press.
3. R. Agrawal, H. Mannila, R. Srikant, H. Toivonen, and A. I. Verkamo. Fast discovery of association rules. In *Advances in Knowledge Discovery and Data Mining*, pages 307–328. AAAI Press, 1996.
4. R. Bayardo, R. Agrawal, and D. Gunopulos. Constraint-based rule mining in large, dense databases. In *Proceedings ICDE'99*, pages 188–197, Sydney, Australia, March 1999.
5. R. J. Bayardo. Efficiently mining long patterns from databases. In *Proceedings of the 1998 ACM SIGMOD International Conference on Management of Data*, pages 85–93. ACM Press, 1998.

6. J.-F. Boulicaut, A. Bykowski, and C. Rigotti. Approximation of frequency queries by mean of free-sets. In *Proc. of the 4th European Conf. on Principles and Practice of Knowledge Discovery in Databases (PKDD'00)*, pages 75–85, Lyon, France, September 2000.
7. J.-F. Boulicaut, A. Bykowski, and C. Rigotti. Free-sets : a condensed representation of boolean data for the approximation of frequency queries. *Journal of Data Mining and Knowledge Discovery*, 7(1):5–22, 2003.
8. J.-F. Boulicaut, M. Klemettinen, and H. Mannila. Querying inductive databases: A case study on the MINE RULE operator. In *Proc. PKDD'98*, volume 1510 of *LNAI*, pages 194–202, Nantes, F, 1998. Springer-Verlag.
9. A. Bykowski and C. Rigotti. A condensed representation to find frequent patterns. In *Proc. of the Twentieth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems (PODS'01)*, pages 267–273, Santa Barbara, CA, USA, May 2001. ACM.
10. A. Bykowski and C. Rigotti. Disjunction-bordered condensed representation of frequent patterns. *Information Systems*, To appear.
11. L. Dehaspe and H. Toivonen. Discovery of frequent datalog patterns. *Journal of Data Mining and Knowledge Discovery*, 3(1):7–36, 1999.
12. J. Hipp, U. Güntzer, and G. Nakhaeizadeh. Algorithms for association rule mining – a general survey and comparison. *SIGKDD Explorations*, 2(1):58–64, July 2000.
13. T. Imielinski and H. Mannila. A database perspective on knowledge discovery. *Communications of the ACM*, 39(11):58–64, Nov. 1996.
14. B. Liu, W. Hsu, and Y. Ma. Pruning and summarizing the discovered associations. In *Proc. of the Fifth Int. Conference on Knowledge Discovery and Data Mining (KDD'99)*, pages 125–134, San Diego, CA, USA, August 1999.
15. H. Mannila. Inductive databases and condensed representations for data mining. In *Proc. ILPS'97*, pages 21–30, Port Jefferson, USA, 1997. MIT Press.
16. N. Pasquier, Y. Bastide, R. Taouil, and L. Lakhal. Efficient mining of association rules using closed itemset lattices. *Information Systems*, 24(1):25–46, 1999.
17. G. Piatetsky-Shapiro. Discovery, analysis, and presentation of strong rules. In *Knowledge Discovery in Databases*, pages 229–248. AAAI Press, Menlo Park, CA, 1991.
18. A. Savasere, E. Omiecinski, and S. B. Navathe. An efficient algorithm for mining association rules in large databases. In *Proc. VLDB'95*, pages 432–444, Zurich, Switzerland, September 1995. Morgan Kaufmann.
19. A. Silberschatz and A. Tuzhilin. On subjective measures of interestingness in knowledge discovery. In *Proc. of the First Int. Conference on Knowledge Discovery and Data Mining (KDD'95)*, pages 275–281, Montreal, Canada, August 1995.
20. R. Srikant and R. Agrawal. Mining quantitative association rules in large relational tables. In *Proc. ACM SIGMOD'96*, pages 1–12, Montreal, Quebec, Canada, June 1996. ACM Press.
21. J. Ullman. *Database and Knowledge-Base Systems, vol. II.* Computer Science Press, Rockville, MD, 1989.