Mining Borders of the Difference of Two Datacubes

Alain Casali

Laboratoire d'Informatique Fondamentale de Marseille (LIF)
CNRS UMR 6166, Université de la Méditerranée
Case 901, 163 Avenue de Luminy, 13288 Marseille Cedex 9, France
casali@lif.univ-mrs.fr

Abstract. In this paper we use the novel concept of minimal cube transversals on the cube lattice of a categorical database relation for mining the borders of the difference of two datacubes. The problem of finding cube transversals is a sub-problem of hypergraph transversal discovery since there exists an order-embedding from the cube lattice to the power set lattice of binary attributes. Based on this result, we propose a levelwise algorithm and an optimization which uses the frequency of the disjunction for mining minimal cube transversals. Using cube transversals, we introduce a new OLAP functionality: discovering the difference of two uni-compatible datacubes or the most frequent elements in the difference. Finally we propose a merging algorithm for mining the boundary sets of the difference without computing the two related datacubes. Provided with such a difference of two datacubes capturing similar informations but computed at different dates, a user can focus on what is new or more generally on how evolve the previously observed trends.

1 Introduction and Motivation

Hypergraph transversals [8, 10] have various applications in binary data mining and various kinds of knowledge can be discovered: minimal keys and minimal functional dependencies [13], connection between positive and negative borders of theories [14]. When mining minimal transversals, we show in [4] that the power set lattice is not really suitable when addressing multidimensional data mining problems, and suggest, as an alternative, an algebraic structure which is called cube lattice of a categorical database relation. Cube lattice is a set of tuples representing multidimensional patterns, provided with a generalization order between tuples. A similar lattice has been independently proposed by Lakshmanan *et al.* [12]. Based on this structure, the authors define the quotient cube lattice, a succinct summary of the datacube, preserving the Rollup/Drilldown semantics of cube.

Cube lattice provides a sound basis and a graded search space for extracting semantics from the datacube such as Roll-Up dependencies [3], multidimensional associations [5], iceberg cubes [2, 11], concise representation of hight frequency multidimensional patterns [6] and reduced cubes [12, 5]. In this paper, following

Y. Kambayashi et al. (Eds.): DaWaK 2004, LNCS 3181, pp. 391–400, 2004.

from this semantics trend, we use the concept of cube transversals [5] in order to compute the borders of the difference of two uni-compatible datacubes. More precisely, we make the following contributions.

- We propose an optimization for mining minimal cube transversals based on the Bonferroni inequalities and a levelwise algorithm which enforces the optimization.
- We formally define borders of the difference. Computing the difference of two uni-compatible datacubes is closely related to the discovery of emerging patterns originally proposed by [7] in the power set lattice framework. Our characterization and algorithm differ from the approach of emerging patterns since they are based on minimal cube transversals and fit in the cube lattice framework.

Because data warehouses are regularly refreshed, it is specially interesting to observe how evolves the mined knowledge: are great trends (for instance in consumer behavior) still increased or are they decreasing? Provided with two versions of a datacube computed at different dates, a user can answer these question very easily.

The remainder of the paper is organized as follows. Cube lattice framework [4] is described in section 2 as a graded search space for multidimensional data mining. In section 3, we recall the concept of cube transversal and a merging algorithm for mining minimal relation transversals from categorical database relations. In section 4, we propose an application using the minimal cube transversals: the "difference of two uni-compatible datacubes" and characterize its borders.

2 Background: The Cube Lattice Framework [4]

Throughout the paper, we make the following assumptions and use the introduced notations. Let r be a relation over the schema $\mathcal{R}$. Attributes of $\mathcal{R}$ are divided in two sets (i) $\mathcal{D}$ the set of dimensions, also called categorical or nominal attributes, which correspond to analysis criteria for OLAP, classification or concept learning and (ii) $\mathcal{M}$ the set of measures (for OLAP) or class attributes. Moreover, the set of attributes called $\mathcal{D}$ is totally ordered (the underlying order is denoted by $<_{\mathcal{D}}$) and for each attribut $A \in \mathcal{D}, Dim(A)$ stands for the projection of r over A.

The multidimensional space of the categorical database relation r groups all the valid combinations built up by considering the value sets of attributes in $\mathcal{D}$, which are enriched with the symbolic value ALL. The latter, introduced in [9] when defining the Cube-By operator, is a generalization of all the possible values of any dimension $(\forall A \in \mathcal{D}, \forall a \in Dim(A), \{a\} \subset \text{ALL})$.

The multidimensional space of r is noted and defined as follows: $Space(r) = \{\times_{A \in \mathcal{D}}(Dim(A) \cup \text{ALL})\} \cup \{<\emptyset, \ldots, \emptyset>\}$ where $\times$ symbolizes the Cartesian product, and $<\emptyset, \ldots, \emptyset>$ stands for the combination of empty values. Any combination belonging to the multidimensional space is a tuple and represents a multidimensional pattern.

Example 1. - Table 1 presents the categorical database relation used all along the paper to illustrate the introduced concepts. In this relation, Sky, AirTemp and Humidity are dimensions and EnjoySport is a class. The following tuples $t_1 = <S, W, \text{ALL}>$, $t_2 = <S, W, N>$, $t_3 = <S, C, N>$, $t_4 = <S, \text{ALL}, N>$, $t_5 = <S, \text{ALL}, \text{ALL}>$, $t_6 = <\text{ALL}, W, \text{ALL}>$ and $t_7 = <\text{ALL}, W, H>$ are elements of $Space(r)$.

Table 1. Relation example r

Sky	AirTemp	Humidity	EnjoySport
S	W	N	Yes
S	W	H	Yes
R	C	H	No

The multidimensional space of r is structured by the generalization order between tuples which is denoted by $\geq_g$. If $u \geq_g v$, we say that u is more general than v in $Space(r)$. In the multidimensional space of our relation example, we have: $t_5 \geq_g t_2$, i.e. t_5 is more general than t_2 and t_2 is more specific than t_5. Moreover any tuple generalizes the tuple $<\emptyset, \emptyset, \emptyset>$ and specializes the tuple $<\text{ALL}, \text{ALL}, \text{ALL}>$. When applied to a set of tuples, the operators *min* and *max* yield the tuples which are the most general ones in the set or the most specific ones respectively.

The two basic operators provided for tuple construction are: Sum (denoted by $+$) and Product (noted $\bullet$). The Semi-Product operator (noted $\odot$) is a constrained product used for the candidate generation in levelwise algorithms.

- The Sum of two tuples yields the most specific tuple which generalizes the two operands. If $t = u + v$, then we say that t is the Sum of the tuples u and v. In our example of $Space(r)$, we have $t_2 + t_3 = t_4$. This means that t_4 is built up from (or aggregates data of) the tuples t_2 and t_3.
- The Product of two tuples yields the most general tuple which specializes the two operands. If it exists, for these two tuples, a dimension A having distinct and real world values (i.e. existing in the original relation), then the only tuple specializing them is the tuple $<\emptyset, \ldots, \emptyset>$ (apart from it, the tuple sets which can be used to construct them are disjoined). If $t = u \bullet v$, then we say that t is the Product of the tuples u and v. In our example of $Space(r)$, we have $t_1 \bullet t_4 = t_2$. This means that t_1 and t_4 generalize t_2 and t_2 participates to the construction of t_1 and t_4 (directly or not). The tuples t_1 and t_3 have no common point apart from the tuple of empty values.

In some cases, it is interesting to know the attributes for which the values associated with a tuple t are different from the value ALL. This is why the function *Attribute* is introduced. In our example, we have $Attribute(t_1) = \{\text{Sky}, \text{AirTemp}\}$.

The Semi-Product operator is a constrained product operator useful for candidate generation in a levelwise approach [14]. Provided with multidimensional

patterns at the level i, we only generate candidates of level $i+1$, if they exist (else the constrained product yields $<\emptyset, \ldots, \emptyset>$). Moreover, each tuple is generated only once. In our example, we have $t_5 \odot t_6 = t_1$ and $t_5 \odot t_7 = <\emptyset, \ldots, \emptyset>$.

By providing the multidimensional space of r with the generalization order between tuples and using the above-defined operators Sum and Product, we define an algebraic structure which is called cube lattice. Such a structure provides a sound foundation for multidimensional data mining issues.

Theorem 1. *[4] - Let r be a categorical database relation over $\mathcal{D} \cup \mathcal{M}$. The ordered set $CL(r) = \langle Space(r), \geq_g \rangle$ is a complete, graded, atomistic and coatomistic lattice, called cube lattice in which Meet ($\bigwedge$) and Join ($\bigvee$) operators are given by:*

1. $\forall\ T \subseteq CL(r),\ \bigwedge T = +_{t \in T}\ t$
2. $\forall\ T \subseteq CL(r),\ \bigvee T = \bullet_{t \in T}\ t$

Through the following proposition, we characterize the order-embedding from the cube lattice to the powerset lattice of the whole set of attribute values. For avoiding ambiguities, we choose to prefix each value by the name of the concerned attribute.

Proposition 1 - *Let $\mathcal{L}(r)$ be the powerset lattice of the attribute value set, i.e. the lattice $\langle \mathcal{P}(\bigcup_{A \in \mathcal{D}} A.a, \forall a \in Dim(A)), \subseteq \rangle$ [1]. Then it exists an order-embedding:*

$$\Phi : CL(r) \rightarrow \mathcal{L}(r)$$

$$t \mapsto \begin{cases} \bigcup_{A \in \mathcal{D}} A.a, \forall a \in Dim(A) \text{ if } t = <\emptyset, \ldots, \emptyset> \\ \{A.t[A] \mid \forall A \in Attribute(t)\} \text{ elsewhere.} \end{cases}$$

Consequently, we can associate to each categorical relation r a binary relation $\Phi(r) = \{\Phi(t), \forall\ t \in r\}$. The latter can be used to apply binary data mining technics. Let us underline that Φ is not the single order-embedding Φ. We can construct other order-embedding by associating to each value a_i in the dimension of the attribute A a single binary attribute in the power set lattice of the attribute value set.

Finally, we have to highlight two important elements in the cube lattice: its atoms and coatoms. An atom of the cube lattice is a concept similar to the one-itemsets in the power set lattice (one-itemset are the atoms of the power set lattice). We denote by $\mathcal{A}t(CL(r))$ the atoms of the cube lattice (i.e. $\{t \in CL(r) : |\Phi(t)| = 1\}$). The atoms of a tuple t, denoted by $\mathcal{A}t(t)$, constitute the set of the atoms of the cube lattice which are more general than t (i.e. $\{t' \in \mathcal{A}t(CL(r)) : t' \geq_g t\}$). The coatoms of the cube lattice, denoted by $\mathcal{CA}t(CL(r))$, represent the concept dual to the one of atoms of the cube lattice (i.e. $\{t \in CL(r) : |\Phi(t)| = |\mathcal{D}|\}$).

[1] $\mathcal{P}(X)$ is the powerset of X.

3 Cube Transversals

When considering the power set lattice $\mathcal{L}(r)$ as the search space, a binary pattern X is a transversal of $\Phi(r)$ if and only if each transaction t of $\Phi(r)$ contains at least one value of X (X is a transversal of $\overline{\Phi(r)}$ if X is not a subset of any transactions t of $\Phi(r)$ respectively). For example, using our relation example and the order-embedding Φ, the binary pattern $\{Sky.S, AirTemp.W\}$ is a transversal of $\Phi(r)$. Moreover, this pattern is a minimal transversal of $\Phi(r)$ because none of its subsets is transversal of $\Phi(r)$. Unfortunately, in the binary framework, invalid multidimensional patterns are computed. For example, the binary pattern $\{Sky.S, Sky.R\}$ is a minimal transversal of $\Phi(r)$ but it is not a valid multidimensional pattern because each value of this pattern belongs to the very same attribute Sky. And we know that a multidimensional pattern can only encompasses a single value for any given attribute. This is why we have introduced [5] the concept of minimal cube transversal:

Definition 1. ***(Cube Transversal)*** *- We define the relation $\overline{r}$ as the difference of the coatoms of the cube lattice and the relation r ($\overline{r} = \mathcal{CA}t(CL(r))\backslash r$). Let $t \in CL(r)$ be a tuple, t is a cube transversal of r over $CL(r)$ iff $\forall t' \in r, t + t' \neq <ALL, \ldots, ALL>$. t is a cube transversal of $\overline{r}$ iff $\forall t' \in r, t \not\geq_g t'$.*

Lemma 1. *Since the contraint "t is a cube transversal of r (or $\overline{r}$)" is a monotone constraint on the cube lattice, the set of cube transversals is a convex space and thus it can be represented by its minimal border [4]. The set of minimal cube transversals of r and of $\overline{r}$ are denoted by $cTr(r)$ and $cTr(\overline{r})$ respectively and defined as follows:*

- $cTr(r) = \min_{\geq_g}(\{t \in CL(r) \mid \forall t' \in r, t + t' \neq <ALL, \ldots, ALL>\})$
- $cTr(\overline{r}) = \min_{\geq_g}(\{t \in CL(r) \mid \forall t' \in r, t \not\geq_g t'\})$

Due to the convex space property, an unseen tuple t is a transversal of r (or $\overline{r}$ respectively) if it exists at least a tuple t' in $cTr(r)$ ($cTr(\overline{r})$ resp.) which generalizes t. Let $\mathbb{A}$ be an anti-chain of $CL(r)$ (all tuples of $\mathbb{A}$ are not comparable using $\geq_g$). We can constrain the set of minimal cube transversals of r using $\mathbb{A}$ by enforcing each minimal cube transversal t to be more general than at least one tuple u of the anti-chain $\mathbb{A}$. The new related definitions are the following:

- $cTr(r, \mathbb{A}) = \{t \in cTr(r) \mid \exists u \in \mathbb{A} : t \geq_g u\}$
- $cTr(\overline{r}, \mathbb{A}) = \{t \in cTr(\overline{r}) \mid \exists u \in \mathbb{A} : t \geq_g u\}$

Example 2. By considering the multidimensional space of the relation example, the set of minimal cube transversals of r is $\{<S, C, \text{ALL}>, <S, \text{ALL}, H>, <R, W, \text{ALL}>, <\text{ALL}, W, H>\}$ and the set of minimal cube transversals of $\overline{r}$ is $\{<S, C, \text{ALL}>, <R, W, \text{ALL}>, <R, \text{ALL}, N>, <\text{ALL}, C, N>\}$. If we constrain the set $cTr(r)$ with the anti-chain composed by the single tuple $<S, C, H>$, we obtain $cTr(r, <S, C, H>) = \{<S, C, \text{ALL}>, <S, \text{ALL}, H>\}$.

3.1 Optimizing the Discovery of Minimal Cube Transversals

Assessing if a tuple t is whether a cube transversal of r or $\overline{r}$ can require $|r|$ evaluations. Minimizing the number of evaluations improves, in practice, the performance of the algorithm CTR. We introduce a new optimization based on the Bonferroni inequalities which are applied within the cube lattice framework. When the sum of the frequency of atoms of t is lower than 1, no condition is evaluated because t cannot be a cube transversal of r. A dual property is given for testing if t is a minimal cube transversal of $\overline{r}$.

Definition 2. ***(Frequency)*** *- Let $t \in CL(r)$ be a tuple, the frequency of t (denoted by $Freq(t, r)$) is the ratio between the number of tuples of r which are more specific than t and the number of tuples of r. Thus we have:*

$$Freq(t, r) = \frac{|\{t' \in r \mid t \geq_g t'\}|}{|r|}$$

Proposition 2 *Let $t \in CL(r)$ be a tuple, if t is a cube transversal of r then $\sum_{t' \in \mathcal{A}t(t)} Freq(t', r) \geq 1$ and if t is a cube transversal of $\overline{r}$ then $\sum_{t' \in \mathcal{A}t(t)} (1 - Freq(t', r)) \geq 1$.*

3.2 Finding Minimal Cube Transversals

In [10], it is shown that levelwise mining of minimal hypergraph transversals improves complexity results proposed by [8]. Using the cube lattice framework, we propose a levelwise algorithm called CTR (Cube TRansversal) algorithm which computes minimal cube transversals. The candidate generation step does not need any backtrack because we update the set cTr at each level and we use it for the pruning step. We improve our algorithm by a frequency-based optimization. A levelwise approach works very well when the underlying search space is a graded lattice, which is case of the cube lattice.

For mining minimal cube transversals of $\overline{r}$, we must replace conditions at lines 9 and 13 by $\sum_{t' \in \mathcal{A}t(t)} (1 - Freq(t', r)) \geq 1$ and $l \geq_g t$ respectively.

Complexity of CTR: The complexity of a levelwise algorithm for finding minimal transversals of an hypergraph $\mathcal{H}$ on a set E is $\mathcal{O}(2^k |E||Tr(\mathcal{H})|)$ where $k = \max(\{|X| : X \in \mathcal{H}\})$ [10]. Using the cube lattice as the search space, this complexity is preserved with $E = \bigcup_{A \in \mathcal{D}} A.a, \forall a Dim(A)$, $k = |\mathcal{D}|$ and $|cTr(r)| < |Tr(\mathcal{H})|$ since Φ is an order-embedding but not an order isomorphism.

4 Diff_Cube Operator

In ROLAP databases, the relational operator difference is fundamental for analyzing changes betwwen two uni-compatible datacubes (computed at different

Algorithm 1 CTR Algorithm

Input: Categorical database relation r over $\mathcal{D}$ [and an anti-chain $\mathbb{A}$]
Output: $cTr(r[, \mathbb{A}])$

1: $i := 1$; $cTr := \{\emptyset\}$
2: $C_1 := \{t \in \mathcal{A}t(CL(r))\}$
3: $L_1 := \{t \in C_1 \mid t \text{ is a cube transversal }\}$
4: $C_1 := C_1 \backslash L_1$
5: **while** $C_i \neq \emptyset$ **do**
6: $\quad cTr := cTr \cup L_i$
7: $\quad C_{i+1} := \{v = t \odot t' \mid t, t' \in C_i, v \neq <\emptyset, \ldots, \emptyset>, \nexists u \in cTr : u \geq_g v$ [and $\exists u \in \mathbb{A} : v \geq_g u]$ $\}$
8: $\quad$ **for all** $t \in C_{i+1}$ **do**
9: $\quad\quad$ **if** $\sum_{t' \in \mathcal{A}t(t)} Freq(t', r) < 1$ **then** $C^*_{i+1} := C^*_{i+1} \cup t$; $C_{i+1} := C_{i+1} \backslash t$
10: $\quad$ **end for**
11: $\quad$ **for all** $t \in r$ **do**
12: $\quad\quad$ **for all** unmarked $l \in C_{i+1}$ **do**
13: $\quad\quad\quad$ **if** $l + t = <\text{ALL}, \ldots, \text{ALL}>$ **then** mark l
14: $\quad\quad$ **end for**
15: $\quad$ **end for**
16: $\quad L_{i+1} := \{l \in C_{i+1} \mid l \text{ is unmarked }\}$; $C_{i+1} := (C_{i+1} \backslash L_{i+1}) \cup C^*_{i+1}$
17: $\quad i := i + 1$
18: **end while**
19: **return** cTr

instants by using the function COUNT) of two categorical database relations (denoted by r^+ and r^- for uniformity). However, the difference of the two datacubes is not the datacube of the difference of the two relations (datacube(r^+)\ datacube(r^-) $\neq$ datacube($r^+ \backslash r^-$)). We propose a merging algorithm which computes the boundary sets of tuples which are in the difference. Such tuples provide a condensed representation of (i) the difference of the two related datacubes or (ii) the most frequent elements in the difference.

4.1 Emerging Tuples

Let $minfreq \in]0, 1]$ a threshold given by the user, a tuple t is emerging if and only if it satisfies the two following constraints (C_1 and C_2):

- (C_1) $Freq(t, r^-) = 0$ and
- (C_2) $Freq(t, r^+) \geq minfreq$

The set $Diff\_Cube(r^+, r^-)$ encompasses all the emerging tuples of $r = r^+ \cup r^-$; *i.e.* $Diff\_Cube(r^+, r^-) = \{t \in CL(r) \mid t \text{ is emerging}\}$. Thus $Diff\_Cube(r^+, r^-)$ is exactly the difference of the two datacubes of the relations r^+ and r^-.

In RDBMSs provided with OLAP functionalities, such as IBM DB2 or Microsoft SQL Server, $Diff\_Cube(r^+, r^-)$ can be computed by using the Group By Cube operator [9] as follows:

SELECT A_1, ..., A_n
FROM r^+
GROUP BY CUBE (A_1, ..., A_n)
HAVING count(*) $\geq minfreq * |r^+|$
MINUS
SELECT A_1, ..., A_n
FROM r^-
GROUP BY CUBE A_1, ..., A_n;

We assume that $|r^+|$ and $minfreq$ are thresholds given by the user. Due to the underlying Cube-By operations, it is obvious that this query is specially time and space consuming, thus avoiding to evaluated it and instead computing the borders of its result can be of great interest.

4.2 Finding Borders

The constraint C_1 (C_2 resp.) is monotone (antimonotone resp.) w.r.t. $\geq_g$, consequently $Diff\_Cube(r^+, r^-)$ is a convex space of $CL(r)$ [4] and thus it can be represented by the sets (borders) $S = \max_{\geq_g}(Diff\_Cube(r^+, r^-))$ and $G = \min_{\geq_g}(Diff\_Cube(r^+, r^-))$.

The following proposition characterizes the borders of the Diff_Cube set:

Proposition 3 *Let $Diff\_Cube(r^+, r^-)$ be the set of emerging tuples of a categorical database relation $r = r^+ \cup r^-$ and $M = \max_{\geq_g}(\{t \in CL(r^+) \mid Freq(t, r^+) \geq minfreq\}$ then:*

1. $G = \{t \in cTr(\overline{r^-})$ *on* $CL(r^+) \mid Freq(t, r^+) \geq minfreq\}$
2. $S = \{t \in M \mid \exists t' \in G : t' \geq_g t\}$

With the following propositions, the borders S and G of $Diff\_Cube(r^+, r^-)$ can be computed in an efficient way and a merging algorithm can be designed to enforce such a computation.

Proposition 4 *Let $Diff\_Cube(r^+, r^-)$ be the difference of the datacube of r^+ and the one of r^- and $M = \max_{\geq_g}(\{t \in CL(r^+) \mid Freq(t, r^+) \geq minfreq\}$ thus: $\forall t \in cTr(r^+), t \in G \Rightarrow \nexists u \in M \backslash S : t \geq_g u$.*

Proposition 5 *Let $r = r_1 \cup r_2$ be a categorical database relation, $cTr(r_1)$ and $cTr(r_2)$ the minimal cube transversals of r_1 and r_2 respectively. If $cTr(r_1)$ is empty, we consider that $cTr(r_1) = \{<\emptyset, \ldots, \emptyset>\}$ (idem for $cTr(r_2)$), thus we have: $cTr(r) = \min_{\geq_g}(\{t \bullet t' \neq <\emptyset, \ldots, \emptyset> \mid t \in cTr(r_1)$ and $t' \in cTr(r_2)\})$.*

For finding S and G in our new context, we propose the merging algorithm Diff_Cube. Our algorithm includes the function Max_Set_Algorithm which discovers maximal frequent multidimensional patterns. It could be enforced by modifying algorithms such as Max-Miner [1]. The correctness of Diff_Cube algorithm is given by propositions 3, 4 and 5. Its complexity is similar to the complexity of the CTR algorithm.

Algorithm 2 Algorithm Diff_Cube

Input: $r^+, r^- = r_1^-, ..., r_p^- \neq \{\emptyset\}$
Output: S, G
1: $M :=$ Max_Set_Algorithm$(r^+, minfreq)$
2: $S = \{t \in M \mid t$ is a cube transversal of $\overline{r^-}$ on $CL(r^+)\}$
3: $G := \{<\emptyset, \ldots, \emptyset>\}$
4: **for** $i := 1$ to p **do**
5: $G' := \{t \in cTr(\overline{r_i^-}, S)$ on $CL(r^+)\}$ \\ use CTR algorithm
6: $G' := G' \backslash \{t \in G' \mid \exists u \in M \backslash S : u \geq_g t\}$
7: **if** $G' \neq \{\emptyset\}$ **then**
8: **for all** $t \in M$ **do**
9: **for all** tuple $t' \in G'$ unmarked **do**
10: **if** $t' \geq_g t$ **then** mark t'
11: **end for**
12: **end for**
13: $G' := \{t \in G' \mid t$ is marked $\}$
14: $G := \min_{\geq_g}(\{v = t \bullet t' \mid v \neq <\emptyset, \ldots, \emptyset>, t \in G$ and $t' \in G'\})$
15: **end if**
16: **end for**
17: **return** S, G

Example 3. - Let us consider $r^+ = \{t \in r \mid t[EnjoySport] = `Yes'\}$, the relation $r^- = \{t \in r \mid t[EnjoySport] = `No'\}$ and $minfreq = 1/2$. Then, we have:

- $M = \{<S, W, N>, <S, W, H>\}$
- $G = \{<S, \text{ALL}, \text{ALL}>, <\text{ALL}, W, \text{ALL}>, <\text{ALL}, \text{ALL}, N>\}$
- $S = \{<S, W, N>, <S, W, H>\}$

5 Conclusion

The presented work results from a cross-fertilization between the research fields of discrete mathematics, database and machine learning. We introduce the concept of the cube transversals of a categorical database relation. This concept is used to the problem of computing the difference of two uni-compatible datacubes, a new OLAP functionality which can provides users with a focus on new trends emerging from data sets collected at different points along the time. To the best of the author knowledge it is the first time that the problem of the difference of two uni-compatible datacubes (without computing the two cubes) is studied. Set operations on convex spaces of cube lattices are an interesting future work. They allow a merging approach for mining boundary sets of constrained multidimensional patterns (with arbitrary monotone and/or antimonotone constraints).

References

1. R. J. Bayardo, Jr. Efficiently Mining Long Patterns from Databases. In *Proceedings of the International Conference on Management of Data, SIGMOD*, pages 85–93, 1998.

2. K. Beyer and R. Ramakrishnan. Bottom-Up Computation of Sparse and Iceberg CUBEs. In *Proceedings of the International Conference on Management of Data, SIGMOD*, pages 359–370, 1999.
3. T. Calders, R. Ng, and J. Wijsen. Searching for Dependencies at Multiple Abstraction Levels. In *ACM Transactions on Database Systems, ACM TODS*, volume 27(3), pages 229–260, 2002.
4. A. Casali, R. Cicchetti, and L. Lakhal. Cube Lattices: a Framework for Multidimensional Data Mining. In *Proceedings of the 3rd SIAM International Conference on Data Mining, SDM*, pages 304–308, 2003.
5. A. Casali, R. Cicchetti, and L. Lakhal. Extracting semantics from data cubes using cube transversals and closures. In *Proceedings of the 9th International Conference on Knowledge Discovery and Data Mining, KDD*, pages 69–78, 2003.
6. A. Casali, R. Cicchetti, and L. Lakhal. Mining Concise Représentations of Frequent Multidimensional Patterns. In *Proceedings of the 11th International Conference on Conceptual Structures, ICCS*, 2003.
7. G. Dong and J. Li. Efficient Mining of Emerging Patterns: Discovering Trends and Differences. In *Proceedings of the 5th International Conference on Knowledge Discovery and Data Mining, KDD*, pages 43–52, 1999.
8. T. Eiter and G. Gottlob. Identifying The Minimal Transversals of a Hypergraph and Related Problems. In *SIAM Journal on Computing*, volume 24(6), pages 1278–1304, 1995.
9. J. Gray, S. Chaudhuri, A. Bosworth, A. Layman, D. Reichart, M. Venkatrao, F. Pellow, and H. Pirahesh. Data cube: A relational aggregation operator generalizing group-by, cross-tab, and sub-totals. In *Data Mining and Knowledge Discovery*, volume 1(1), pages 29–53, 1997.
10. D. Gunopulos, H. Mannila, R. Khardon, and H. Toivonen. Data mining, hypergraph transversals, and machine learning. In *Proceedings of the 16th Symposium on Principles of Database Systems, PODS*, pages 209–216, 1997.
11. J. Han, J. Pei, G. Dong, and K. Wang. Efficient Computation of Iceberg Cubes with Complex Measures. In *Proceedings of the International Conference on Management of Data, SIGMOD*, pages 441–448, 2001.
12. L. Lakshmanan, J. Pei, and J. Han. Quotient Cube: How to Summarize the Semantics of a Data Cube. In *Proceedings of the 28th International Conference on Very Large Databases, VLDB*, 2002.
13. S. Lopes, J. Petit, and L. Lakhal. Efficient Discovery of Functional Dependencies and Armstrong Relations. In *Proceedings of the 7th International Conference on Extending Database Technology, EDBT*, pages 350–364, 2000.
14. H. Mannila and H. Toivonen. Levelwise Search and Borders of Theories in Knowledge Discovery. In *Data Mining and Knowledge Discovery*, volume 1(3), pages 241–258, 1997.