Journal on Relational Methods in Computer Science, Vol. 1, 2004, pp. 217 - 234

Discovering Regularities in Databases Using Canonical Decomposition of Binary Relations

A. Jaoua[1], S. Elloumi[2], A. Hasnah[1], J. Jaam[1], and I. Nafkha[2]

[1]University of Qatar, College of Science,
Department of Computer Science.
{jaoua;hasnah;jaam}@qu.edu.qa
[2]Faculty of Science of Tunis, University of Manar-Tunis
Unité de Recherche en Programmation Algorithmique et Heuristique
{samir.elloumi;ibtissem.nafkha}@fst.rnu.tn

Abstract. Regularities in databases are directly useful for knowledge discovery and data summarization. As a mathematical background, relational algebra helped for discovering the main data structures and existing dependencies between the different attributes in a relational database. Functional, difunctional and other kinds of dependencies in a relational database describe invariant regular structures that have been used intensively for database decomposition, and for minimizing redundancy. In this paper, we explain why "concepts" or "maximal rectangles" should be considered as the atomic regular structure for decomposing a binary relation which can be useful for different applications. More specifically, we have noticed experimentally, that "optimal concepts" contain pertinent information about data that we have exploited positively for machine learning, dynamic and incremental database organization, text summarization, data reduction, and even for modeling human thinking. Operators on concepts need to be developed because of their general usefulness in data and information engineering. In this paper, we propose to work on a canonical decomposition of binary relations based on two operators f and g, to model some important open problems, as for example on how to put in equation the best optimal conceptual coverage of a binary relation. We first develop an algorithm to find a conceptual coverage of a binary relation. We then exploit Riguet's difunctional relation to put in equation all isolated pairs in a binary relation. Using iteratively these isolated pairs, we find several varieties of efficient solutions for the canonical decomposition problem.

Keywords: regularities in database, maximal rectangles, concepts, knowledge extraction, data reduction

Received by the editors April 29, 2004, and, in revised form, October 21, 2004.
Published on December 10, 2004.

1 Introduction

Regular structures in databases played a major role for database decomposition, and for discovering explicitly some form of knowledge embedded in data. In this paper, we try to make a synthesis or state of art of some important regular structures inside databases. We also ask several questions which have not specific answers. The reason is that some problems are not yet solved. In this work, we would like to invite other researchers to think about their solutions. We are more concerned with the application of relational algebra in solving of some problems in information engineering, than with proving new theorems related to relational algebra. More specifically, we generalize the difunctional relation canonical decomposition proposed in [7] to more general relations. For that purpose, we give an approximate algorithm for the canonical relation decomposition and we exploit Riguet's difunctional relation for the same purpose.

Functional [3] and difunctional dependencies [7] represent some invariant structures we could find in databases that are used for decomposing a database into some sub-schemas for the purpose of minimizing redundancies. These dependencies are general and do not depend on any particular database instance. However, even after such decomposition, instances of database may contain hidden regularities, that we can only discover by looking to the lattice of concepts [4, 11] embedded inside each specific instance. From this lattice [11], it is known that we can extract some association rules. However, these rules are not general, because if we change the database instance, the extracted knowledge from the database may also change.

As an illustration, we use the main concepts for automatic text summarization. We first decompose the text into some elementary sub-texts: (sentences, parts of sentences or sections). Second, we create a binary relation by indexing each elementary piece of text by non empty words included in each elementary sub-text. A possible tested approach for summarization is based on the main concept of the text (i.e. associating the maximum of sentences to the maximum of shared indexing words). In this paper we explain why the strength of this association is measured by the gain of the notion of "optimal concept". By this way, in [12], we have developed a system for generating different summaries each one associated with a different optimal concept. Another important problem concerns the design of good information filters in the search engines, each time we have to search for web pages sharing specific indexing words. A possible modeling of this problem is also to create a binary relation associating for each web page a list of shared indexing words. A conceptual coverage of a binary relation, may be used as a base for extracting the main web page references from the total space of web page references. By this way, the user will receive only different levels of clusters of web pages in decreasing importance level.

In our work, after recalling some known regular structures in a database, we present the conceptual decomposition technique. Among these regular structures, we first define difunctional and functional dependencies and explain their applications. As a second kind and more general and diverse regularities, we define maximal rectangles (or concepts) as the atomic information we may extract from any binary relation. All these regular structures are now used for data mining. We will explain this procedure in Section 6. In Section 4, we propose an approximate algorithm for an approximate coverage of a binary relation with optimal concepts. In Section 5, we give a solution to put in equation the coverage of some kind of binary relations by a minimal number of concepts using Riguet's difunctional relation. Using difunctionality of relations as described by Riguet, we develop an algorithm to find a canonical decomposition of some classes of binary relations. We then explain that we need to generalize the solution of the problem for more complex binary relations.

2 Definition of Regularities

We use relational algebra for formalizing the data space and regularities we may discover embedded in these data. We assume that we are able to map our data into a binary context (i.e. a subset of the Cartesian product of two sets: E the set of objects and P the set of properties). This hypothesis is not too restrictive, because we noticed that most databases may be considered as a binary relation after some transformations. We also apply the proposed work on some available data from the internet or documentary databases or tabular data that are available in most of professional companies. For this kind of databases, we can directly obtain a binary relation linking document (of a web page) references to indexing terms. So all dependencies extracted between the terms of the documentary database give additional information for users. For all these general cases, we always need to extract regular associations between attributes to make the right decision in case of similar situations. We also may exploit regularities to filter information, to keep only a minimal data size. This kind of application is very useful for search engines to only give few pertinent web page references corresponding to the user query.

2.1 Functional Dependencies and Their Application

A functional dependency (fd) is the dependency most frequently used in practice, since the development of Codd's relational model [3]. Functional dependencies have been used to minimize redundancy and to normalize the relational database schema. The universe U of a relational schema is composed of a set of attributes. Each attribute A has a domain $dom(A)$. An element of $dom(A)$ is denoted by a, b, etc. We use capital letters as A, B for single attribute, and X, Y for subsets of attributes. The union of two subsets X and Y is written as XY. We also make the difference between a single attribute A and the set $\{A\}$. A relation s defined on the set of attributes A_1,A_2,...,A_n is a subset of the Cartesian product $dom(A_1) \times ... \times dom(A_n)$. We say that s is an instance of the relational schema $S(U)$. A tuple is an element of s, called also a vector or a sequence of n values associated with the n attributes. For example if $t = (4, 2, 6)$ is a tuple of relation s defined on the relational schema $S(A, B, C)$ then $t[AC] = (4, 6)$ while $t[A] = 4$. Generally $t[X]$ is the restriction of t to the subset of attributes X.

Definition 1 (Functional dependency). *Let X and Y be two subsets of attributes of the universal set U. We say that it exists a functional dependency* (fd) *from X to Y if and only if, for any instance s of the relational schema $S(U)$, if t_1 and t_2 are any two tuples of s, if $t_1[X] = t_2[X]$ then $t_1[Y] = t_2[Y]$. We generally use the notation $X \rightarrow Y$.*

Several properties of these dependencies have been defined by Codd, as follows:

- Reflexivity rule: If $Y \subseteq X$, then $X \rightarrow Y$
- Augmentation rule: If $X \rightarrow Y$ then $XZ \rightarrow YZ$
- Transitivity: $X \rightarrow Y$ and $Y \rightarrow Z$, then $X \rightarrow Z$.
- If $X \rightarrow Y$ then X is by definition a key in the relational schema $S[X \cup Y]$.

2.2 Difunctional Dependencies

Difunctional dependencies [7, 13] are a generalization of functional dependencies. Let R be a binary relation. R is *difunctional* if and only if $R \circ R^{-1} \circ R = R$. Where "$\circ$" is the operator for relational composition, and R^{-1} is the inverse of R. A difunctional is no more no less than the union of the Cartesian product of several pairs of subsets which are disjoint in their domains and their codomains. This general relational equation has been used intensively in software engineering and proved to be a very frequent data structure specifying the link between inputs and outputs. It has been ignored for a long time in data engineering. Its utility has been shown by name correct?? Nlt Lethan and Jaoua between 1985 and 1992, under different names (iso-dependencies or regular relations) [7].

Definition 2 (Difunctional dependencies). *We say that there is a difunctional dependency between X and Y, denoted by $X \leftrightarrow Y$, if and only if, for any instance s of $S(U)$, the binary relation $R[X, Y]$ defined by $s[X, Y]$ is difunctional.*

Example 1. Consider $U = A, B, C$, and s the following instance of $S(U)$ in Table 1. We can see that $A \leftrightarrow B$ is true for s, because the binary relation $R[A, B]$ is difunctional; on the contrary $B \leftrightarrow C$ is false.

Table 1. An instance of $s(U)$

A	B	C
2	3	5
2	4	5
3	3	5
3	4	8

Redundancy reduction. Consider a relational schema $S(U)$ and any instance s of S. Assume that for any s we associate a difunctional binary relation $R[X, Y]$ (with $X \cup Y = U$), which is the union of maximal rectangles whose projections are disjoint. So:

$$R[X, Y] = (A_1 \times B_1) \cup (A_2 \times B_2) \cup ... \cup (A_i \times B_i) \cup ... \cup (A_n \times B_n)$$
$$\text{with } A_i \cap A_j = B_i \cap B_j = \phi, \forall i \neq j.$$

It is easy to see that we can reduce redundancy by decomposing $R[X, Y]$ into two binary relations $R_1[X, C]$ and $R_2[C, Y]$, where C is the attribute class. In $R_1[X, C]$, for each element of the set A_i we associate the value i. In $R_2[C, Y]$, for each value j, we associate all elements of subset B_j.

This kind of decomposition is called "canonical decomposition". Experimentations on several databases have shown that we can save an important amount of memory space by such a decomposition. Even when we don't find difunctional dependencies, we discovered that most of instances of a database contain a uniform part which has a difunctional structure.

Even more general than functional dependencies and generalized to fuzzy difunctional dependencies [13], this kind of dependency has not been directly useful in database, because in most cases, attributes in databases do not have such a uniform structure. But, the most important thing exhibited by a difunctional relation is the notion of concept which is the maximal Cartesian product included in a database, also called maximal rectangle, and rectangular binary relation decomposition [1]. We will discuss this question in the next section.

2.3 Conceptual Dependencies

Definition 3. *Let R be a binary relation defined on a set E. The relation $A \times B$, such that $A \subseteq E$ and $B \subseteq E$, is called rectangular relation (or rectangle) of R [6, 14, 15]. A is the domain of this relation and B is its codomain (or its range).*

In Figure 1, we can see an example of a rectangular relation:

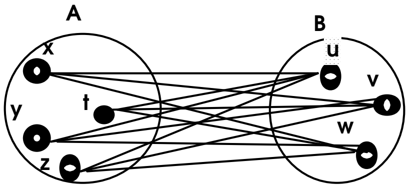


Fig. 1. Rectangular relation *RE* with 12 pairs

Definition 4. *From a memory storage space perspective, the gain which is associated to a given rectangular relation $RE = A \times B$ is assessed by the following heuristic function:*

$$g(RE) = (\parallel A \parallel \times \parallel B \parallel) - (\parallel A \parallel + \parallel B \parallel) \tag{1}$$

where $\parallel A \parallel$ denotes the cardinality of the set A.

Remark 1. This definition is introduced in [2]. We explain this formula by the fact that a rectangular relation (or rectangle) associates $\parallel A \parallel$ values to $\parallel B \parallel$ values. So, when we cluster in one side all the values of A, and in the other side all the values of B, we can replace $\parallel A \parallel \times \parallel B \parallel$ direct pairs of RE (Figure 1) by $\parallel A \parallel + \parallel B \parallel$ indirect pairs by using an intermediary element i also called extra-symbol which links any element of A to any element of B as illustrated by Figure 2.

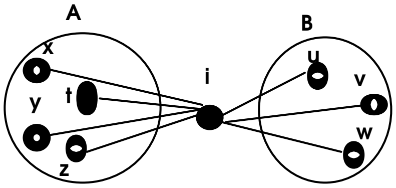


Fig. 2. A resumed rectangular representation of *RE*

Definition 5. *A rectangle $RE = A \times B$ which contains an element (a,b) of a binary relation R is said to be optimal if it realizes the maximal gain g(RE) among all rectangles which contain (a,b).*

Remark 2. Searching for the optimal rectangle containing (a, b) is an NP-complete problem [1, 5]. Several heuristics which are based on a branch and bound principle have been implemented and applied for database decomposition [2], object oriented system decomposition [6] and data mining [17].

Remark 3. Optimal rectangles have a particular meaning because it represents the most important data associations. Several rectangles may be optimal, because they realize the same maximal gain. So with respect to some equivalence relation, we can assimilate the class of all rectangles with the same gain to only one representative element.

Definition 6. *[2] A rectangular relation (or rectangle) $RE = A \times B$ is said degenerate if and only if $\| A \| = 1$ (Figure 3a) or $\| B \| = 1$ (Figure 3b).*

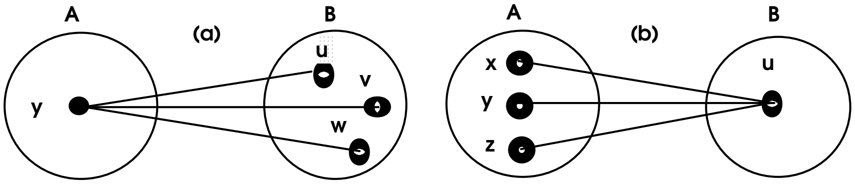


Fig. 3. Examples of degenerate rectangles

A concept is a maximal rectangle (i.e. a rectangle that cannot be extended simultaneously in the domain and in the codomain). Assume that you have a binary context R. We are always able to extract all concepts included in R. Wille proved in [16], that this set of concepts is a complete lattice. This lattice structure has been used intensively for knowledge extraction from data (i.e. dependencies between attributes or association rules between the terms contained in a documentary data base or in a single document). Importance of the notion of concept has been discovered by the scientific groups working on graph theory. Starting from 1990, we applied it to extract knowledge from data. Another group on relational algebra discovered applications of concepts for software and data decomposition [9], for machine learning [8], text summarization and several other applications. Because of its simple and uniform structure, we believe more and more that an atomic information is something like a directed pair of two subsets (i.e. a complete bipartite sub-graph). So we assume that the data are composed of a set of concepts.

2.4 Gain of a Binary Relation

The gain in $W(R)$ of binary relation R is given by:

$$W(R) = (\frac{r}{d \times c}) \times (r - (d + c)) \tag{2}$$

Where:

- r is the cardinality of R (i.e. the number of pairs in R)
- d is the cardinality of the domain of R
- c is the cardinality of the range of R

Remark 4. The quantity $\frac{r}{d \times c}$ provides a measure of the density of the relation R. The quantity $r - (d + c)$ is a measure describing how economical information is represented. It is a logical extension of the corresponding definition from a concept to a general relation. This definition will be used in the proposed heuristic in Section 4.

2.5 Elementary Relation (noted PR)

If R is a finite binary relation (i.e., subset of $E \times F$, where E is a set of objects and F a set of properties) and $(a, b) \in R$, then the union of rectangles containing (a, b) is the elementary relation PR (i.e. subset of R) given by:

$$PR = \Phi_R(a, b) = I(b.R^{-1}) \circ R \circ I(a.R) \tag{3}$$

where:

- I is the identity relation.
- R^{-1} is the inverse relation of R (i.e. set of inverted pairs of R).
- "$\circ$" refers to the relative product operator, where:

$$R \circ R' = \{(x, y) | \exists z : (x, z) \in R \wedge (z, y) \in R'\} \tag{4}$$

Let $A \subseteq E$, then $I(A) = \{(a, a) | a \in A\}$.
PR is the sub-relation of R, pre-restricted by the antecedents of b (i.e. $b.R^{-1}$), and post-restricted by the set of images of a (i.e. $a.R$). In the next section, we use such elementary relations PR to find the coverage of a relation by some "minimal" number of optimal concepts. Note that the problem is NP-complete. For that reason, we will only propose an approximate solution in Section 4, based on a greedy method using the gain function W.

3 Conceptual Binary Relation Coverage and Canonical Decomposition

We may consider any binary relation R as the union of concepts. The problem is that among the different possible combinations of concepts covering R, we have to select the most economical ones in terms of memory. Finding the minimal coverage of R is an NP-complete problem [5]. For that reason, in [1, 6], we used some approximate algorithms to decompose huge binary relations based on the function "gain" given in Definition 4. The problem of finding the optimal rectangle with a maximum "gain" is also NP-complete [5]. For that reason, we think that in the future, we should make more research investigations about formal properties of such coverages to find better approximate methods. An open problem is to find new efficient algorithms to update an initial conceptual coverage of some binary relation R when we add or remove some pairs in R. These researches will have an impact on conceptual data mining systems.

Assume that $\{A_1 \times B_1,\ A_2 \times B_2,...,A_p \times B_p\}$ is some minimal coverage of the binary relation R (i.e. $R = A_1 \times B_1 \cup A_2 \times B_2 \cup ... \cup A_p \times B_p$). If we define the two following operators:

$$f(R) = A_1 \times \{c_1\} \cup A_2 \times \{c_2\} \cup ... \cup A_p \times \{c_p\} \tag{5}$$

$$g(R) = \{c_1\} \times B_1 \cup \{c_2\} \times B_2 \cup ... \cup \{c_p\} \times B_p \tag{6}$$

generally the number of pairs in R is much higher than the number of pairs in $f(R) \cup g(R)$, while $R = f(R) \circ g(R)$. Here $\{c_1, c_2, ..., c_p\}$ are extra-symbols different from any element in the domain or range of R, which are created to represent the different concepts in R. Another open problem is related to an incremental conceptual binary relation transformation: the question is to find an efficient method to calculate $f(R \cup \{a, b\})$, $g(R \cup (\{a, b\})$, $f(R - \{a, b\})$, $g(R - \{a, b\})$ using only $f(R)$ and $g(R)$. The objective is to continue to update the conceptual coverage of R using its minimal representation by the two relations $f(R)$ and $g(R)$, by removing or adding the minimal number of extra-symbols. Operators f and g might be defined automatically by some relational operator. It is even interesting to give options for specific functions with interesting properties we could use for mapping binary relations to their canonical forms. Using this kind of decomposition in many experimental databases, we saved a huge amount of memory space. In the following two sections, we first propose an approximate solution (Section 4), and second the difunctional of Riguet (Section 5) for deriving a coverage of a binary relation with optimal concepts.

4 Approximate Algorithm for Canonical Decomposition

In this section we propose an approximate algorithm to find a set of optimal rectangles that provides a coverage of a given relation R: an approximate solution for a canonical decomposition of a binary relation. The algorithm is explained in Figures 9 and 10. But here we explain the steps using the following relation R. Let R be a finite binary relation between two sets as illustrated below in Figure 5:

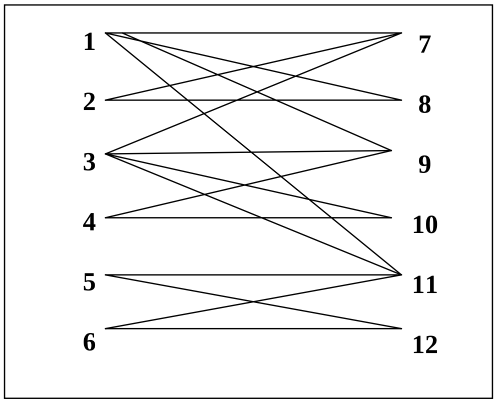


Fig. 4. An example of a binary relation R

- Step 1: Divide the relation R into disjoint sub-relations ,..., Here we have only one sub-relation (also called elementary relation).
- Step 2: For each elementary relation PR_i, search the optimal rectangle, which includes an element of PR_i.

If PR_i is a rectangle, then it is an optimal rectangle containing (a, b), else check if PR_i contains other elements (X, Y) in the form (a, Y) or (X, b) by trying all the images of a and all the antecedents of b (see Figure 6).

$PR(1, 7) = \Phi_R(1, 7) = I(7.R^{-1}) \circ R \circ I(1.R)$

So we search with an iterative way the optimal rectangles of $PR\ (1, 7)$ which successively contain the elements $(1, 8)$, $(1, 9)$, $(1, 11)$, $(2, 7)$, and $(3, 7)$.

- First Iteration: from the five elementary relations of the above mentioned elements select the first that gives a maximal gain W defined in Section 2. As a matter of fact the relation with the maximum gain represents the best compromise between density, and information economy.

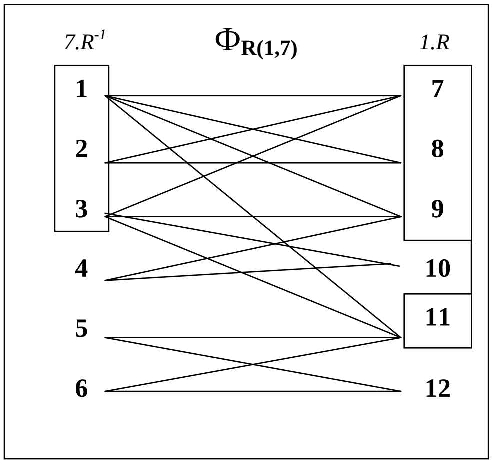


Fig. 5. Elementary relation $\Phi_{R(1,7)} = PR(1,7)$

1. $PR'_{1,8} = \Phi_{PR_{1,7}}(1,8); W(PR'_{1,8}) = 0$
2. $PR'_{1,9} = \Phi_{PR_{1,7}}(1,9); W(PR'_{1,9}) = 7/8$ ✓ Selected
3. $PR'_{1,11} = \Phi_{PR_{1,7}}(1,11); W(PR'_{1,11}) = 7/8$
4. $PR'_{2,7} = \Phi_{PR_{1,7}}(2,7); W(PR'_{2,7}) = 0$
5. $PR'_{3,7} = \Phi_{PR_{1,7}}(3,7); W(PR'_{3,7}) = 7/9$

The selected elementary relation $PR'_{1,9}$ is not a rectangle, so the algorithm continues on the already selected elements i.e. (1,7) and (1,9) as shown in Figure 7.

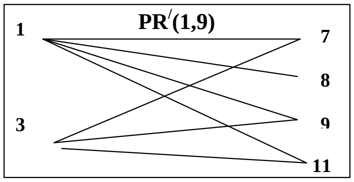


Fig. 6. Elementary relation $PR'_{(1,9)}$

- Second Iteration: Search now the optimal rectangles of $PR'_{(1,9)}$ that successively contain elements $(1,8)$, $(1,11)$, and $(3,9)$. This step provides three elementary relations:

1. $PR''_{1,8} = \Phi_{PR'_{1,9}}(1,8); W(PR''_{1,8}) = -1$
2. $PR''_{1,11} = \Phi_{PR'_{1,9}}(1,11); W(PR''_{1,11}) = 7/8$
3. $PR''_{3,9} = \Phi_{PR'_{1,9}}(3,9); W(PR''_{3,9}) = 7/8$ ✓ Selected

$PR''_{3,9}$ is a rectangle, so it is an optimal one that contains element (1,7) of R. The following Figures 8 and 9 illustrate the iterations of searching the optimal rectangle.

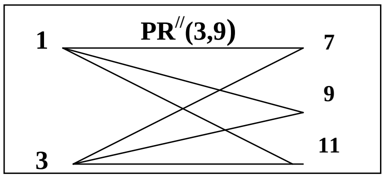


Fig. 7. Elementary relation $PR''_{(3,9)}$

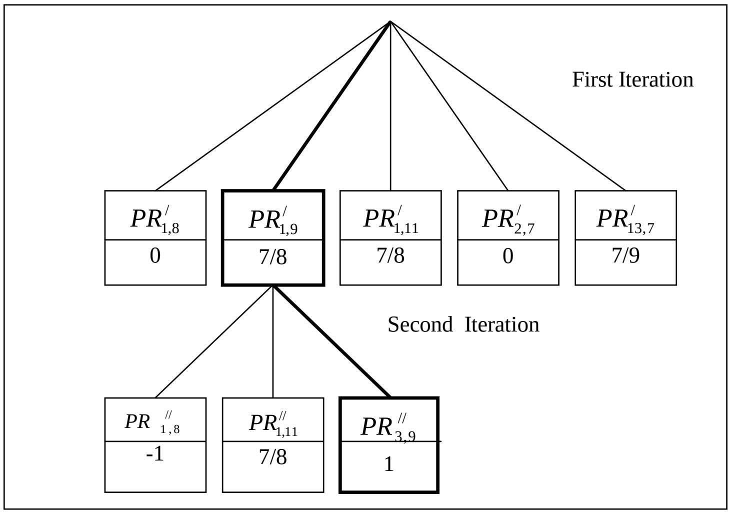
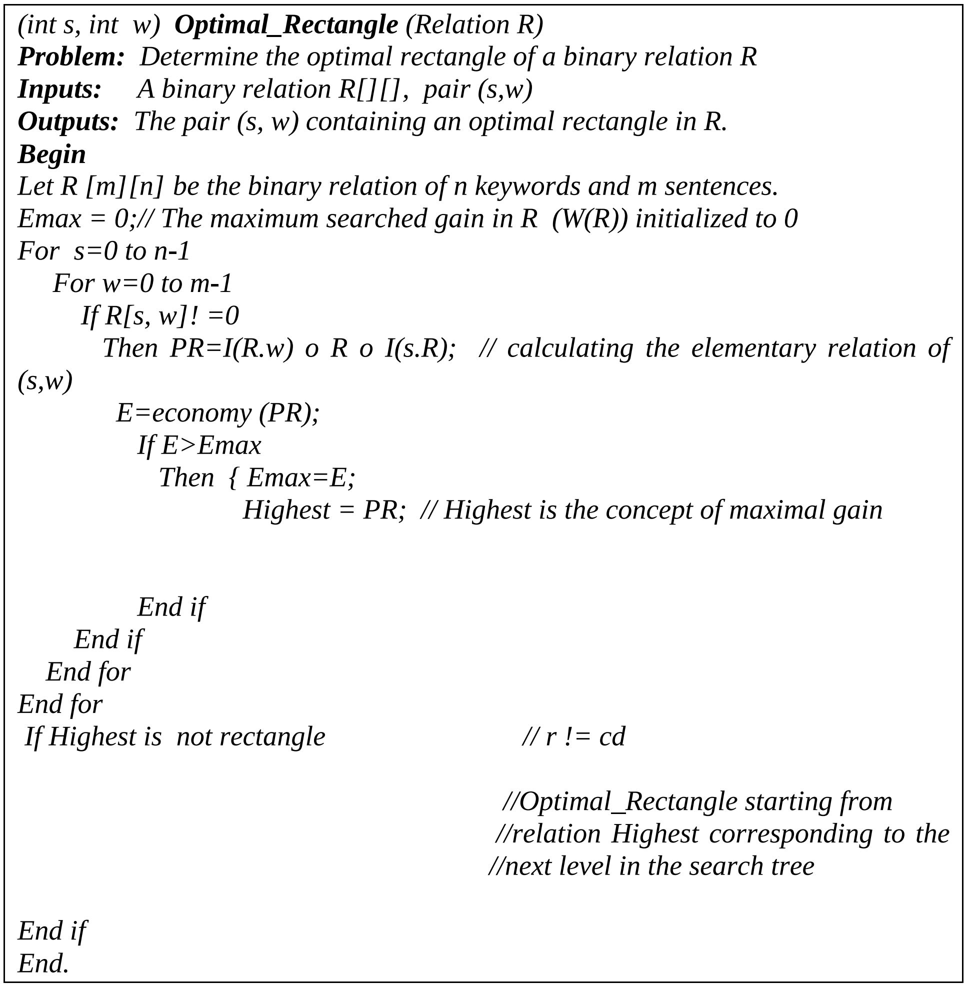


Fig. 8. The search tree for optimal concept

In bold you can see the selected elementary relation at each level of the search tree. Each level is associated with an iteration in the proposed algorithm. The proposed algorithm is polynomial (Figure 10 and 11). When we find an optimal rectangle, we continue to search for a next optimal one containing another pair not already selected. Here if we select the pair (6,12), we find at the first iteration the concept: $PR_{6,12} = 5, 6 \times 11, 12$. Then if we select the pair (4,10), we obtain the concept: $PR_{4,10} = 3, 4 \times 9, 10$. Finally, if we select the pair (2,8), we obtain the concept: $PR_{2,8} = 2, 1 \times 7, 8$. The selected coverage is composed of: $\{PR''_{3,9}$, $PR_{6,12}$, $PR_{4,10}$, $PR_{2,8}\}$



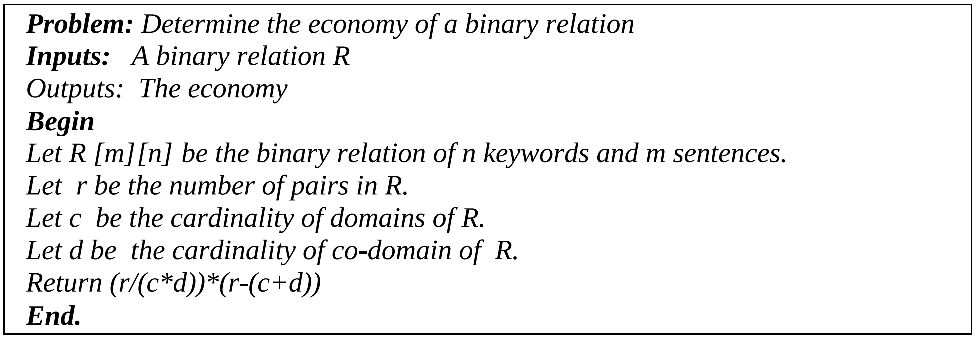
```
(int s, int  w)  Optimal_Rectangle (Relation R)
Problem:  Determine the optimal rectangle of a binary relation R
Inputs:     A binary relation R[][],  pair (s,w)
Outputs:  The pair (s, w) containing an optimal rectangle in R.
Begin
Let R [m][n] be the binary relation of n keywords and m sentences.
Emax = 0;// The maximum searched gain in R  (W(R)) initialized to 0
For  s=0 to n-1
    For w=0 to m-1
        If R[s, w]! =0
          Then  PR=I(R.w) o R o I(s.R);  // calculating the elementary relation of
(s,w)
            E=economy (PR);
              If E>Emax
                Then  { Emax=E;
                        Highest = PR;  // Highest is the concept of maximal gain


              End if
        End if
    End for
End for
 If Highest is  not rectangle                         // r != cd

                                                    //Optimal_Rectangle starting from
                                                   //relation Highest corresponding to the
                                                  //next level in the search tree

End if
End.
```

Fig. 9. Algorithm calculating an optimal rectangle in a binary relation R



```
Problem: Determine the economy of a binary relation
Inputs:   A binary relation R
Outputs:  The economy
Begin
Let R [m][n] be the binary relation of n keywords and m sentences.
Let  r be the number of pairs in R.
Let c  be the cardinality of domains of R.
Let d be  the cardinality of co-domain of  R.
Return (r/(c*d))*(r-(c+d))
End.
```

Fig. 10. Economy of a binary relation calculus

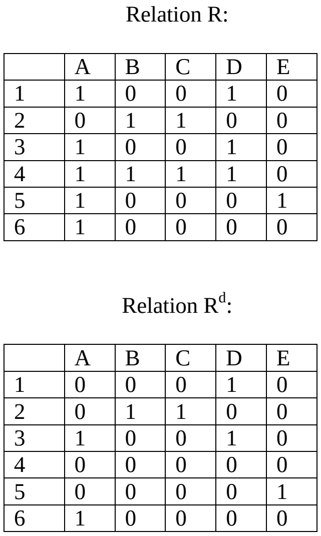
5 Relational Calculus for Conceptual Coverage Extraction

5.1 Relational Calculus with the Difunctional of Riguet

Is it possible to find a specific coverage of a binary relation R by a minimal number of concepts using relational methods? The answer is that this is possible if each concept of the coverage contains at least one isolated pair in R. An isolated element, by definition is an element which belongs to only one concept c in R. In this case, concept c belongs to any conceptual coverage of R. Fortunately, in [10] Khcherif, et al. proved that we can extract all existing isolated elements by calculating the following difunctional R^d proposed by Riguet in 1995:

$$R^d = \overline{R \circ \overline{R^{-1}} \circ R} \cap R \tag{7}$$

Here, $\overline{R}$ is the complement of R. From the domain D_i of each rectangle of R^d, we find a concept by using Galois connection operators f_1 and f_2, where $S_i = f_1(D_i)$ gives the set of all common images of D_i, $N_i = f_2(S_i)$, calculates all common antecedents of elements belonging to S_i with respect to relation R. The concept $N_i \times S_i$ is included in R and belongs to any possible conceptual coverage of R. In the following example in Figure 5, we can see how can we find the conceptual coverage of R:



Relation R:

	A	B	C	D	E
1	1	0	0	1	0
2	0	1	1	0	0
3	1	0	0	1	0
4	1	1	1	1	0
5	1	0	0	0	1
6	1	0	0	0	0

Relation R^d:

	A	B	C	D	E
1	0	0	0	1	0
2	0	1	1	0	0
3	1	0	0	1	0
4	0	0	0	0	0
5	0	0	0	0	1
6	1	0	0	0	0

Fig. 11. Calculation of R^d

Then from R^d, using Galois connection operators f_1 and f_2 on R, we find the following conceptual coverage of R by concepts C_1, C_2, C_3 and C_4 where $C_1 = \{1,3,4\} \times \{A,D\}$, $C_2 = \{2,4\} \times \{B,C\}$, $C_3 = \{5\} \times \{A,E\}$, and $C_4 = \{1,3,4,5,6\} \times \{A\}$.

5.2 An Open Problem

The relational calculus using the difunctional of Riguet gives the optimal coverage of only some kind of binary relations. The reason is that not all concepts of a binary relation contain isolated elements (i.e. some or all elements in R belong to more than one concept). In that case, we can remove initial isolated elements, from R, and then calculate R'^d in the remaining relation R'. We reiterate this last step until we find the coverage of R. The problem is that R^d may be empty. In that case, the problem is to find by some other relational calculation the most economical conceptual coverage of R.

6 Optimal Concepts and Applications

A machine learning system may be considered as a continual concepts reorganization. What do we mean by the central ideas we have in computer "mind"? How do we optimize storage space by continuously creating new symbols replacing unorganized associations between existing symbols? Here we could define several ways for associating new objects into the space of symbols. We generally associate new objects with the central idea which optimizes the total space storage. So learning may be considered as an optimization task, by looking for the maximum of stability obtained by always giving a priority to the most economical concepts. In the previous Sections 3 and 4, we have defined two operators f and g, to map a binary relation R into its economical form $(f(R), g(R))$. But here we can notice that f and g are not defined in the same direction. Because while f is associating with each element a of the domain of R to all the symbols representing all concepts to which a belongs, g is associating with each symbol c representing a concept C, all elements of the range of R belonging to the range of C.

7 Application of Canonical Decomposition for Text Summarization and Improving Search Engines

The idea is to extract a summary from a document. For that purpose, we first decide about how to decompose a text: into chapters or sections or sentences. Then we may ask users if he/or she wants to get automatically a summary or extract association rules from the text. Assume that the user decides to consider that a sentence is the atomic structure in the text that we are not allowed to change or reduce. The proposed system already implemented in January 2004, will first create a binary context R, where objects are sentence numbers (recognized by their position in the text) and properties are words (each word is also recognized by a position in a hash table). So by definition (i, j) belongs to R if and only if word number j belongs to sentence number i or is very similar to another word in sentence number i. If the word is empty we do not consider it. A table of empty words is first consulted. Now, the crucial question about how to recognize that two words are similar has been resolved with an approximate way. We assume that two words are similar if they contain a longest common sub-sequence with some relative size greater than some value p near to 1. Here, when we decrease the value of p, we obtain more similar words and of course this has some impact on the quality of the summary. As a next step, starting from the binary context R, we use the method proposed in the previous section to find the optimal concept. Then we select all sentences in the domain of this concept to generate a summary. If the user would like more precision about the document, he/she may ask about the next optimal concept, and obtain by the same way a complement of information. We can repeat this until covering the entire document. We realized a system for experimentation using many documents, and we are generally satisfied by the selected sentences. We think that our system is suitable to provide several improvements. This same method may be used for improving search engines, by first selecting documents corresponding to the optimal concept.

8 Conclusion

Most of the research on relational methods in data mining should concentrate on studying different properties of regular structures in binary relations. Algorithms related to graph theory about incremental conceptual restructuring should also be improved to use as a model for machine learning and classification. Properties of operators f and g defined in Section 3 should be studied in depth in the future to give fundamental bases for database organization, for improving the quality of the current search engines by structuring information. An important question is to find the canonical decomposition of $f(R)$ and $g(R)$. This generalization needs to find different heuristics for economical decomposition, as for example by associating a weight to extra-elements, very probably equal to the gain of the concept they are representing. Canonical decomposition may be generalized to fuzzy concepts, to deal with imprecision. In the future, we need to discover some hidden invariant rules i.e. holding even if we change the database instance. Relational studies must be investigated to find more efficiently the common association rules of different data instances with incremental approaches. Cooperative information retrieval and knowledge extraction need more and more studies about regular structures using intersection, union or join merging operators [13]. The question is now to study different kinds of interactions between these concepts (i.e, operations as union, intersection, or composition). Also, assume that you want to merge arriving concepts from different sides, how do we reorganize the space of concepts? We should be able to organize it incrementally into a minimal number of merged and transformed new concepts. If we assume that our data is organized as a union of equally overlapped concepts, is there a mathematical relational structure more general than difunctional relations? What are the main categories of a uniform space of concepts? Finally, is it possible to consider that thinking is a continual reorganization of regular structures into other optimized regular concepts?

Acknowledgment: *We would like to thank all Anonymous Reviewers for their precious comments.*

References

1. K. Arour, A. Jaoua, H. Ounelli, and N. Belkhiter. Rectangular decomposition of n-ary relations. In *Proc. of the 7th Siam Conference on Discrete Mathematics, Albuquerque, Nouveau Mexique*, june 1994.
2. N. Belkhiter, C. Bourhfir, M. M. Gammoudi, A. Jaoua, N. Lethan, and N. Reguig. Décomposition rectangulaire optimale d'une relation binaire: application aux bases de données documentaires. *INFormation Systems and Operational Research Journal*, 32(1):33–54, 1994.
3. C. J. Date. *An Introduction to Database Systems Vol I.* Addison Wesley, 1987.
4. B. Ganter and R. Wille. *Formal Concept Analysis.* Springer-Verlag, Heidelberg, 1999.
5. M. R. Garey and D. S. Johnson. *Computers and Intractability: A guide to the theory of NP-Completness.* W. H. Freeman, 1979.
6. A. Jaoua, J. M. Beaulieu, N. Belkhiter, A. C. Debaque, J. Desharnais R. Lelouche, T. Moukam, and M. Reguig. Rectangular decomposition of object-oriented software architectures. In *Proc. of the 14th Int. Conf. on Soft. Eng. (ICSE 14), Melbourne, Australia*, Mai 1992.
7. A. Jaoua, N. Belkhiter, and T. Moukam. Propriétés des dépendances difonctionnelles dans les bases de données relationnelles. *INFormation Systems and Operational Research Journal*, 30(1):297–316, 1992.
8. A. Jaoua and S. Elloumi. Galois connection, formal concept and Galois lattice in real binary relation: Applications in a real classifier. *Journal Systems and Software*, 60(2):149–163, March 2002.
9. A. Jaoua, H. Ounelli, and N. Belkhiter. Automatic Entity Extraction From an N-ary Relation: Towards a General Law for Information Decomposition. *Journal Systems and Software*, pages 216–232, November 1995.
10. R. Khcherif, M. Gammoudi, and A. Jaoua. Using Difunctional Relations in Information Organization. *Information Science*, 1-4(125):153–166, June 2000.
11. G. W. Mineau and R. Godin. Automatic Structuring of Knowledge Bases by Conceptual Clustering. *IEEE Transactions On Knowledge and Data Engineering*, 7(5):824–829, 1995.
12. T. Mosaid, F. Hassen, and H. Salah. Conceptual Text Summarization. Senior project, University of Qatar, 2004.

13. H. Ounelli and A. Jaoua. On Fuzzy Difunctional Relations. *Journal of Information Sciences*, (95):216–232, 1996.
14. J. Riguet. Relations binaires, fermetures et correspondances de Galois. 76:114–145, 1948.
15. G. Schmidt and Ströhlein. *Relations and Graphs.* Springer Verlag, 1989.
16. R. Wille. Restructuring lattice theory : an approach based on hierarchies of concepts. In *Proc. of Nato Advanced Study Institute, Ed. by I. Rival, Reidel Publ. Dordrecht*, volume 81, pages 445–470, 1982.
17. S. Ben Yahia, K. Arour, Y. Slimani, and A. Jaoua. Discovery of Compact Rules in Relational Databases. *Information Science Journal*, 4(3):497–511, October 2000.

Journal on Relational Methods in Computer Science, Vol. 1, 2004, pp. 217 - 234
Received by the editors April 29, 2004, and, in revised form, October 21, 2004.
Published on December 10, 2004.

This article may be accessed via WWW at http://www.jormics.org.