愛知工業大学研究報告
第27号B 平成4年

Using Relaxation Techniques to Evaluate Queries in Deductive Databases

緩和法による演繹データベースの問い合わせ評価

Susumu Suzuki[†], Toshihide Ibaraki[††], Masahichi Kishi[†]
鈴木 晋, 茨木 俊秀, 岸 政七

ABSTRACT *Relaxation method is a general framework used to improve the efficiency of answering a query $q(\mathbf{a},\mathbf{x})$ given to a deductive database P. It first solves problem $(q'(\mathbf{a}',\mathbf{x}'),P^{RLX})$, where P^{RLX} is a relaxation of the original database P and $q'(\mathbf{a}',\mathbf{x}')$ is the modified query to P^{RLX}, to derive a set $PREL$ of predicate occurrences that is known to contain the answer set ANS in P, and construct database P^{MDF} by augmenting P with the restriction that solution space is constrained to $PREL$, and finally solves problem $(q(\mathbf{a},\mathbf{x}),P^{MDF})$ to get the desired answer set ANS. If the relaxation P^{RLX} is properly defined, $(q'(\mathbf{a}',\mathbf{x}'),P^{RLX})$ can be efficiently solved since P^{RLX} is simpler than P, and $(q(\mathbf{a},\mathbf{x}),P^{MDF})$ can also be efficiently solved as the solution space is restricted. Several methods are proposed to construct such relaxations. It is also argued that the original form of magic set method [2] can be described in the context of the relaxation method.*

1 Introduction

Given a query $q(\mathbf{a},\mathbf{x})$, where $\mathbf{a}$ is a vector of constants and $\mathbf{x}$ is a vector of variables, in a deductive database $P=(R,F)$, specified by a set of Horn rules R and a set of facts F, we consider to derive the set of all answers ANS to the query. It is assumed that each rule in R is range restricted (i.e. every variable in the head of a rule appears in its body) and includes no negation, and every argument of each predicate is not a function. Let $N(P)(=\{p,q,r,\ldots\})$ be the set of all predicate names in P, $C(P)(=\{a,b,c,\ldots\})$ the set of all constants in P, $PO(N(P),C(P))$ $(=\{p(a,\ldots,a),p(a,\ldots,b),\ldots,q(a,\ldots,a),$ $q(a,\ldots,b),\ldots\})$ the set of all forms obtained by substituting constants to the arguments of predicates. We call each element in $PO(N(P),C(P))$ a predicate occurrence (abbreviated to po).

† 愛知工業大学 情報通信工学科 (豊田市)
†† 京都大学 工学部 数理工学科 (京都市)

Note that a po $p(\mathbf{a})$ may not be a fact of an extensional predicate defined in F. A form $p(\mathbf{c},\mathbf{y})$, where $\mathbf{c}$ is a constant vector and $\mathbf{y}$ is a variable vector, is also called po, and stands for all po $p(\mathbf{c},\mathbf{a}),p(\mathbf{c},\mathbf{b}),\ldots$ obtained by substituting constant vectors for $\mathbf{y}$.

From now on, unless otherwise specified, constants are denoted by $a,b,c,\ldots$, constant vectors by $\mathbf{a},\mathbf{b},\mathbf{c},\ldots$, variables by $x,y,z,\ldots$, variable vectors by $\mathbf{x},\mathbf{y},\mathbf{z},\ldots$, EDB predicates(i.e. extensional database predicates used in F) by $A,B,C,\ldots$, and IDB predicates(i.e. intensional database predicates not in F) by $p,q,r,\ldots$. Let $IMP(P)(\subseteq PO(N(P),C(P)))$ denote the set of all predicate occurrences that can be deduced from P. Then the answer to a query $q(\mathbf{a},\mathbf{x})$ is the set:

$$ANS=\{q(\mathbf{a},\mathbf{x})\mid q(\mathbf{a},\mathbf{x})\in IMP(P)\}.$$

Many methods have been proposed for efficient evaluation of queries in deductive databases [2] - [8], [10] - [14]. To reduce the number of intermediate predicate occurrences in $IMP(P)$ generated to answer a query is an important idea to improve efficiency.

Definition: If some derivation tree that derives a po $s(\mathrm{d}) \in IMP(P)$ includes a po $p(\mathrm{z})$ as one of its nodes, $p(\mathrm{z})$ is said to be relevant to $s(\mathrm{d})$ (expressed as $(p(\mathrm{z}), P) \rightarrow *s(\mathrm{d})$). If $p(\mathrm{z})$ is relevant to some answer $q(\mathrm{a}, \mathrm{b})(\in ANS)$ to a query $q(\mathrm{a}, \mathrm{x})$, $p(\mathrm{z})$ is relevant to query $q(\mathrm{a}, \mathrm{x})$. The set of relevant po's to a query $q(\mathrm{a}, \mathrm{x})$, $REL(q(\mathrm{a}, \mathrm{x}), P)(\subseteq IMP(P))$, is therefore defined by

$$REL(q(\mathrm{a}, \mathrm{x}), P) = \{p(\mathrm{z}) \mid (p(\mathrm{z}), P) \rightarrow * \; q(\mathrm{a}, \mathrm{x})\}.$$

Set $REL(q(\mathrm{a}, \mathrm{x}), P)$ is usually referred to as the relevant set. □

In other words, the relevant set consists of all predicate occurrences that can be used to derive answers, among those deducible from P. Therefore, if we can know set $REL(q(\mathrm{a}, \mathrm{x}), P)$ in advance, we can restrict the search of answers only within the relevant set, thereby improving the efficiency of the answering process. However, as obvious from definition, computing the relevant set is equivalent to giving the exact answer set ANS, and therefore we try to generate sets of po, $PREL$, which is a relaxation of $REL(q(\mathrm{a}, \mathrm{x}), P)$, i.e., satisfies

$$\text{relaxation condition:} \quad PREL \supseteq REL(q(\mathrm{a}, \mathrm{x}), P).$$

This set of predicate occurrences $PREL$ is called a potentially relevant set. The original database P is then solved for the query $q(\mathrm{a}, \mathrm{x})$ under the additional constraint that only those predicate occurrences contained in $PREL$ are generated during its process of searching the set of answers ANS. The notion of relevant facts was first introduced in [7] and was explained in [3]. The above definition of a potentially relevant set is more general in the sense that not only facts but also predicate occurrences are taken into account. The efficiency of our method depends on the size of its potentially relevant set $PREL$.

We propose in this paper a general frameworks of relaxation method and give several methods to derive $PREL$, as well as examples that show its effectiveness. The well known magic set method [2], which eliminates irrelevant predicate occurrences by adding a restriction derived from the constants in a query to P, can also be discussed within the framework of the relaxation method, though there are some nontrivial differences in the details of computation process. [4] showed that a number of known methods could be defined within the framework of the generalized magic set method. It is not difficult to see that most of such methods can also be viewed as special cases of the relaxation method. There are however some cases of the relaxation method that are not regarded as the generalized magic set method, and some cases of the generalized magic set method that are not regarded as the relaxation method.

2 An Example

We give an example of a deductive database and a query, and derive its relevant set. This example is used throughout this paper.

Example 1: Let a deductive database $P1 = (R1, F1)$ consist of a rule set $R1 = \{r1, r2, r3, r4, r5\}$:

$$\begin{aligned}
r1 &: p(x_1, x_2, y_1, y_2) :- A(x_1, y_1), B(x_2, y_2). \\
r2 &: p(x_1, x_2, z_1, z_2) :- \\
&\quad p(x_1, x_2, y_1, y_2), A(y_1, z_1), B(y_2, z_2). \\
r3 &: s(x_1, x_2, y_1, y_2) :- C(x_1, y_1), D(x_2, y_2). \\
r4 &: s(x_1, x_2, z_1, z_2) :- \\
&\quad s(x_1, x_2, y_1, y_2), C(y_1, z_1), D(y_2, z_2). \\
r5 &: q(x_1, x_2, y_1, y_2) :- \\
&\quad p(x_1, x_2, y_1, y_2), s(x_1, x_2, y_1, y_2).
\end{aligned}$$

and a fact set $F1$:

$$\begin{aligned}
F1 = & \{A(i,j) \mid j = i+1, 51 \le i < 100, \\
& i : \text{integer}\} \\
\cup & \{A(i,i) \mid 51 \le i \le 100, i : \text{integer}\} \\
\cup & \{B(i,j) \mid j = i+1, 1 \le i < 100, \\
& i : \text{integer}\} \\
\cup & \{B(i,i) \mid 1 \le i \le 100, i : \text{integer}\} \\
\cup & \{C(i,j) \mid j = i+1, 1 \le i < 100, \\
& i : \text{integer}\} \\
\cup & \{C(i,i) \mid 1 \le i \le 100, i : \text{integer}\} \\
\cup & \{D(i,j) \mid j = i+1, 51 \le i < 100, \\
& i : \text{integer}\} \\
\cup & \{D(i,i) \mid 51 \le i \le 100, i : \text{integer}\}.
\end{aligned}$$

A query to P1 is

$$q(x_1, x_2, 75, 75).$$

This problem can be interpreted as follows. There are two vehicles Vp and Vs on grids of dimension two with axes 1 and 2. A po $A(x_1,y_1)(C(x_1,y_1))$ means that the first coordinate x_1 of Vp(Vs) can move directly to y_1, and $B(x_2,y_2)(D(x_2,y_2))$ means that the second coordinate x_2 of Vp(Vs) can move to y_2. Similarly, $p(x_1,x_2,y_1,y_2)$ $(s(x_1,x_2,y_1,y_2))$ means that the location (x_1,x_2) of Vp(Vs) in two-dimensional grid can reach location (y_1,y_2), and $q(x_1,x_2,y_1,y_2)$ means that both Vp and Vs, starting from (x_1,x_2), can reach (y_1,y_2). The query $q(x_1,x_2,75,75)$ therefore asks to compute the set of all initial locations (x_1,x_2), from which both Vp and Vs can reach $(75,75)$:

$$ANS1 = \{q(x_1,x_2,75,75) \mid q(x_1,x_2,75,75) \in IMP(P1)\}. \square$$

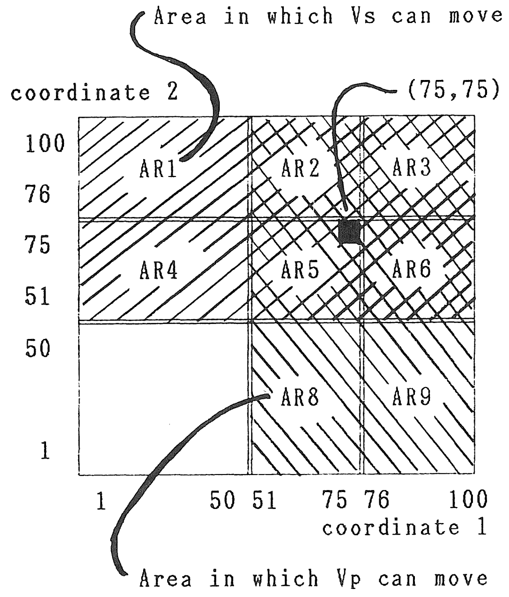

Fig.1 Two dimensional areas in which vehicles Vp and Vs can move

Fig.1 shows the areas in which Vp and Vs can move. Let $AR1,\dots,AR9$ be the areas as shown in Fig.1, respectively. Then, Vp can move in $AR2 \cup AR3 \cup AR5 \cup AR6 \cup AR8 \cup AR9$ and Vs in $AR1 \cup AR2 \cup AR3 \cup AR4 \cup AR5 \cup AR6$. It is easy to see that the answer set for this example is

$$ANS1 = \{q(x_1,x_2,75,75) \mid (x_1,x_2) \in AR5\}.$$

Let $IMP_p(P1)(\subset IMP(P1))$ be the set of predicate occurrences with predicate name p, which can be deduced from $P1$, and $REL_p(q(x_1,x_2,75,75),P1)$ $(\subset REL(q(x_1,x_2,75,75),P1))$ be the set of relevant predicate occurrences with predicate name p. $IMP_p(P1)$ in the above problem example is given by

$$IMP_p(P1) = \{p(x_1,x_2,y_1,y_2) \mid x_1 \le y_1, x_2 \le y_2, (x_1,x_2),(y_1,y_2) \in AR2 \cup AR3 \cup AR5 \cup AR6 \cup AR8 \cup AR9\}.$$

In order to derive REL_p, note that predicate occurrences of P that can be used in rule $r5$ to derive some answer $q(z_1,z_2,75,75)(\in ANS1)$ are $p(z_1,z_2,75,75)$ satisfying $(z_1,z_2) \in AR5$, and those that can be used in rule $r1$ or rule $r2$ to derive its predicate occurrences $p(z_1,z_2,75,75)$ ($(z_1,z_2) \in AR5$) are $p(x_1,x_2,y_1,y_2)$ satisfying $(x_1,x_2),(y_1,y_2) \in AR5$ and $x_1 \le y_1, x_2 \le y_2$. Therefore, we have

$$REL_p(q(x_1,x_2,75,75),P1) = \{p(x_1,x_2,y_1,y_2) \mid x_1 \le y_1, x_2 \le y_2, (x_1,x_2),(y_1,y_2) \in AR5\}.$$

3 Outline of Relaxation Method

Now, we will explain the idea of the relaxation method. The potentially relevant set $PREL$ becomes smallest when it is precisely equal to the relevant set REL. The relaxation method tries to compute a potentially relevant set $PREL$ that is close to REL, without spending too much computation time, by solving $(q'(\mathrm{a}',\mathrm{x}'),P^{RLX})$, where P^{RLX} is a relaxation of P and $q'(\mathrm{a}',\mathrm{x}')$ is the query $q(\mathrm{a},\mathrm{x})$ modified to meet the relaxation of P. Here a database P^{RLX} is called a relaxation of P, if there is a mapping f from the set of po's of P to the set of po's in P^{RLX} such that the following *relevancy condition* holds:

Let $q'(\mathrm{a}',\mathrm{x}') = f(q(\mathrm{a},\mathrm{x}))$. Then, for any $p(\mathrm{y}) \in REL(q(\mathrm{a},\mathrm{x}),P)$, $p'(\mathrm{y}')(= f(p(\mathrm{y})))$ belongs to $REL(q'(\mathrm{a}',\mathrm{x}'),P^{RLX})$.

The relaxation method then proceeds as follow.
(1) Solve $(q'(\mathrm{a}',\mathrm{x}'),P^{RLX})$ to obtain $REL(q'(\mathrm{a}',\mathrm{x}'),P^{RLX})$. Let $PREL = f^{-1}(REL(q'(\mathrm{a}',\mathrm{x}'),P^{RLX}))$.
(2) Solve $(q(\mathrm{a},\mathrm{x}),P)$ by restricting the domain of po's for search to set $PREL$ (i.e., ignoring all po's $\notin PREL$ generated in the computation). The database P restricted to $PREL$ is denoted by P^{MDF} and called a modified database of P. In other words, the answer set ANS of $(q(\mathrm{a},\mathrm{x}),P)$ can be obtained by solving $(q(\mathrm{a},\mathrm{x}),P^{MDF})$, say,

by the semi-naive method [1], i.e.,

$$\begin{aligned} ANS &= \{q(\mathrm{a},\mathrm{x}) \mid q(\mathrm{a},\mathrm{x}) \in IMP(P)\} \\ &= \{q(\mathrm{a},\mathrm{x}) \mid q(\mathrm{a},\mathrm{x}) \in IMP(P) \cap PREL\} \\ &= \{q(\mathrm{a},\mathrm{x}) \mid q(\mathrm{a},\mathrm{x}) \in IMP(P^{MDF})\} \end{aligned}$$

The relaxation method gives the correct answer set ANS since relaxation condition

$$PREL \supseteq REL(q(\mathrm{a},\mathrm{x}), P)$$

follows from the relevancy condition of f associated with relaxation P^{RLX}. The computation of the relaxation method is mostly spent on solving the relaxation $(q'(\mathrm{a}',\mathrm{x}'), P^{RLX})$ and the modified database $(q(\mathrm{a},\mathrm{x}), P^{MDF})$. Therefore, it is important how to define relaxations P^{RLX} which are easy to solve and yet are close approximations of the original P. In the following, we present two examples which are both based on the so-called argument elimination strategy.

4 Relaxation Method Based on Argument Elimination Strategy

From now on, we denote relaxation j of database $P1$ of Example 1 by $P1^{RLXj}$, the mapping from the set of po's of $P1$ to the set of po's in $P1^{RLXj}$ by fj, the po set $fj^{-1}(REL(q'(\mathrm{a}',\mathrm{x}'), P1^{RLXj}))$ by $PRELj$ and the modified database, which is database $P1$ restricted to $PRELj$, by $P1^{MDFj}$.

Example 2: A relaxation $P1^{RLX1} = (R1^{RLX1}, F1^{RLX1})$ is constructed from $P1$ of Example 1 by considering coordinate 1 only. First, the set of rules $R1^{RLX1}$ is given by

$$\begin{aligned} r6 &: p_1(x_1,y_1) :- A(x_1,y_1). \\ r7 &: p_1(x_1,z_1) :- p_1(x_1,y_1), A(y_1,z_1). \\ r8 &: s_1(x_1,y_1) :- C(x_1,y_1). \\ r9 &: s_1(x_1,z_1) :- s_1(x_1,y_1), C(y_1,z_1). \\ r10 &: q_1(x_1,y_1) :- p_1(x_1,y_1), s_1(x_1,y_1). \end{aligned}$$

The mapping $f1$ from the set of po's of $P1$ to the set of po's in $P1^{RLX1}$ is then defined by

$$\begin{aligned} f1: \quad & p(x_1,x_2,y_1,y_2) \rightarrow p_1(x_1,y_1), \\ & s(x_1,x_2,y_1,y_2) \rightarrow s_1(x_1,y_1), \\ & q(x_1,x_2,y_1,y_2) \rightarrow q_1(x_1,y_1), \\ & A(x_1,y_1) \rightarrow A(x_1,y_1), \quad B(x_2,y_2) \rightarrow \varepsilon, \\ & C(x_1,y_1) \rightarrow C(x_1,y_1), \quad D(x_2,y_2) \rightarrow \varepsilon, \end{aligned}$$

where ε means that $P1^{RLX1}$ does not have the corresponding EDB predicate in it. The set of facts in $P1^{RLX1}$ is similarly given by

$$\begin{aligned} F1^{RLX1} &= \{A(i,j) \mid j = i+1, 51 \leq i < 100, \\ &\quad i: \text{integer}\} \\ &\cup \{A(i,i) \mid 51 \leq i \leq 100, i: \text{integer}\} \\ &\cup \{C(i,j) \mid j = i+1, 1 \leq i < 100, \\ &\quad i: \text{integer}\} \\ &\cup \{C(i,i) \mid 1 \leq i \leq 100, i: \text{integer}\}. \end{aligned}$$

This relaxation $P1^{RLX1}$ describes a necessary condition of $P1$ in the sense that it characterizes the conditions concerned with the first coordinates, even though the vehicles move in the two dimensional plane. For example, if a po $p(x_1,x_2,y_1,y_2)$ can be deduced from $P1$, the corresponding po $p_1(x_1,y_1)(= f1(p(x_1,x_2,y_1,y_2)))$ is also deduced from $P1^{RLX1}$ (though the converse is not always true). Extracting only the first coordinate of query $q(x_1,x_2,75,75)$ for $P1$ gives query $q_1(x_1,75)(= f1(q(x_1,x_2,75,75)))$ for $P1^{RLX1}$. The relevant sets $REL_{p_1}(q_1(x_1,75), P1^{RLX1})$, $REL_{s_1}(q_1(x_1,75), P1^{RLX1})$, $REL_{q_1}(q_1(x_1,75), P1^{RLX1})$ become

$$\begin{aligned} & REL_{p_1}(q_1(x_1,75), P1^{RLX1}) = \\ & \quad \{p_1(x_1,y_1) \mid 51 \leq x_1 \leq y_1 \leq 75\} \\ & REL_{s_1}(q_1(x_1,75), P1^{RLX1}) = \\ & \quad \{s_1(x_1,y_1) \mid 51 \leq x_1 \leq y_1 \leq 75\} \\ & REL_{q_1}(q_1(x_1,75), P1^{RLX1}) = \\ & \quad \{q_1(x_1,75) \mid 51 \leq x_1 \leq 75\}. \end{aligned}$$

Therefore this $f1$ satisfies the relevancy condition:

$$\begin{aligned} & \{f1(r(\mathrm{y})) \mid r \in \{p,s,q\}, \\ & \quad r(\mathrm{y}) \in REL(q(x_1,x_2,75,75), P1)\} \\ & \subseteq REL(q1(x_1,75), P1^{RLX1}). \end{aligned}$$

($REL_p(q(x_1,x_2,75,75), P1)$ is given in section 2.) Let

$$\begin{aligned} PREL1 &= f^{-1}(REL_{p_1}(q_1(x_1,75), P1^{RLX1})) \\ &\cup f^{-1}(REL_{s_1}(q_1(x_1,75), P1^{RLX1})) \\ &\cup f^{-1}(REL_{q_1}(q_1(x_1,75), P1^{RLX1})), \end{aligned}$$

and the modified database, which is database $P1$ restricted to $PREL1$, be now given by $P1^{MDF1} = (R1^{MDF1}, F1^{MDF1})$, where its rule set $R1^{MDF1}$ is obtained by adding predicates representing the derived constraints to rules 1 - 5:

$$r11 : p(x_1,x_2,y_1,y_2) :- p_1(x_1,y_1),$$

$$\begin{array}{rll} & & A(x_1,y_1), B(x_2,y_2). \\ r12 & : & p(x_1,x_2,z_1,z_2) : -p_1(x_1,z_1), \\ & & p(x_1,x_2,y_1,y_2), A(y_1,z_1), B(y_2,z_2). \\ r13 & : & s(x_1,x_2,y_1,y_2) : -s1(x_1,y_1), \\ & & C(x_1,y_1), D(x_2,y_2). \\ r14 & : & s(x_1,x_2,z_1,z_2) : -s_1(x_1,z_1), \\ & & s(x_1,x_2,y_1,y_2), C(y_1,z_1), D(y_2,z_2). \\ r15 & : & q(x_1,x_2,y_1,y_2) : -q_1(x_1,y_1), \\ & & p(x_1,x_2,y_1,y_2), s(x_1,x_2,y_1,y_2). \end{array}$$

and its fact set $F1^{MDF1}$ is defined by

$$\begin{aligned} F1^{MDF1} = F1 & \cup REL_{p_1}(q_1(x_1,75), P1^{RLX1}) \\ & \cup REL_{s_1}(q_1(x_1,75), P1^{RLX1}) \\ & \cup REL_{q_1}(q_1(x_1,75), P1^{RLX1}). \end{aligned}$$

(Though relevant predicate occurrences $p_1(\mathbf{x},\mathbf{y})$, $s_1(\mathbf{x},\mathbf{y})$, $q_1(\mathbf{x},\mathbf{y})$ of $P1^{RLX1}$ are treated as facts of $P1^{MDF1}$ in this formulation, they can also be defined by rules.) Finally, the po set $IMP(P1^{MDF1})$ is generated by solving $P1^{MDF1}$ in the bottom up manner (e.g.,by the semi-naive method). Then, the following answer set $ANS1$ is derived.

$$ANS1 = \{q(x_1,x_2,75,75) \mid (x_1,x_2) \in AR5\}. \quad \square$$

In general, a pair (f, P^{RLX}) constructed by the the argument elimination strategy satisfies the relevancy condition, if the following conditions hold:

1. Every rule r in P^{RLX} is range restricted.
2. Let r be any rule of P and r' the relaxed rule in P^{RLX} corresponding to r. Then, for any IDB predicate $p(\mathbf{x})$ in rule r, rule r' has the corresponding IDB predicate $p'(\mathbf{x}')(= f(p(\mathbf{x})))$.

It is easy to show that $(f1, P1^{RLX1})$ defined in the above example satisfies these conditions.

To see the efficiency of the relaxation of Example 2, note that set $IMP_p(P1^{MDF1})$, for example, is given by

$$\begin{aligned} & IMP_p(P1^{MDF1}) = \\ & \quad \{p(x_1,x_2,y_1,y_2) \mid x_1 \le y_1, x_2 \le y_2, \\ & \quad (x_1,x_2),(y_1,y_2) \in AR2 \cup AR5 \cup AR8\}, \end{aligned}$$

which does not contain po's corresponding to the areas $AR3, AR6$ and $AR9$, even though $IMP_p(P1)$ generated from $P1$ contains them. Since the area of search is smaller, we may say that this relaxation method solves problem $(q(x_1,x_2,75,75), P1)$ more efficiently.

Example 3: It is possible to define another relaxation $P1^{RLX2}$ by extracting the information of the second coordinate from $P1$. The process is similar to Example 2. Then we can combine these two restrictions together to obtain yet another relaxation $P1^{RLX1\cdot2}$. The computation of the answer set in the modified database $P1^{MDF1\cdot2}$ from $P1^{RLX1\cdot2}$ is restricted to the following po set:

$$\begin{aligned} & PREL1\cdot2 = \\ & \quad \{p(x_1,x_2,y_1,y_2) \mid x_1 \le y_1, x_2 \le y_2, \\ & \quad (x_1,x_2),(y_1,y_2) \in AR5\} \\ & \cup \{s(x_1,x_2,y_1,y_2) \mid x_1 \le y_1, x_2 \le y_2, \\ & \quad (x_1,x_2),(y_1,y_2) \in AR5\} \\ & \cup \{q(x_1,x_2,y_1,y_2) \mid x_1 \le y_1, x_2 \le y_2, \\ & \quad (x_1,x_2),(y_1,y_2) \in AR5\}. \end{aligned}$$

It is easy to see that $PREL1\cdot2$ is much smaller than $PREL1$ or $PREL2$, and the efficiency of the relaxation method using $P1^{RLX1\cdot2}$ can be higher than that using $P1^{RLX1}$ or $P1^{RLX2}$ alone. □

For a database $P = (R, F)$, denote the number of different variables in a rule r by $rule\_deg(r)$ and $\max_{r\in R}\{rule\_deg(r)\}$ by $db\_deg(P)$ and the number of different values of constants appeared in F of P by $cnum(P)$. The processing time for P by the semi-naive method can be roughly estimated as $cnum(P)^{db\_deg(P)}$. The argument elimination strategy used in Example 2 and 3 can be considered to reduce the size of a relaxation P^{RLX} to a manageable level, by decreasing $db\_deg(P)$. In Example 2, $db\_deg(P1) = 6$, $db\_deg(P1^{RLX1}) = 3$ and $cnum(P1) = cnum(P1^{RLX1}) = 100$.

5 Relaxation Methods Based on Other Strategies

There are various strategies of defining relaxations. We have investigated so far the following strategies:

1. Argument elimination(as discussed in section 4),
2. Predicate decomposition,
3. Rule decomposition,
4. Constant grouping.

Though we omit detailed explanation, the argument elimination, the predicate decomposition and the rule decomposition construct a relaxation P^{RLX} by decreasing $db\_deg(P)$, while the constant grouping decreases $cnum(P)$. We now give an example below, in which another relaxation $P1^{RLX3}$ of database $P1$ of Example 1 is constructed by following the strategies of predicate decomposition, rule decomposition and constant grouping together.

Example 4: Consider for database $P1$ of Example 1 a mapping $f3 = (f3_t, f3_c)$ that consists of a mapping of predicates $f3_t$ and a mapping of constants $f3_c$:

$$\begin{aligned}
f3_t \ :\ & p(x_1, x_2, y_1, y_2) \rightarrow \\
& \quad p_2(x_1, y_1, y_2) \wedge p_3(x_2, y_1, y_2), \\
& s(x_1, x_2, y_1, y_2) \rightarrow \\
& \quad s_2(x_1, y_1, y_2) \wedge s_3(x_2, y_1, y_2), \\
& q(x_1, x_2, y_1, y_2) \rightarrow q(x_1, x_2, y_1, y_2), \\
& A(x_1, y_1) \rightarrow A(x_1, y_1), \\
& B(x_2, y_2) \rightarrow B(x_2, y_2), \\
& C(x_1, y_1) \rightarrow C(x_1, y_1), \\
& D(x_2, y_2) \rightarrow D(x_2, y_2), \\
f3_c \ :\ & f3_c(10(k-1)+j) = I_k, \\
& j = 1, 2, \ldots, 10, \quad k = 1, 2, \ldots, 10.
\end{aligned}$$

For example,

$$\begin{aligned}
& f3(p(x_1, x_2, y_1, y_2)) = \\
& \quad p_2(f3_c(x_1), f3_c(y_1), f3_c(y_2)) \\
& \quad \wedge\ p_3(f3_c(x_2), f3_c(y_1), f3_c(y_2)), \\
& f3(p(55, 5, 75, 65)) = \\
& \quad p_2(I_6, I_8, I_7) \wedge p_3(I_1, I_8, I_7).
\end{aligned}$$

Further, consider a relaxation $P1^{RLX3} = (R1^{RLX3}, F1^{RLX3})$ that consists of the following rule set $R1^{RLX3}$:

$$\begin{aligned}
r16 \ &:\ p_2(x_1, y_1, y_2) : -A(x_1, y_1), B(x_2, y_2). \\
r17 \ &:\ p_3(x_2, y_1, y_2) : -A(x_1, y_1), B(x_2, y_2). \\
r18 \ &:\ p_2(x_1, z_1, z_2) : -p_{21}(x_1, y_2, z_1), B(y_2, z_2). \\
r19 \ &:\ p_{21}(x_1, y_2, z_1) : -p_2(x_1, y_1, y_2), A(y_1, z_1). \\
r20 \ &:\ p_3(x_2, z_1, z_2) : -p_{31}(x_2, y_2, z_1), B(y_2, z_2). \\
r21 \ &:\ p_{31}(x_2, y_2, z_1) : -p_3(x_2, y_1, y_2), A(y_1, z_1). \\
r22 \ &\sim\ r27 : \text{rules defined for s similarly to} \\
&\quad\ r16, \ldots, r21 \\
r28 \ &:\ q(x_1, x_2, y_1, y_2) : - \\
&\quad\ p_2(x_1, y_1, y_2), p_3(x_2, y_1, y_2), \\
&\quad\ s_2(x_1, y_1, y_2), s_3(x_2, y_1, y_2).
\end{aligned}$$

and the fact set $F1^{RLX3}$:

$$\begin{aligned}
F1^{RLX3} \ =\ & \{E_i'(\mathbf{x}_i') \mid E(\mathbf{x}) \in F1, \\
& \quad E_i'(\mathbf{x}_i') \in f3(E(\mathbf{x}))\} \\
=\ & \{A(I_i, I_j) \mid j = i+1, 6 \le i < 10, \\
& \quad i : \text{integer}\} \\
\cup\ & \{A(I_i, I_i) \mid 6 \le i \le 10, i : \text{integer}\} \\
\cup\ & \{B(I_i, I_j) \mid j = i+1, 1 \le i < 10, \\
& \quad i : \text{integer}\} \\
\cup\ & \{B(I_i, I_i) \mid 1 \le i \le 10, i : \text{integer}\} \\
\cup\ & \{C(I_i, I_j) \mid j = i+1, 1 \le i < 10, \\
& \quad i : \text{integer}\} \\
\cup\ & \{C(I_i, I_i) \mid 1 \le i \le 10, i : \text{integer}\} \\
\cup\ & \{D(I_i, I_j) \mid j = i+1, 6 \le i < 10, \\
& \quad i : \text{integer}\} \\
\cup\ & \{D(I_i, I_i) \mid 6 \le i \le 10, i : \text{integer}\}.
\end{aligned}$$

Note that $P1^{RLX3}$ contains restriction on both coordinate 1 and coordinate 2 of $P1$, even though

$$db\_deg(P1^{RLX3}) = 4 < db\_deg(P1) = 6. \qquad \Box$$

The notion of constructing a simplified database P' by the strategy of predicate decomposition is also found in [9], where it is discussed when P and P' become equivalent.

6 Comparison with the Magic Set Methods

The well known magic set methods construct the restriction to be added to the original database P by introducing the predicates that contain only the arguments carrying the binding information from the constant vector $\mathbf{a}$ in a query $q(\mathbf{a}, \mathbf{x})$ [2, 4, 10, 11, 12]. The set of such rules is called the magic rule set R^{magic}, and the resulting po. set is the magic set MS. The relaxation method using the argument elimination strategy, as exemplified in section 4, is similar to the magic set methods. In this section, the relaxation method using the argument elimination strategy is compared with the magic set methods. The original magic set method was introduced in [2]. The generalized magic set method [4] and the magic templates method [10] are generalizations of the original magic set method that can treat a wider class of databases and can generate stronger restrictions. The Alexander method [11, 12] is also based on the same idea as the generalized magic set method.

First, we show that, for a general database $P = (R, F)$, the relaxation method can define a modified database P^{ADN_MDF} that generates the same po set $IMP(P^{ADN\_MDF})$ as $IMP(P^{magic})$ generated by the original magic set method. For a general database $P = (R, F)$, the original magic set method first constructs the adorned rule set R^{ADN} from R by replacing each predicate $p(x_1, \ldots, x_m)$ in rule $r \in R$ with predicate, e.g., $p^{\mathrm{bbf}\cdots\mathrm{f}}(x_1, \ldots, x_m)$, adorned with some sequence of length m of b's and f's, where b(f) means that the corresponding argument (does not) carries the binding information from the constant vector $\mathbf{a}$ in a query $q(\mathbf{a}, \mathbf{x})$. (Those rules not carrying the binding information are not included in R^{ADN}.) For example, a rule for query $q(a, x)$,

$$r : q(x, y) : -A(x, x'), B(y, y'), q(y', x'),$$

gives an adorned rule:

$$r' : q^{\mathrm{bf}}(x, y) : -A(x, x'), B(y, y'), q^{\mathrm{fb}}(y', x').$$

The original magic set method then constructs the rule set R' by replacing each rule r' of R^{ADN} by the rule r'' that is obtained from r' by eliminating all arguments with adornment f. For example, the above rule $r'(\in R^{ADN})$ becomes

$$r'' : q^{\mathrm{bf}}(x) : -A(x, x'), q^{\mathrm{fb}}(x').$$

of R'. (If a rule $r'(\in R^{ADN})$ has more than one intensional predicate in its body, r' is first decomposed into rules $r1', r2', \ldots$ so that each rj' may contain only one intensional predicate in its body, and then the arguments with f are eliminated from each rj'.) The magic rule set R^{magic} is then obtained from R' by interchanging the head predicate of each rule with the intensional predicate in its body. The magic set $MS(= IMP((R^{magic}, \{q^{\mathrm{b}}(\mathrm{a})\} \cup F)))$ is then deduced in the bottom up manner. Finally, P^{magic} is the database $P^{ADN} = (R^{ADN}, F)$ argumented with the restriction that solution space is constrained to MS (it is achieved by adding the predicates of R^{magic} to the rules of R^{ADN} in certain manner). It is known that P^{magic} is equivalent to P with respect to query $q(\mathbf{a}, \mathbf{x})$ in the sense that both have the same answer set ANS.

Now it is not difficult to observe that the rules in the above R' can be regarded as those obtained from R^{ADN} by the argument elimination strategy of Section 4, and $P^{ADN\_RLX} = (R', F)$ is a relaxation of P^{ADN}. The relaxation method, however, does not construct R^{magic} but directly evaluate P^{ADN_RLX} in the bottom up manner. Then, to obtain the relevant set $REL(q^{\mathrm{b}}(\mathrm{a}), P^{ADN\_RLX})$, it again evaluates P^{ADN_RLX} in the opposite direction (i.e., in the top down manner) from $q^{\mathrm{b}}(\mathrm{a})$ under the added restriction of $IMP(P^{ADN\_RLX})$.. Finally, P^{ADN_MDF} is obtained by adding restriction REL to P^{ADN} in the same way as adding MS to P^{ADN}. The following expression:

$$IMP(P^{magic}) = IMP(P^{ADN\_MDF}),$$

follows from the above definitions of MS and REL, even though

$$MS \supseteq REL(q^{\mathrm{b}}(\mathrm{a}), P^{ADN\_RLX}).$$

Although the complexity of computing MS and REL is generally difficult to compare, the computation time of the last phase, i.e., computation of $IMP(P^{magic})$ and $IMP(P^{ADN\_MDF})$, is much less than the computation of the original $IMP(P^{ADN})$.

To illustrate the above approach, we consider the original magic set method applied to Example 1. (In this example, adorned database $P^{ADN} = (R^{ADN}, F)$ is not introduced.) It is not difficult to show that the original magic set method generates the following set of po's:

$$\begin{aligned} & IMP(P1^{magic}) \\ &= IMP_p(P1^{magic}) \cup IMP_s(P1^{magic}) \\ &\quad \cup IMP_q(P1^{magic}) \\ &= \{p(x_1, x_2, y_1, y_2) \mid x_1 \le y_1, x_2 \le y_2, \\ &\quad (x_1, x_2), (y_1, y_2) \in AR5 + AR8\} \\ &\cup \{s(x_1, x_2, y_1, y_2) \mid x_1 \le y_1, x_2 \le y_2, \\ &\quad (x_1, x_2), (y_1, y_2) \in AR4 + AR5\} \\ &\cup \{q(x_1, x_2, y_1, y_2) \mid x_1 \le y_1, x_2 \le y_2, \\ &\quad (x_1, x_2), (y_1, y_2) \in AR5\}, \end{aligned}$$

where $P1^{magic}$ is the database obtained by adding the magic set restriction $MS1$ to $P1$. If relaxation $P1^{RLX4}$ is defined by eliminating the first and second arguments from each predicate p, s, q in $P1$, the modified database $P1^{MDF4}$ from $P1^{RLX4}$ satisfies

$$IMP(P1^{MDF4}) = IMP(P1^{magic}).$$

Next, we give a case where the relaxation method is more efficient than the original magic set method. Comparing the above set $IMP(P1^{magic})$ with $IMP(P1^{MDF1\cdot 2})$ of Example 3, where $IMP(P1^{MDF1\cdot 2}) = PREL1 \cdot 2$, we see that $IMP(P1^{MDF1\cdot 2})$ is a proper subset of

$IMP(P1^{magic})$. This is because more flexible utilization of sideways information passing [4] is possible in the framework of the relaxation method. For example, relaxation $P1^{RLX1\cdot 2}$ of Example 3 includes predicates $p1(x_1, y_1)$ and $p2(x_2, y_2)$ to restrict not only the third and fourth arguments of predicate $p(x1_1, x_2, y_1, y_2)$ in P but also its first and second arguments. The relaxation method can utilize sideways information passing through extensional and/or intensional predicates of P, while the original magic set method can utilize only sideways information passing through extensional predicates. Furthermore, in the body $p_1(x_1, y_1) \wedge s_1(x_1, y_1)$ of rule $r10$ in P^{RLX1}, the relaxation method makes use of sideways information passing not only in the direction from variable $x_1(y_1)$ of predicate $p(x_1, x_2, y_1, y_2)$ to variable $x_1(y_1)$ of predicate $s(x_1, x_2, y_1, y_2)$ but also in the opposite direction in rule $r5$ of $P1$. The relaxation method can utilize cyclic sideways information passing between predicate arguments, while the original magic set method can utilize only acyclic sideways information passing.

The generalization family of the original magic set method can also generate restriction sets that cannot be generated by the original magic set method. There are some such generalized restriction sets which cannot be realized by the relaxation method, and there are some restriction sets generated by the relaxation method which cannot be realized by the generalization family of the original magic set method.

7 Conclusions

The relaxation method has the following advantages:

1. It can utilize various restrictions flexibly and efficiently.

2. It is intuitively easy to understand how to define restrictions. Therefore, there is a chance to find useful relaxations that reflect the essence of a given database in natural way.

References

[1] F.Bancilhon,: Naive Evaluation of Recursively Defined Relations, *On Knowledge Base Management Systems - Integrating Database and AI Systems, Blodie and Mylopoulous, Eds., Springer-Verlag, Berlin, pp.165-178, 1985.*

[2] F.Bancilhon, D.Maier, Y.Sagiv and J.Ullman,: Magic sets and other strange ways to implement logic programs, *Proc. 5th ACM SIGMOD-SIGACT Symp. on Principles of Database System(PODS), 1986*

[3] F.Bancilhon and R.Ramakrishnan, : An amateur's introduction to recursive query processing strategies, *Proc. ACM-SIGMOD Conf. on Management of Data(SIGMOD), Washington,D.C., 1986.*

[4] C.Beeri and R.Ramakrishnan,: On the power of magic, *Proc. 6th ACM SIGACT-SIGMOD-SIGART Symp. on Principles of Database Systems(PODS), San Diego, Calif., 1987.*

[5] A.V.Gelder,: A Message Passing Framework for Recursive Query Evaluation, *Proc. SIGMOD, 1986*

[6] L.Henschen and S.Naqvi,: On Compiling Queries in Recursive First-Order Data Bases, *JACM 31, pp.47-85, 1984.*

[7] E.Lozinskii,: Evaluating Queries in Deductive Databases by Generating, *Proc. 11th Int. Joint Conf. on Artificial Intelligence, 1985.*

[8] D.Mckay and S.Shapiro,: Using Active Connection Graphs for Reasoning with Recursive Rules, *Proc. 7th Int. Joint Conf. on Artificial Intelligence, 1981*

[9] J.F.Naughton, R.Ramakrishnan, Y.Sagiv and J.D.Ullman,: Argument Reduction by Factoring, *Proc. VLDB, pp.173-182, 1989.*

[10] R.Ramakrishnan,: Magic templates: A spellbinding approach to logic programs, *Proc. 5th Int. Conf. and Symp. on Logic Programming(ICLP/SLP), Seattle, Wash., 1988.*

[11] J.Rohmer, R.Lescoeur and J.M.Kerisit,: The Alexander method, a technique for the processing of recursive axioms in deductive database, *New Generation Computing 4(3), 1986.*

[12] H.Seki,: On the power of Alexander templates, *Proc. 8th ACM SIGACT-SIGMOD-SIGART Symp. on Principles of Database Systems(PODS), Philadelphia, Penn., 1989.*

[13] J.Ullman,: Implementation of Logical Query Languages for Databases, *TOLD, 10(3), pp.289-321, 1985.*

[14] L.Vieille,: Recursive axioms in Deductive Databases. The Query/Subquery Approach, *Proc. First Int. Conf. on Expert Database Systems, Charleston, 1986*

(受理 平成 4 年 3 月 20 日)