Zigzag: a new algorithm for mining large inclusion dependencies in databases

Fabien De Marchi, Jean-Marc Petit
Laboratoire LIMOS, UMR CNRS 6158
Université Blaise Pascal - Clermont-Ferrand II,
24 avenue des Landais 63 177 Aubière cedex, France
{demarchi,jmpetit}@math.univ-bpclermont.fr

Abstract

In the relational model, inclusion dependencies (INDs) convey many information on data semantics. They generalize foreign keys, which are very popular constraints in practice. However, one seldom knows the set of satisfied INDs in a database. The IND discovery problem in existing databases can be formulated as a data-mining problem. We underline in this article that the exploration of IND expressions from most general (smallest) INDs to most specific (largest) INDs does not succeed whenever large INDs have to be discovered. To cope with this problem, we introduce a new algorithm, called $Zigzag$, which combines the strength of levelwise algorithms (to find out some smallest INDs) with an optimistic criteria to jump more or less to largest INDs. Preliminary tests, on synthetic databases, are presented and commented on. It is worth noting that the main result of this paper is general enough to be applied to other data-mining problems, such as maximal frequent itemsets mining.

1. Introduction

In the relational model, inclusion dependencies (INDs) convey many information on the data structure and data semantics, and generalize in particular foreign keys [10].

Their utility is recognized for many tasks such as semantic query optimization [18, 12], logical and physical database design and tuning [26, 11, 23, 14] or data integration [28]. In practice, the implementation of these technics is generally made impossible by the ignorance of satisfied INDs in the database. One can then be interested in the problem of discovering INDs satisfied in a database [20]. In [13], we proposed an approach which, given a database, finds out most specific satisfied INDs, referred to as the *positive border of satisfied INDs* in the sequel. An original proposal for unary IND discovery is given, together with a levelwise algorithm which walks through the IND search space from most general (smallest) INDs to most specific (largest) INDs. We qualify this approach of "pessimistic", since the positive border to be discovered is supposed not to be too far away from the most general INDs. In other words, most INDs to be tested appear to be not satisfied in the database. Not surprisingly, experiments show that this approach does not succeed whenever large INDs have to be discovered.

Of course, the same observation has been already made in similar contexts, typically for *large frequent itemsets* discovery [24, 5, 17, 2]: when the positive border to reach is made up of large candidates, levelwise methods do not scale up. Works related to large frequent itemsets discovery can be classified as follows:

- those which use the properties of frequent itemsets in order to avoid data accesses, one can quote [5, 30, 4, 9, 8].
- those which conjecture that there exist large itemsets [24, 17]. We will qualify these approaches of *optimistic*, since they benefit from the assumption that the majority of the tests to be carried out against the database are expected to be true.

Note also that an analogy between frequent itemsets and satisfied INDs can me made: if one guesses one of them of size n, then approximately $O(2^n)$ tests against data can be saved. Nevertheless, the problem arising for optimistic IND discovery is distinct from optimistic frequent itemsets discovery due to the nature of INDs:

- There does not exist, to our knowledge, properties making it possible to undoubtedly deduce satisfied large INDs without accessing the data (except properties given by IND inference rules).
- The search space exploration can be made only from INDs of small size to INDs of high size: let us consider without loss of generality two relations R and S having N attributes all of the same type: we have $O(N^2)$

candidate INDs of size 1 and more than $N!$ candidate INDs of size N. Note that this is false for the itemsets: consider a transaction database with N items, the number of candidate itemsets of size i is equal to that of size $N-i$, for $i \in \{0, N\}$. That means that the levelwise discovery of frequent itemsets in a bottom-up fashion is only justified by the hope to have a positive border composed of elements of small size.

- To our knowledge, there is not theoretical justification making it possible to be optimistic with frequent itemsets whereas we will see that such a justification exists for INDs: we will show how an axiom of interaction between INDs and Functional Dependencies (FDs) can be used to justify the following intuition: "The more large is the size of known satisfied INDs, the more chance candidate INDs of higher size are satisfied."

Contribution. For IND discovery in a database, we propose a new algorithm called $Zigzag$ combining a pessimistic approach for small INDs with an optimistic approach for large INDs. The main result of this paper gives a characterization of the *optimistic positive border* of the satisfied INDs according to the known unsatisfied INDs. This characterization is different from that given in a similar context [24, 17]. In order to guide the search, a key factor of $Zigzag$ is to estimate the distance between the optimistic positive border and the positive border to be discovered.

Experimental results of this approach are described on synthetic databases. Let us note that the majority of the propositions apply to all search spaces provided with a specialization relation in which a monotonous property is required (as for frequent itemsets).

Related works. To our knowledge, only few contributions address a subset of the initial problem of IND discovery: problem declaration [20], unary IND discovery [6], or theoretical frameworks in which the problem of IND discovery could be solved [19, 27]. During the writing of this article, we were informed of an interesting contribution which addresses the problem of large IND discovery [21]; the idea is to build an optimistic positive border starting from a set of known satisfied INDs, by introducing the concept of maximal hyperclique of a regular hypergraph. The authors do not mention theoretical justification for the use of an optimistic approach, and they do not integrate the distance with the positive border in the event of unsatisfied large INDs, to guide the algorithm during the search of the positive border. Moreover, their generation of hyperclique is based on an adaptation of heuristics known for the maximal clique problem which do not seem to be very efficient.

In [19], the authors propose an algorithm "Dualize and Advance" for search spaces provided with a relation of specialization. In their approach, the positive border in construction is always a subset of the positive border to be discovered. At each step, from some elements of the positive border, they generate the so-called negative border, i.e. elements which should not be satisfied in the database. If one of them appears to be satisfied, they generate a specialization of it which belongs to the positive border and they reiterate the process until each element of the negative border is indeed not satisfied. A difference with our algorithm is that the same strategy is always made to explore the candidates, it can not be guided by an estimation of the distance to the positive border. Another point is that it starts from an empty positive border, while a levelwise strategy initializes the negative border in our algorithm.

Paper organization. Section 2 points out some concepts of the relational model, and specifies the theoretical framework of IND discovery. Section 3 presents the proposed approach: a result linking the generation of large candidates to the computation of minimal transversals of an hypergraph is proposed (Section 3.1), with several justifications for the use of an hybrid approach combining optimism and pessimism (Section 3.2). An exploitation of the error made on the large INDs is also proposed to help in the orientation of the search. An algorithm is given in Section 3.3, and some practical considerations in Section 3.4. Section 3.5 presents preliminary experimental results, and we conclude in Section 4 giving some prospects.

2. Preliminaries

2.1. Relational model and inclusion dependencies

Some concepts of the relational databases, essential to the comprehension of the article, are briefly recalled (see for example [1, 22] for more details).

Let R be a finite set of *attributes*. For each attribute $A \in R$, the set of all its possible values is called the *domain of* A and denoted by $Dom(A)$. A *tuple* over R is a mapping $t : R \rightarrow \cup_{A \in R} Dom(A)$, where $t(A) \in Dom(A), \forall A \in R$. A *relation* is a set of tuples. The cardinality of a set X is denoted by $|X|$. We say that r is a relation *over* R and R is the *relation schema* of r. If $X \subseteq R$ is an attribute set[1] and t is a tuple, we denote by $t[X]$ the restriction of t to X. The projection of a relation r onto X, denoted as $\pi_X(r)$, is defined by $\pi_X(r) = \{t[X] \mid t \in r\}$.

A *database schema* **R** is a finite set of *relation schemas* R_i. A *relational database instance* **d** (or *database*) over **R** corresponds to a set of relations r_i over each R_i of **R**. Given a database **d** over **R**, the set of distinct domains (e.g. int, string ...) is denoted by $Dom(\mathbf{d})$.

[1] Letters from the beginning of the alphabet introduce single attributes whereas letters from the end introduce attribute sets.

An attribute sequence (e.g. $X = < A, B, C >$ or simply ABC) is an ordered set of attributes. When it is clear from context, we do not distinguish a sequence from its underlying set.

Two attributes A and B are said to be *compatible* if $Dom(A) = Dom(B)$. Two distinct attribute sequences X and Y are *compatible* if $|X| = |Y| = m$ and if for $j = [1, m]$, $Dom(X[j]) = Dom(Y[j])$.

An *inclusion dependency* (IND) over a database schema **R** is a statement of the form $R_i[X] \subseteq R_j[Y]$, where $R_i, R_j \in \mathbf{R}, X \subseteq R_i, Y \subseteq R_j$, X and Y are compatible sequences[2].

The size (or arity) of an IND $i = R[X] \subseteq R[Y]$, noted $|i|$ is such that $|i| = |X| = |Y|$. We call *unary inclusion dependency* an IND of size 1.

Let **d** be a database over a database schema **R**, where $r_i, r_j \in$ **d** are relations over $R_i, R_j \in$ **R** respectively. An inclusion dependency $R_i[X] \subseteq R_j[Y]$ is *satisfied* in a database **d** over **R**, denoted by $\mathbf{d} \models R_i[X] \subseteq R_j[Y]$, if and only if $\forall u \in r_i, \exists v \in r_j$ such that $u[X] = v[Y]$ (or equivalently $\pi_X(r_i) \subseteq \pi_Y(r_j)$).

Let I_1 and I_2 be two sets of inclusion dependencies, I_1 is a *cover* of I_2 if $I_1 \models I_2$ (this notation means that each dependency in I_2 holds in any database satisfying all the dependencies in I_1) and $I_2 \models I_1$.

A sound and complete axiomatization for INDs was given in [29]. If I is a set of INDs, we have:

1. (reflexivity) $I \models R[A_1, ..., A_n] \subseteq R[A_1, ..., A_n]$
2. (projection and permutation) if $I \models R[A_1, ..., A_n] \subseteq S[B_1, ..., B_n]$ then $I \models R[A_{\sigma 1}, ..., A_{\sigma m}] \subseteq S[B_{\sigma 1}, ..., B_{\sigma m}]$ for each sequence $\sigma 1, ..., \sigma m$ of distinct integers from $\{1, ..., n\}$
3. (transitivity) if $I \models R[A_1, ..., A_n] \subseteq S[B_1, ..., B_n]$ et $I \models S[B_1, ..., B_n] \subseteq T[C_1, ..., C_n]$ then $I \models R[A_1, ..., A_n] \subseteq T[C_1, ..., C_n]$
4. (attribute equality) if $I \models R[AB] \subseteq S[CC]$, then A and B can be substituted to each other in all satisfied IND expressions.
5. (redundancy) if $I \models R[X] \subseteq S[Y]$, then $I \models R[XU] \subseteq S[YV]$, where $R[U] \subseteq S[V]$ can be obtained from $R[X] \subseteq S[Y]$ using second inference rule.

2.2. A framework for IND discovery

The problem of IND discovery can be formulated as follows: " Let **d** be a database, find a cover of all satisfied INDs in **d**". The number of potentially satisfied INDs, which constitutes the basic search space, is more than factorial in the number of attributes. We point out it here, and add some notations used in the sequel.

Definition 1 *Let* **R** *be a database schema. The search space of INDs over* **R** *, denoted by C, is defined by:*

$$C = \{R[< A_1...A_n >] \subseteq S[< B_1...B_n >] \mid R, S \in \mathbf{R}$$

$$\text{and } \forall 1 \leq i < j \leq n, (A_i < A_j) \vee (A_i = A_j \wedge B_i < B_j)\}$$

Thus, C contains IND expressions defined over **R** whose sequences are sorted in order to naturally restrict the search space to only one permutation of each IND expression (according to the second inference rule of INDs). In the sequel, let **d** be a database over a schema R, and C the corresponding IND search space.

Definition 2 *Let i and j be two IND expressions, we say that i* generalizes *j (or j* specializes *i), noted $i \preceq j$, if i can be obtained by projection on j.*

For example, $R[AC] \subseteq S[DF] \preceq R[ABC] \subseteq S[DEF]$. We note $i \prec j$ for $i \preceq j$ and $i \neq j$.

The satisfaction property of an IND in a database **d** is anti-monotonous with respect to the relation $' \preceq'$:

Property 1 *Let $i, j \in C$ such that $j \preceq i$. We have:* $\mathbf{d} \not\models j \Rightarrow \mathbf{d} \not\models i$.

Thus, any set of INDs can be represented by its most specialized elements, i.e. its positive border:

Definition 3 *Let $I \subseteq C$. The positive border of I, noted $\mathcal{B}d^+(I)$, is defined by: $\mathcal{B}d^+(I) = \{i \in I \mid \nexists j \in I, i \prec j\}$.*

A set I of INDs can also be represented by the most general elements which it does not contain, i.e. its negative border:

Definition 4 *Let $I \subseteq C$. The negative border of I, noted $\mathcal{B}d^-(I)$, is defined by: $\mathcal{B}d^-(I) = \{i \in C \setminus I \mid \nexists j \in C \setminus I, j \prec i\}$.*

Note that when I is the set of the satisfied INDs in **d** , $\mathcal{B}d^+(I)$ answers the IND discovery problem. In [27], an levelwise approach is suggested to discover $\mathcal{B}d^+(I)$. The algorithm $MIND$ [13] in based on this idea, using an $AprioriGen$ like candidate generation [3]. Such an approach will be qualified in the sequel of *pessimistic*, because its effectiveness is based on the presence on each level of many unsatisfied INDs, in order to prune a great part of the remaining space. The experiments conducted in [13] show that such an approach is scalable according to the number of tuples and attributes: the greatest database had 90000 tuples and 200 attributes, the IND positive border of the database was composed of four unary IND and one IND of size 6.

[2] Usually, the IND definition excludes repeated attributes in the sequences on left and right-hand sides. In this paper, we prefer to adopt a less restrictive framework in order to simplify the presentation of the principal result of this paper (cf Section 3.1) ; the exclusion of the repeated attributes is considered after the presentation of the algorithm (cf. Section 3.4).

Nevertheless, such a pessimistic approach is not adapted when large INDs have to be discovered; indeed, to discover an IND i of size n, it is necessary to have discovered the 2^n INDs which generalize i.

3. An hybrid approach for IND discovery

The principle of the method is to combine pessimistic and optimistic approaches. We first give some definitions and results which will make it possible to carry out "optimistic jumps" in the search space.

3.1. The optimistic positive border

The simple remark on which this work is founded is the following: a set of unsatisfied INDs makes it possible to prune a certain number of candidates by anti-monotony, and thus to define an *optimistic set of satisfied INDs*.

Definition 5 *Let C be the IND search space associated to a database **d** and $NI \subseteq C$ be a set such that $\forall i \in NI, \mathbf{d} \not\models i$. The* optimistic set of INDs *with respect to NI, denoted by $I_{opt}(NI)$, is defined by: $I_{opt}(NI) = \{i \in C \mid \not\exists j \in NI, j \preceq i\}$.*

Moreover, the *optimistic positive border*, denoted by $\mathcal{B}d^+(I_{opt}(NI))$, is the set of most specific INDs in $I_{opt}(NI)$. When clear from context, we will note $\mathcal{B}d^+(I_{opt})$ instead of $\mathcal{B}d^+(I_{opt}(NI))$.

Remark that $\mathcal{B}d^+(I_{opt})$ corresponding to a set NI is exactly equal to $\mathcal{B}d^+(I_{opt})$ corresponding to the most general elements of NI.

We shall see that the optimistic positive border can be calculated by using the concept of minimal transversals of an hypergraph. As in [19], we need a representation as sets of INDs, given in the following definition:

Definition 6 *Let I_1 be the set of unary INDs in C. Each IND i in C can be associated with a set of unary INDs, thanks to the function $ens : C \longrightarrow 2^{I_1}$ defined by: $ens(i) = \{j \in I_1 \mid j \preceq i\}$.*

Example 1 *Consider the three unary INDs: $i_1 = R[A] \subseteq S[E]$, $i_2 = R[B] \subseteq S[F]$ and $i_3 = R[C] \subseteq S[G]$. The set representation of $i = R[ABC] \subseteq S[EFG]$ is $ens(i) = \{i_1, i_2, i_3\}$.*

The function ens is bijective, and thus imply a function ens^{-1} which associates to any IND of C one element of 2^{I_1}.

The following proposal follows from the representation as set of INDs and the definition of the relation $'\preceq'$:

Property 2 *If i and j are two IND expressions, $i \preceq j \Leftrightarrow ens(i) \subseteq ens(j)$.*

Thanks to this representation as set of INDs we can define an hypergraph associated with a set of INDs:

Definition 7 *Let I be a subset of C. The hypergraph (V, E) associated with I, denoted by $\mathcal{H}(I)$, is defined by: $V = \{i \in C \mid |i| = 1\}$ and $E = \{ens(i) \mid i \in I\}$.*

From this representation of INDs, we obtain a relationship between a set of unsatisfied INDs and the corresponding optimistic positive border.

Theorem 1 *Let $NI \subseteq C$ be a set of unsatisfied INDs in **d** and I_1 the set of unary INDs in C. The positive border w.r.t. NI is such that:*

$$\mathcal{B}d^+(I_{opt}) = \{i \in C \mid \overline{ens(i)} \in \mathcal{T}r(\mathcal{H}(NI))\}$$

where $\overline{ens(i)}$ is the complement of $ens(i)$ in I_1 and $\mathcal{T}r(\mathcal{H}(NI))$ the set of minimal transversals[3] of $\mathcal{H}(NI)$.

Proof *Let $i \in C$ be an IND expression.*
First, we show that $i \in I_{opt} \Leftrightarrow \overline{ens(i)}$ is a transversal of $\mathcal{H}(NI)$:
$i \in I_{opt}$
$\Leftrightarrow \forall j \in NI, j \not\preceq i$
$\Leftrightarrow \forall j \in NI, ens(j) \not\subseteq ens(i)$
$\Leftrightarrow \forall j \in NI, ens(j) \cap \overline{ens(i)} \neq \emptyset$
$\Leftrightarrow \overline{ens(i)}$ is a transversal of $\mathcal{H}(NI)$.
Then we show that i is maximal in $I_{opt} \Leftrightarrow \overline{ens(i)}$ is a minimal transversal of $\mathcal{H}(NI)$:
Let $i \in \mathcal{B}d^+(I_{opt})$. Since $i \in I_{opt}$, $\overline{ens(i)}$ is a transversal of $\mathcal{H}(NI)$. Suppose $\overline{ens(i)}$ is not minimal: $\exists X \in 2^{I_1}$, X transversal of $\mathcal{H}(NI)$ and $X \subset \overline{ens(i)}$, and thus $ens(i) \subset \overline{X}$. Then $ens^{-1}(\overline{X}) \in I_{opt}$ and $i \prec ens^{-1}(\overline{X})$ from proposition 2, which contradict the fact that $i \in \mathcal{B}d^+(I_{opt})$.
Now, let $X \in \mathcal{T}r(\mathcal{H}(NI))$. X is a transversal, then $ens^{-1}(\overline{X}) \in I_{opt}$. Suppose that $ens^{-1}(\overline{X})$ is not maximal: $\exists j \in I_{opt} \mid ens^{-1}(\overline{X}) \prec j$. Then $\overline{ens(j)}$ is a transversal of $\mathcal{H}(NI)$ with $\overline{X} \subseteq ens(j)$, and thus $\overline{ens(j)} \subseteq X$, which contradict the fact that X is a minimal transversal.

3.2. Principles of the algorithm

The proposed method can be decomposed into two main steps: a pessimistic exploration of the most general INDs until a given level, and then a "zigzag" between the negative border in construction and the corresponding optimistic positive border. Before entering into some details, some properties of INDs are studied to give some confidence in our optimistic criteria.

Why to be pessimistic ? Several factors justify to be pessimistic for the candidate INDs of small size. The first one

[3] A minimal transversal of an hypergraph (V, E) is a set of elements of V such that (1) it has an non empty intersection with every element of E and (2) it is minimal w.r.t. this property.

is that the first level contains the lowest number of candidates in all the search space, and thus is the best *entry point* of any exploration of C.

The second one is that in practice, a great proportion of unary IND candidates are not satisfied. Thus a significant part of the search space is disqualified by anti-monotony, justifying a pessimistic approach for this level.

Lastly, an efficient method was proposed in [13] for unary IND discovery, based on a data reorganization. The salient feature of this approach is not to make as many database passes as candidates exist (as it is the case in general for dependence discovery [27]), but only one database pass for all the candidates (as it is for example the case for frequent itemsets). To the best of our knowledge, from the level 2, no other method than generating and testing candidates one by one exists. Continuing a pessimistic approach is likely to succeed only if the majority of the candidates turns out to be false.

Why to be optimistic ? A justification for an optimistic approach can be expressed as follows: if all generalizations of size k of a candidate IND i are satisfied, then i has more chances to be satisfied when k increases. This result is justified by an inference rule[4] of Functional Dependencies (FDs) and INDs given by:

Proposition 1 *Let $\{r, s\}$ be a database, C the corresponding IND search space, and $I_k = \{i \in C, |i| = k \mid \{r, s\} \models i\}$, $k \geq 2$. Let $i = R[X] \subseteq S[Y], i \in C, |i| = n, n > k$ such that $\forall j \in C, |j| = k, j \prec i$, we have $j \in I_k$.*
If $\exists Y_1 \subseteq Y$ such that $|Y_1| = k - 1$ and $s \models Y_1 \rightarrow Y \setminus Y_1$, then $\{r, s\} \models i$

Proof *Let $\mathbf{d} = \{r, s\}$ be a database and C the corresponding IND search space. Let $i = R[X] \subseteq S[Y] \in C$ an IND expression of arity $n \geq 3$, and an integer $k < n$. Suppose that all INDs which generalize i, of size lower or equal to k, are satisfied. And let $Y_1 \subseteq Y$ be such that $|Y_1| = k - 1$ and $s \models Y_1 \rightarrow Y \setminus Y_1$.*
Let us put $Y \setminus Y_1 = B_1...B_{n-k+1}$. We note X_1 the sub-sequence of X in which the position of elements in X correspond to the position of elements of Y_1 in Y, and $A_1, ..., A_{n-k+1}$ the elements of X in which the position of elements in X correspond to the position respectively of $B_1, ..., B_{n-k+1}$ in Y.
Let $t \in r$. We have $\mathbf{d} \models R[X_1A_1] \subseteq S[Y_1B_1]$, since this IND is of arity k. Thus $\exists u_1 \in s$ such that $u_1[Y_1B_1] = t[X_1A_1]$. In the same way, $\mathbf{d} \models R[X_1A_2] \subseteq S[Y_1B_2]$, then $\exists u_2 \in s$ such that $u_2[Y_1B_2] = t[X_1A_2]$. We know that $s \models Y_1 \rightarrow B_2$ and thus $u_1[B_2] = u_2[B_2]$ since $u_1[Y_1] = u_2[Y_1]$. Thus, $u_1[Y_1B_1B_2] = t[X_1A_1A_2]$. We can repeat $n - k + 1$ times the same reasoning, to show that $u_1[Y_1B_1B_2...B_{n_k+1}] = t[X_1A_1A_2...A_{n_k+1}]$, and then $u_1[Y] = t[X]$. This is true for all tuple in r, and we have $\mathbf{d} \models R[X] \subseteq S[Y]$.

[4] The inference rule stated by the proposition 1 does not form part of the Mitchell system [29], but of course is inferred by this system which is sound and complete. The demonstration suggested here seemed more comprehensible and shorter.

Example 2 *Consider the IND $i = R[ABCDEF] \subseteq S[GHIJKL]$. Suppose that the 20 INDs of size 3 which generalize i are satisfied; then if there exists two attributes of $GHIJKL$ that determine the others, for example $GH \rightarrow IJKL$, we have $\mathbf{d} \models i$.*

Thus, the principle justified by this rule is, starting from an explored level k, to build the highest IND expressions for which all sub-INDs of size k are true. Notice that the larger k is, the more there are chances that sets of attributes of size $k - 1$ determine the others, meeting the conditions of the proposition 1.

However, when a suspected large IND i of size n is detected as false, it is necessary to choose between two alternatives: turn back to the level $k + 1$ "to consolidate" basic knowledge, or maintain an optimistic attitude by testing the INDs of the level $n - 1$ which generalize i. The following paragraph treats choice between these two alternatives.

Estimating the distance from positive border. Each time a candidate generated by an optimistic approach is detected to be false against the database, we try "to estimate" the distance between this element and the positive border of satisfied INDs. The idea is to count the number of tuples which does not satisfy the IND; we propose for that to use the error measure g_3' [25] given by:

$$g_3'(R[X] \subseteq S[Y], \mathbf{d}) = 1 - \frac{max\{|\pi_X(r')| \mid r' \subseteq r, (\mathbf{d} - \{r\}) \cup \{r'\} \models R[X] \subseteq S[Y]\}}{|\pi_X(r)|}$$

Intuitively, g_3' is the proportion of distinct values one has to remove from X to obtain a database $\mathbf{d}'$ such that $\mathbf{d}' \models R[X] \subseteq S[Y]$. Such a computation can be implemented with SQL queries on top of RDBMS. Clearly, if $j \preceq i$, then $g_3'(j) \leq g_3'(i)$. Thus, when an IND i of the optimistic positive border is false, but with a very small error, one can reasonably hope to find a satisfied IND among its nearest generalizations. Thus we consider the generalizations of i from more specific ones to most general ones, i.e. implementing a top-down approach. Inversely when the error is large, i.e. a great number of values contradicts the IND, we start again the search in a bottom-up fashion.

3.3. The algorithm $Zigzag$

The principle of the algorithm 1 is to mix top-down and bottom-up approaches for eliciting the positive border of satisfied INDs. Initially (line 1) a purely pessimistic approach is performed from an adaptation of the levelwise algorithm $MIND$ [13], until the level k fixed by the user is reached. We then know I_k and NI_k, the set of the most specialized satisfied INDs and the set of the most general unsatisfied INDs (resp.) of size smaller or equal to k. I_k thus corresponds to an initialization of $\mathcal{B}d^+(I)$ and NI_k to an initialization of $\mathcal{B}d^-(I)$. The optimistic positive border $\mathcal{B}d^+(I_{opt})$ is then computed from $\mathcal{B}d^-(I)$ thanks to the theorem 1 (line 3)[5]. The algorithm terminates when every element of $\mathcal{B}d^+(I_{opt})$ has already been tested as true in previous passes, i.e. $\mathcal{B}d^+(I_{opt}) \setminus \mathcal{B}d^+(I)$ is empty (line 4). Otherwise, INDs of $\mathcal{B}d^+(I_{opt}) \setminus \mathcal{B}d^+(I)$ are evaluated against the database: Those satisfied are added to $\mathcal{B}d^+(I)$, the others are divided into two groups according to the committed error: the "almost true" ones in the optimistic set $optDI$ and the others in the pessimistic set $pessDI$. The INDs which generalize the INDs of $optDI$ are traversed in a top-down fashion, from the most specific to the more general; $\mathcal{B}d^+(I)$ and $\mathcal{B}d^-(I)$ are updated accordingly (lines 14 to 21). Lastly, INDs of size $k+1$ which generalize the INDs of $pessDI$ are tested, $\mathcal{B}d^+(I)$ and $\mathcal{B}d^-(I)$ are also updated (lines 23 to 26). $\mathcal{B}d^+(I_{opt})$ is then updated for the next iteration (line 28).

Algorithm 1 Zigzag : IND cover discovery

Require: a database **d** , an integer k;
Ensure: $\mathcal{B}d^+(I)$ cover of the satisfied INDs in **d** .
1: Compute I_k and NI_k by a pessimistic aproach.
2: $\mathcal{B}d^+(I) = I_k; \mathcal{B}d^-(I) = NI_k$;
3: Compute $\mathcal{B}d^+(I_{opt})$ from $\mathcal{B}d^-(I)$;
4: **while** $\mathcal{B}d^+(I_{opt}) \setminus \mathcal{B}d^+(I) \neq \emptyset$ **do**
5: $optDI = pessDI = \emptyset$;
6: **for all** $i \in \mathcal{B}d^+(I_{opt}) \setminus \mathcal{B}d^+(I)$ **do**
7: **if** $(g'_3(i, \mathbf{d}) = 0)$ **then** $\mathcal{B}d^+(I) = \mathcal{B}d^+(I) \cup \{i\} \setminus \{j \in \mathcal{B}d^+(I) \mid j \prec i\}$;
8: **else**
9: $\mathcal{B}d^-(I) = \mathcal{B}d^-(I) \cup i$;
10: **if** $(g'_3(i, \mathbf{d}) \leq \epsilon$ and $|i| > k+1)$
11: **then** $optDI = optDI \cup \{i\}$;
12: **else** $pessDI = pessDI \cup \{i\}$;
13: **end for**
14: **while** $optDI \neq \emptyset$ **do**
15: $candidats = \cup_{i \in optDI}\{j \mid j \preceq i, |j| = |i| - 1$ and $|j| > k\}$;
16: **for all** $i \in candidats$ **do**
17: **if** $(\mathbf{d} \models i)$ **then** $\mathcal{B}d^+(I) = \mathcal{B}d^+(I) \cup \{i\} \setminus \{j \in \mathcal{B}d^+(I) \mid j \prec i\}$; $candidats = candidats \setminus \{i\}$;
18: **else** $\mathcal{B}d^-(I) = \mathcal{B}d^-(I) \cup \{i\} \setminus \{j \in \mathcal{B}d^-(I) \mid i \prec j\}$;
19: **end for**
20: $optDI = candidats$;
21: **end while**
22: $C_{k+1} = \cup_{i \in pessDI}\{j \mid j \prec i, |j| = k+1\}$;
23: **for all** $i \in C_{k+1}$ **do**
24: **if** $(\mathcal{B}d^+(I) \models i$ or $\mathbf{d} \models i)$ **then** $\mathcal{B}d^+(I) = \mathcal{B}d^+(I) \cup \{i\} \setminus \{j \in \mathcal{B}d^+(I) \mid j \prec i\}$;
25: **else** $\mathcal{B}d^-(I) = \mathcal{B}d^-(I) \cup \{i\} \setminus \{j \in \mathcal{B}d^-(I) \mid i \prec j\}$;
26: **end for**
27: $k = k+1$;
28: Compute $\mathcal{B}d^+(I_{opt})$ from $\mathcal{B}d^-(I)$;
29: **end while**
30: Return $\mathcal{B}d^+(I)$.

3.4. Practical aspects and optimizations

Unsatisfied unary INDs. In algorithm 1, line 2, the most general unsatisfied INDs are added in the initialization of $\mathcal{B}d^-(I)$. In practice, we do not add unsatisfied unary INDs to $\mathcal{B}d^-(I)$. It is enough not to take them into account during the computation of the complements of minimal transversal of the hypergraph associated with $\mathcal{B}d^-(I)$.

Dealing with repeated attributes in INDs. The usual IND definition rejects the repeated attributes in the left-hand side or the right-hand side, which reduces of course in a significant way the size of the search space; it is thus important to remove candidates with duplicate attributes in the optimistic positive border. For that, it is enough to introduce into the negative border, during its initialization (line 2 of algorithm 1) the set of INDs of size 2 with repeated attributes made up of two satisfied unary INDs. Indeed, consider an IND i having at least one repeated attribute on the left-hand side[6], i.e. $i = R[X_1AAX_2] \subseteq S[Y_1BCY_2]$. Thus there exist at least one IND of size 2, here $j = R[AA] \subseteq S[BC]$, which generalizes i. If j belong to the negative border, then i cannot belong to the corresponding optimistic positive border, according to definition 5.

Parallelization of $Zigzag$. An interesting property of the IND search space can be given, making it possible to break up our exploration method into several independent courses.

Property 3 *Let* **d** *be a database over a schema* ***R*** *and* I *the set of satisfied INDs in* **d** *. Then* $\mathcal{B}d^+(I) = \bigcup_{(R,S) \in \mathbf{d}^2} \mathcal{B}d^+(I_{R \rightarrow S})$ *where* $I_{R \rightarrow S}$ *denotes INDs from* R *to* S.

Thus, the IND discovery can be made through independent tasks, one for each couple of relations in the database. In addition to the natural parallelization of the algorithm, this decomposition also makes it possible to decrease the size of the negative border in each iteration, and thus to improve the performances of the optimistic positive border generation. The current implementation of the algorithm considers the couples of relation one by one, except for unary IND discovery, which as mentioned, does not consist of a candidate enumeration, but of a global data reorganization [13].

Optimistic positive border generation. In addition to the number of tested INDs, which remains factorial in worst cases, $Zigzag$ complexity comes from the optimistic pos-

[5]The optimistic positive border generation is not detailed here. We use an adaptation of the algorithm proposed in [15].

[6]The same justification still holds for right-hand side.

itive border computation, which induces the calculation of minimal transversals of the hypergraph corresponding to the negative border in construction. The computation of minimal transversals of an hypergraph is a well-known problem studied in the literature (see e.g. [16] for a comprehensive survey). The exact complexity is, up to our knowledge, an open problem. The existence of incremental algorithms w.r.t the edges of the hypergraph is very interesting in our case [7, 15, 16]. Indeed, the first calculation occurs at the level k and then, new edges are considered for updating the set of minimal transversals.

An optimization can also be brought to the calculation of $\mathcal{B}d^+(I_{opt})$. Indeed, at each iteration, this set contains at the same time the largest "guessed" INDs, but also some satisfied INDs of smaller size which were already discovered. The idea is thus to characterize more finely the elements of $\mathcal{B}d^+(I_{opt})$: those which are to be discovered, and those already explored. The following results just follows from the theorem 1.

Proposition 2 *Let I_k be the set of satisfied INDs of size less or equal to k, NI the set of unsatisfied INDs, and n the number of unary INDs. We have:*
$i \in (\mathcal{B}d^+(I_{opt}) \setminus I_k) \Longleftrightarrow \overline{ens(i)} \in \mathcal{T}r(\mathcal{H}(NI))$
and $|\overline{ens(i)}| \leq n - k$

In practice, this condition leads to optimize the generation of minimal transversals since candidates exceeding the size allowed can be safely removed.

3.5. Experimental results

Preliminary tests were carried out on synthetic databases. They were performed on an INTEL Pentium III 500 MHz, with 384 MB of main memory and running Windows 2000 Pro. The algorithms were implemented using C++/STL language. The test databases are stored under Oracle 9i, and data accesses were carried out via ODBC drivers. The tests were conducted on three databases having 2 relations, with 25 attributes and 90000 tuples in each relation. The databases differ on the constitution of the positive border of satisfied INDs to discover:

- database 1: 10 INDs, with arities of 2,5,6 and 7;
- database 2: 10 INDs, with arities of 3,5,6 and 11;
- database 3: 20 INDs, with arities of 6,8,13,17 and 18;

Table 1 gives execution times for IND discovery using algorithm $Zigzag$, with $k = 2$. Times are compared with those given by the levelwise algorithm $Mind$ [13]. The value for $Mind$ on the third database is an estimate: it multiplies the number of tests to be carried out with the average cost of a test.

Table 1. Experimental results

database	$Zigzag$	$Mind$
1	1 754 s.	2 790 s.
2	3 500 s.	25 626 s.
3	7 729 s.	$\geq$ 1 year (estimate)

First of all, these results confirm the failure of levelwise approach for large IND discovery, and thus reinforce the interest of proposing alternatives. Moreover, algorithm *Zigzag* makes it possible to reach INDs of size 18 in about only two hours (while $Mind$ would have take more than one year!), and thus shows the feasibility of the approach.

4. Conclusion

A new approach for the IND discovery in an existing database was proposed in this article. Since a levelwise approach does not work when INDs of size higher than 10 have to be discovered, we proposed an approach allowing to carry out *optimistic jumps* in the search space, performing "Zigzags" between the negative and positive borders in construction. This approach relies on a theoretical framework which could be apply to other search spaces. The principle of an optimistic attitude is justified by a structural property of the relational model. The proposed method exploits an error measure to guess how far the positive border of satisfied INDs is w.r.t. the optimistic positive border. Moreover, the characterization of a positive border according to a set of counterexamples is a general result: it can be applied to other data-mining tasks.

This work is integrated in a more general project devoted to DBA assistance and relational databases logical tuning, called "DBA Companion" [14]. We propose methods for discovering INDs, functional dependency and keys, and for computing small samples to represent data semantics (informative Armstrong databases).

We finish this article by mentioning some perspectives which could supplement this work:

- A complete study of $Zigzag$ complexity;
- The development of a more "probabilistic" approach, in order to estimate the value of k (the level from which optimist approach is likely to succeed) and the value of ϵ (to estimate the distance between positive optimistic border and positive border to be discovered);
- The adaptation and the implementation of this work for similar problems such as large frequent itemset discovery or functional dependency discovery.

References

[1] S. Abiteboul, R. Hull, and V. Vianu. *Foundations of Databases*. Addison-Wesley, Reading, Mass., 1995.

[2] C. C. Aggarwal. Towards long pattern generation in dense databases. *SIGKDD Explorations*, 3(1):20–26, 2001.

[3] R. Agrawal and R. Srikant. Fast algorithms for mining association rules in large databases. In J. B. Bocca, M. Jarke, and C. Zaniolo, editors, *International Conference on Very Large Data Bases (VLDB'94), Santiago de Chile, Chile*, pages 487–499. Morgan Kaufmann, 1994.

[4] Y. Bastide, R. Taouil, N. Pasquier, G. Stumme, and L. Lakhal. Mining frequent patterns with counting inference. *ACM SIGKDD Exploration*, 2(2):66–75, 2000.

[5] R. J. Bayardo. Efficiently mining long patterns from databases. In L. M. Haas and A. Tiwary, editors, *ACM SIGMOD International Conference on Management of Data, Seattle, USA*, pages 85–93. ACM Press, 1998.

[6] S. Bell and P. Brockhausen. Discovery of constraints and data dependencies in databases (extended abstract). In N. Lavrac and S. Wrobel, editors, *European Conference on Machine Learning (ECML'95), Crete, Greece*, pages 267–270, 1995.

[7] C. Berge. Hypergraphs. *North Holland Mathematic Library*, 45, 1989.

[8] J.-F. Boulicaut, A. Bykowski, and C. Rigotti. Free-sets: A condensed representation of boolean data for the approximation of frequency queries. *Data Mining and Knowledge Discovery*, 7(1):5–22, 2003.

[9] T. Calders and B. Goethals. Mining all non-derivable frequent itemsets. In *Principles of Data Mining and Knowledge Discovery*, volume 2431 of *Lecture Notes in Computer Science*. Springer, 2002.

[10] M. Casanova, R. Fagin, and C. Papadimitriou. Inclusion dependencies and their interaction with functional dependencies. *Journal of Computer and System Sciences*, 24(1):29–59, Feb. 1984.

[11] M. A. Casanova, L. Tucherman, and A. L. Furtado. Enforcing inclusion dependencies and referencial integrity. In F. Bancilhon and D. J. DeWitt, editors, *International Conference on Very Large Data Bases (VLDB'88), Los Angeles, California, USA*, pages 38–49. Morgan Kaufmann, 1988.

[12] Q. Cheng, J. Gryz, F. Koo, T. Y. C. Leung, L. Liu, X. Qian, and B. Schiefer. Implementation of two semantic query optimization techniques in DB2 universal database. In M. P. Atkinson, M. E. Orlowska, P. Valduriez, S. B. Zdonik, and M. L. Brodie, editors, *International Conference on Very Large Data Bases (VLDB'99), Edinburgh, Scotland, UK*, pages 687–698. Morgan Kaufmann, 1999.

[13] F. De Marchi, S. Lopes, and J.-M. Petit. Efficient algorithms for mining inclusion dependencies. In C. S. Jensen, K. G. Jeffery, J. Pokorný, S. Saltenis, E. Bertino, K. Böhm, and M. Jarke, editors, *International Conference on Extending Database Technology (EDBT'02), Prague, Czech Republic*, volume 2287 of *Lecture Notes in Computer Science*, pages 464–476. Springer, 2002.

[14] F. De Marchi, S. Lopes, J.-M. Petit, and F. Toumani. Analysis of existing databases at the logical level: the DBA companion project. *ACM Sigmod Record*, 32(1):47–52, 2003.

[15] J. Demetrovics and V. Thi. Some remarks on generating armstrong and inferring functional dependencies relation. *Acta Cybernetica*, 12(2):167–180, 1995.

[16] T. Eiter and G. Gottlob. Identifying the minimal transversals of a hypergraph and related problems. *SIAM Journal on Computing*, 24(6):1278–1304, 1995.

[17] K. Gouda and M. J. Zaki. Efficiently mining maximal frequent itemsets. In N. Cercone, T. Y. Lin, and X. Wu, editors, *International Conference on Data Mining, San Jose, USA*. IEEE Computer Society, 2001.

[18] J. Gryz. Query folding with inclusion dependencies. In *International Conference on Data Engineering (ICDE'98), Orlando, Florida, USA*, pages 126–133. IEEE Computer Society, 1998.

[19] D. Gunopulos, R. Khardon, H. Mannila, and H. Toivonen. Data mining, hypergraph transversals, and machine learning. In *Symposium on Principles of Database Systems (PODS'97), Tucson, Arizona*, pages 209–216. ACM Press, 1997.

[20] M. Kantola, H. Mannila, K. J. Räihä, and H. Siirtola. Discovering functional and inclusion dependencies in relational databases. *International Journal of Intelligent Systems*, 7:591–607, 1992.

[21] A. Koeller and E. A. Rundensteiner. Discovery of highdimentional inclusion dependencies (poster). In *International Conference on Data Engineering*. IEEE Computer Society, 2003.

[22] M. Levene and G. Loizou. *A Guided Tour of Relational Databases and Beyond*. Springer, 1999.

[23] M. Levene and M. W. Vincent. Justification for inclusion dependency normal form. *IEEE Transactions on Knowledge and Data Engineering*, 12(2):281–291, 2000.

[24] D.-I. Lin and Z. M. Kedem. Pincer search: A new algorithm for discovering the maximum frequent set. In H.-J. Schek, F. Saltor, I. Ramos, and G. Alonso, editors, *Extending Database Technology, Valencia, Spain*, volume 1377 of *Lecture Notes in Computer Science*, pages 105–119. Springer, 1998.

[25] S. Lopes, J.-M. Petit, and F. Toumani. Discovering interesting inclusion dependencies: Application to logical database tuning. *Information System*, 17(1):1–19, 2002.

[26] H. Mannila and K.-J. Räihä. Inclusion dependencies in database design. In *International Conference on Data Engineering (ICDE'86), Los Angeles, California, USA*, pages 713–718. IEEE Computer Society, 1986.

[27] H. Mannila and H. Toivonen. Levelwise Search and Borders of Theories in Knowledge Discovery. *Data Mining and Knowledge Discovery*, 1(1):241–258, 1997.

[28] R. J. Miller, M. A. Hernández, L. M. Haas, L. Yan, C. T. H. Ho, R. Fagin, and L. Popa. The clio project: Managing heterogeneity. *ACM SIGMOD Record*, 30(1):78–83, 2001.

[29] J. C. Mitchell. The implication problem for functional and inclusion dependencies. *Information and Control*, 56(3):154–173, Mar. 1983.

[30] 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.