数理解析研究所講究録
第 363 巻 1979 年 165-187

RELATIONAL STRATEGIES FOR PROCESSING UNIVERSALLY QUANTIFIED QUERIES TO LARGE DATA BASES

Koichi Furukawa
Computer Science Division, Electrotechnical Laboratory
Tokyo, Japan

ABSTRACT

A new algorithm is developed to process universally quantified queries under the closed world assumption. Our algorithm evaluates a universally quantified conditional query very efficiently by reducing the problem to the application of the operations of set intersection, summary and join to the answers to subqueries. Furthermore, the algorithm is extended to handle numerical quantifiers.

The sum-of-product decomposition of a relation is introduced as an efficient representation schema to express a set of uniform intensional data of the form $(\forall\vec{x}/\vec{\tau})P(\vec{x})$. Queries to the original relation are transformed to those to the decomposed relations using a query transformation axiom. Furthermore, universally quantified queries to those relations are converted to quantifier-free queries by the symbolic division.

Keywords: deductive question answering, relational data base, relational algebra, closed world assumption, query transformation, query evaluation, universal quantifier, numerical quantifier.

1. INTRODUCTION

It has been shown in Chang [1978], Furukawa [1977], Minker [1978], Kellog et. al. [1978] and Reiter [1978b] that deductive logic offers considerable potential for improving on-line access to large, complex data base domains. The common feature of the current researches to that direction is the attempt to combine a deductive component with relational data bases.

Chang [1978] divided the approaches of these researches into two groups: One is the evaluational approach where intensions are used to transform queries and extensions are used to evaluate queries. The other is the non-evaluational one where both intensions and extensions are used to prove a question represented by a formula in the same manner. It has been shown in Reiter [1978b] that the evaluational approach is more feasible for data bases with very large extensions and comparatively small intensions.

Codd [1972] has defined an algorithm to convert queries expressed in relational calculus to a sequence of relational algebraic operations. In his algorithm, universally quantified queries are converted to the application of the division operation to the answers to the subqueries. Parelmo [1974] has improved Codd's algorithm by planning the evaluation process locally to keep intermediate results as small as possible.

Reiter [1978b] has developed a general framework in the restricted first order logic which enables one to get indefinite answers to any queries, and reformulated Codd's algorithm in his framework. Reiter [1978a] has also developed an efficient query conversion algorithm under the closed world assumption (CWA).

In this paper, a new algorithm to evaluate universally quantified conditional queries under the CWA is presented. Summary operation on a relation is added to the set of primitive operations of the relational algebla. The algorithm reduces the problem to the application of the operations of set intersection, summary and join to the answers to the subqueries. The above strategy is considered to perform a global planning and optimization.

Furthermore, the algorithm is extended to handle numerical quantifiers such as "at least two", "exactly three" or "more than a half".

In deductive relational data bases developed so far, only extensions have been considered to compose relations. Intensions have been stored in knowledge bases outside the relational data bases. However, it sometimes happens that we have a set of uniform intensional data of the form $(\forall\vec{x}/\vec{\exists})P(\vec{x})$. In this paper, a new representation schema based on the sum-of-product decomposition of a relation is introduced as a suitable way to express such data. It will be shown that the above representation schema increases the power of expression of relational data bases nearly as high as that of hierarchical data bases.

Queries to the decomposed relations are evaluated using a set of axioms which give users the image of the original relations as a logical view. Furthermore, it will be shown that universally quantified queries on those relations can be evaluated very efficiently by a symbolic division.

In this paper, proofs of the theories are omitted because of the limit of the space.

2. REVIEW OF QUERY EVALUATION UNDER THE CWA

In this section, query evaluation under the CWA developed by Reiter [1978a] will be reviewed quickly as well as some extentions.

All queries have the form $[x_1/\tau_1,\ldots,x_n/\tau_n:(q_1y_1/\theta_1)\ldots(q_my_m/\theta_m)W(x_1,\ldots,x_n,y_1,\ldots y_m)]$ where $W(x_1,\ldots,x_n,y_1,\ldots,y_m)$ is a quantifier free formula with free variables $x_1,\ldots,x_n,y_1,\ldots,y_m$ and q_i is either $\forall$ or $\exists$, $i = 1,\ldots,m$. We shall use the abbreviated notation $[\vec{x}/\vec{\tau} : (q\vec{y}/\vec{\theta})W(\vec{x},\vec{y})]$ to express a typical query. The τ's and θ's, which are called types, are sets of constant signs which the variables associated with them range over. A sequence of types $\vec{\tau} = (\tau_1,\ldots,\tau_n)$ is the set $\tau_1\times\ldots\times\tau_n$.

A data base (DB) is a set of clauses containing no functional signs.

Under the CWA, if no proof of a positive ground literal exists, then the negation of that literal is assumed true. This can be viewed as equivalent to implicitly augmenting the given data base with all such negated literals.

Since worlds are completely specified under the CWA, the set of CWA answers $!Q!_{CWA}$ to $Q = [\vec{x}/\vec{\tau} : (Q\vec{y}/\vec{\theta})W(\vec{x},\vec{y})]$ is defined as follows:

$$!Q!_{CWA} = \{\vec{c} : \vec{c} \in \vec{\tau} \text{ and } DB \cup \overline{EDB} \vdash (q\vec{y}/\vec{\theta})W(\vec{x},\vec{y})\}$$

where $\overline{EDB} = \{\overline{P}c : P$ is a predicate sign, $\vec{c}$ a tuple of constant signs and $DB \nvdash P\vec{c}\}$.

It seems to be impractical to get the set of CWA answers to Q since $\overline{EDB}$ may contain infinite numbers of literals. But it has been shown in Reiter [1978a] that the CWA answers to an atomic query can be obtained without $\overline{EDB}$, i.e. $!Q!_{CWA}=!Q!_{OWA}$, if $DB \cup \overline{EDB}$ is consistent ($!Q!_{OWA}$ is a set of minimal answers to Q under the open world assumption). Henceforth, we consider only the CWA answers and abbreviate $!Q!_{CWA}$ to $!Q!$.

The evaluation of any queries are reduced to the applications of the relational algebraic operations to the answers to atomic queries. Before listing query conversion rules, we define some of the operations of relational algebra.

Let $Q = [\vec{x}/\vec{\tau}, z/\psi : W]$, and $\vec{x}$ is the n-tuple $x_1,\ldots,x_n$. Then $!Q!$ is a set of (n+1)-tuples, and the <u>projection of</u> $!Q!$ <u>with</u>

respect to z, π_z!Q! is the set of n-tuples obtained from !Q! by deleting the (n+1)st component from each (n+1)-tuple of !Q!.

Let Q = $[\vec{x}/\vec{\tau}, z/\psi : W]$. Then the quotient of !Q! by z, Δ_z!Q!, is a set of tuples and is defined as follows:

$\vec{c} \in \Delta_z$!Q! iff $(\vec{c}, a) \in$!Q! for all $a \in \psi$.

The operater Δ_z is called the division with respect to z.

Let Q1 = $[\vec{x}/\vec{\tau}, z/\psi : W1]$ and Q2 = $[\vec{y}/\vec{\theta}, z/\psi : W2]$. The join of !Q1! and !Q2! with respect to z, *z(!Q1!,!Q2!), is a set of tuples and defined as follows:

$(\vec{c}, d, \vec{e}) \in$ *z(!Q1!,!Q2!)

iff $(\vec{c}, d) \in$!Q1! and $(\vec{e}, d) \in$!Q2!.

We sometimes denote *z(!Q1!,!Q2!) as (!Q1! *z !Q2!).

The set of query conversion rules are listed below:

Rule 1. (Decomposition of AND/OR queries)

1. $![\vec{x}/\vec{\tau} : W1 \wedge W2]! = ![\vec{x}/\vec{\tau} : W1]! \cap ![\vec{x}/\vec{\tau} : W2]!$.
2. $![\vec{x}/\vec{\tau} : W1 \vee W2]! = ![\vec{x}/\vec{\tau} : W1]! \cup ![\vec{x}/\vec{\tau} : W2]!$.

Rule 2. (Elimination of negation)

1. $![\vec{x}/\vec{\tau} : \overline{W}]! = \vec{\tau} - ![\vec{x}/\vec{\tau} : W]!$.
2. $![\vec{x}/\vec{\tau} : W1 \wedge \overline{W2}]! = ![\vec{x}/\vec{\tau} : W1]! - ![\vec{x}/\vec{\tau} : W2]!$.

Rule 3. (Distribution of quantifiers)

1. $[\vec{x}/\vec{\tau} : (\exists\vec{y}/\vec{\theta})(W1 \vee W2)]$

   $= [\vec{x}/\vec{\tau} : (\exists\vec{y}/\vec{\theta})W1] \vee [\vec{x}/\vec{\tau} : (\exists\vec{y}/\vec{\theta})W2]$.
2. $[\vec{x}/\vec{\tau} : (\forall\vec{y}/\vec{\theta})(W1 \wedge W2)]$

$= [\vec{x}/\vec{\tau} : (\forall \vec{y}/\vec{\theta})W1] \wedge [\vec{x}/\vec{\tau} : (\forall \vec{y}/\vec{\theta})W2].$

Rule 4. (Elimination of exsistential quantifiers)

$!\,[\vec{x}/\vec{\tau} : (\exists \vec{y}/\vec{\theta})W]! = \pi_{\vec{y}}!\,[\vec{x}/\vec{\tau}, \vec{y}/\vec{\theta} : W]!$

where $\pi_{\vec{y}}$ denotes $\pi_{y1}\ldots\pi_{ym}$ and π_{yi} is the projection with respect to yi.

Rule 5. (Elimination of universal quantifiers)

$!\,[\vec{x}/\vec{\tau} : (\forall \vec{y}/\vec{\theta})W]! = \Delta_{\vec{y}}!\,[\vec{x}/\vec{\tau}, \vec{y}/\vec{\theta} : W]!$

where $\Delta_{\vec{y}}$ denotes $\Delta_{y1}\ldots\Delta_{ym}$ and Δ_{yi} is the division with respect to yi.

Rule 6. (Decomposition of existentially quantified AND queries)

$!\,[\vec{x}/\vec{\tau} : (\exists \vec{y}/\vec{\theta})(W1 \wedge W2)]!$

$= (!\,[\vec{x}1/\vec{\tau}1, \vec{y}/\vec{\theta} : W1]! * !\,[\vec{x}2/\vec{\tau}2, \vec{y}/\vec{\theta} : W2]!)$

where $\vec{x}i$ is a sub-sequence of $\vec{x}$ which appears in Wi, and $\vec{\tau}i$ is the corresponding sub-sequence of $\vec{\tau}$, for i = 1, 2. The join operation * is performed with respect to the variables $\vec{y}$ and $\vec{x}1 \cap \vec{x}2$.

Most of the above rules are derived from Reiter [1978a]. The extended points are as follows:

1. In rule 2, the restriction that W, W1 and W2 must be quantifier free is removed. For example, if $W = (\exists \vec{y}/\vec{\theta})W'$, then

$!\,[\vec{x}/\vec{\tau} : \overline{W}]! = !\,[\vec{x}/\vec{\tau} : \overline{(\exists \vec{y}/\vec{\theta})W'}]! = !\,[\vec{x}/\vec{\tau} : (\forall \vec{y}/\vec{\theta})\overline{W'}]!.$

Rule 2 claims that

$!\,[\vec{x}/\vec{\tau} : (\forall \vec{y}/\vec{\theta})\overline{W'}]! = \vec{\tau} - !\,[\vec{x}/\vec{\tau} : (\exists \vec{y}/\vec{\theta})W']!$

but not that

$$![\vec{x}/\vec{\tau} : (\exists\vec{y}/\vec{\theta})\overline{W'}]! = \vec{\tau} - ![\vec{x}/\vec{\tau} : (\exists\vec{y}/\vec{\theta})W']!,$$

which has been presented as a counter example in Reiter [1978a].

2. Rule 3.2 and Rule 5 are new. Reiter has developed a general framework which enables one to get indefinite answers. The division operator defined by Codd [1972] has been properly generalized in the framework. However, the handling of universal quantifiers in the CWA has been ommitted in Reiter [1978a]. But there is no reason to exclude the handling of universal quantifiers in the CWA and moreover the set of conversion rules becomes more well-structured by adding these rules.

3. Rule 6 is new, too. It is easily shown that this conversion reduces the amount of computation. Let Qi = $[\vec{x}/\vec{\tau},\vec{y}/\vec{\theta} : Wi]$ and $\vec{\tau}_i^c = \pi_{\bar{x}_i}(\vec{\tau})$, i.e., the compliment subsequence of $\vec{\tau}_i$, i = 1, 2. We would have to evaluate Q = $[\vec{x}/\vec{\tau} : (\exists\vec{y}/\vec{\theta})(W1 \wedge W2)]$ according to $!Q! = \pi_{\vec{y}}(!Q1! \cap !Q2!)$, unless we have Rule 6. But it is easily shown that

$$!Qi! = \vec{\tau}_i^c \times ![\vec{x}i/\vec{\tau}i, \vec{y}/\vec{\theta} : Wi]!$$

for i = 1, 2. By applying Rule 6, we can avoid the execution of the above costly Cartesian product operations.

3. THE HANDLING OF UNIVERSAL QUANTIFIERS

Let us consider how to compute the division $\Delta_z!Q!$ where Q = $[\vec{x}/\vec{\tau},z/\psi : (q\vec{y}/\vec{\theta})W]$. The following algorithm generates the answer Δ_z!Q!.

Algorithm D.

1. Group the answer set !Q! to the quey Q by $\vec{x}$ and make a list of z's for each $\vec{x}$ (we denote the list by $Z\vec{x}$).
2. For each $Z\vec{x}$, test whether it includes the set ψ. If the test succeeds, then put the $\vec{x}$ in the answer set Δ_z!Q!.

Rule 5 says that Δ_z!Q! is the answer set for the universally quantified query $Q0 = [\vec{x}/\vec{\tau} : (\forall z/\psi)(q\vec{y}/\vec{\theta})W]$. Notice that Q0 is the abbreviation of the conditional query $Q0' = [\vec{x}/\vec{\tau} : (\forall z)(z \in \psi \rightarrow (q\vec{y}/\vec{\theta})W)]$. The inclusion test $\psi \subset Z\vec{x}$ in Algorithm D reflects the implication in Q0', where the consequent $z \in \psi$ in Q0' corresponds to the lefthand set ψ and the antecedent $(q\vec{y}/\vec{\theta})W$ to the righthand set $Z\vec{x}$.

We can use the division operation to efficiently evaluate a class of conditional queries $Qc = [\vec{x}/\vec{\tau} : (\forall z/\psi)(W1 \rightarrow W2)]$ such that W1 does not contain any free variables in $\vec{x}$. The following theorem states the fact more precisely.

Theorem 1. Let $Qc = [\vec{x}/\vec{\tau} : (\forall z/\psi)(W1 \rightarrow W2)]$ be a universally quantified conditional query and assume that W1 does not contain any free varibles in $\vec{x}$, $Q = [\vec{x}/\vec{\tau}, z/\psi : W2]$, and $Qz = [z/\psi : W1]$. If $DB \cup \overline{EDB}$ is consistent, then

$$!Qc! = \Delta_{z \in !Q_z!} !Q! \text{ if } !Qz! \neq \phi,$$
$$= \vec{\tau} \quad \text{otherwise.}$$

We cannot use Theorem 3 if W1 contains any variables in

$\vec{x}$. An example of such queries is: "List the departments that sell all items that are supplied"; that is,

[x/DEPT : ($\forall$y/ITEM) SUPPLIED(y,x) $\rightarrow$ SELL(x,y)].

We can deal with the general case by introducing the following cut operation:

Let R = ![$\vec{x}/\vec{\tau}$,z/ψ : W($\vec{x}$, z)]!. Then the <u>cut of</u> R <u>with respect to</u> $\vec{x}$, denoted by $\rho_{\vec{x}}$(R), is a set of pairs (c_i, D_i) where $\vec{c}_i \in \vec{\tau}$ and D_i is a set of $d_i \in \psi$ such that

$D_i = \{d_k : (\vec{c}_i, d_k) \in R\}$.

We regard $\rho_{\vec{x}}$(R) to be a function from $\pi_{\vec{z}}$R to $2^{\pi_{\vec{z}}R}$ and denote D_i as $\rho_{\vec{z}}(R)(\vec{c}_i)$. $\rho_{\vec{z}}(R)(\vec{c}_i)$ is called the <u>cross-section of</u> $\rho_{\vec{z}}$(R) <u>at</u> $\vec{c}_i$.

The set of D_i is exactly the same as $Z\vec{c}_i$ in Algorithm D. The cut operation corresponds to the group-by operation in SEQUEL2 (Chamberlin et. al. [1976]).

<u>Theorem 2</u>. Let Q = [$\vec{x}/\vec{\tau}$: ($\forall$z/ψ)(W1 $\rightarrow$ W2)]. If DB $\cup$ $\overline{\text{EDB}}$ is consistent, then

$$!Q! = (\vec{\tau} - ![\vec{x}/\vec{\tau} : (\exists z/\psi)W1]!) \cup \{\vec{c} : \phi \neq \rho_{\vec{z}}(R1)(\vec{c}) \subset \rho_{\vec{z}}(R2)(\vec{c})\},$$

where Ri = ![$\vec{x}/\vec{\tau}$,z/ψ: Wi]! for i = 1, 2.

<u>Corollary 2.1</u>. Let Q = [$\vec{x}/\vec{\tau}$: ($\forall$z/ψ)(W1 $\rightarrow$ W2)] where W1 does not contain any free variables in $\vec{x}$, R1 = ![z/ψ: W1]! and R2 = ![$\vec{x}/\vec{\tau}$,z/ψ: W2]!. If DB $\cup$ $\overline{\text{EDB}}$ is consistent, then

$$!Q! = \{\vec{c} : R1 \subset \rho_{\vec{z}}(R2)(\vec{c})\} \quad \text{if } R1 \neq \phi,$$
$$= \vec{\tau} \quad \text{otherwise.}$$

Since $A \subset B$ iff $|A \cap B| = |A|$ in case A and B are finite sets, we can replace the set inclusion test in Theorem 2 by a simple comparison of the cardinal numbers of two sets. This strategy has been implemented in GM-RDMS (Whitney [1974]), where an operation called summary has been used to construct a set of pairs $(\vec{c}_i, |D_i|)$ which summarize the pairs $(\vec{c}_i, D_i)$.

Example

The following example is taken from Palermo [1974]. There are four relations: SUPPLY, SUPPLIERS, PROJECT and PART, as shown in Figure 1 (The relation names are underlined to distinguish them from the corresponding predicates. They are considered to refer sets, rather than predicates.). The relation SUPPLY (SID, PID, JID) is a set of tuples (x, y, z) such that a supplier x supplies a part y to a project z. SUPPLIER (SID, SLOC, SNAME) has information on suppliers' locations and their names. PROJECT (JID, JLOC, JNAME) has information on projects' locations and their names. PART (PID, PTYPE) defines the type of each part.

Let us consider the following query to the above data base:

Query. Find the name and location of suppliers each of whom has supplied at least one project located in San Jose with at least one of every part of type A.

This query is expressed as follows:

Q = [x/SNAME, y/SLOC :

$$(\exists s/SID)(SUPPLIER(s,y,x) \wedge (\exists j/PID)(PROJECT(SJ,j) \wedge (\forall p/PID)(PART(p,A) \rightarrow SUPPLY(s,p,j)))))]$$

The evaluation of Q proceeds as follows:

1. Let Q1 = [x/SNAME,y/SLOC,s/SID : SUPPLIER(s,y,x)] and Q2 = [s/SID : ($\exists$j/JID)(PROJECT(SJ,j) $\wedge$ ($\forall$p/PID)(PART(p,A) SUPPLIER(s,p,j))))]. By applying RUle 6 to Q, we get

   $$!Q! = \pi_s(!Q1! \ast_s !Q2!).$$

2. Let Q21 = [j/JID : PROJECT(SJ,j)] and Q22 = [s/SID,j/JID : ($\forall$p/PID)(PART(p,A) $\rightarrow$ SUPPLY(s,p,j))]. From Rule 6, we obtain

   $$!Q2! = \pi_j(!Q21! \ast_j !Q22!).$$

3. Let R1 = ![p/PID : PART(p,A)]! and

   R2 = ![s/SID,j/JID,p/PID : SUPPLY(s,p,j)]!

   Since R1 is not an empty set,

   $$!Q22! = \{(s1,j1) : R1 \subset \rho_{(s,j)}(R2)(s1,j1)\}$$

   from Corollary 2.1.

The snapshots of the above evaluation process is shown in Figure 2, where the summary operation is used to evaluate Q22.

Numerical quantifiers such as "at least two", "exactly three" or "more than a half" can be neatly dealt with by using the summary operation. We adopt the Chang's notation for numerical quantifiers (Chang [1978]): Let op be one of the relative operators $<, \leq, >, \geq$ or $=$. The fact (x op n) is expressed as ($\exists$ op nx). For example, $(x > 2)$ is expressed as ($\exists > 2x$). The following theorems give the algorithms to evaluate numerically quantified queries.

Theorem 3. Let Q = $[\vec{x}/\vec{t} : (\exists \text{ op } nz/\psi)W]$ and R = $![\vec{x}/\vec{t}, z/\psi : W]!$. If DB $\cup$ $\overline{\text{EDB}}$ is consistent, then

$$!Q! = \{\vec{c} : |\psi \cap \rho_{\vec{x}}(R)(\vec{c})| \text{ op } n\}.$$

Theorem 4. Let Q = $[\vec{x}/\vec{t} : (\exists \text{ op } nz/\psi)(W1 \rightarrow W2)]$. If DB $\cup$ $\overline{\text{EDB}}$ is consistent, then

$$!Q! = \{\vec{c} : |\rho_{\vec{x}}(R1)(\vec{c}) \cap \rho_{\vec{x}}(R2)(\vec{c})| \text{ op } n\}$$

where Ri = $![\vec{x}/\vec{t}, z/\psi : Wi]!$ for i = 1, 2.

In order to deal with proportional quantifiers such as "a half" or "80%", we need to slightly expand the Chang's notation. We denote (x op (n/100)•#div) as ($\exists$ op n%x) where #div is the cardinal number of the divisor ψ in Theorem 3 or the cross-section $\rho_{\vec{x}}(R1)(\vec{c})$ in Theorem 4. Note that the universal quantifier is equal to the extream case of the proportional quantifiers, that is, "100%" or ($\exists$ = 100%x).

4. THE SUM-OF-PRODUCT DECOMPOSITION OF A RELATION

Suppose that the answer of the universally quantified query Q22 in the previous example is saved. Then, we can answer to the same query immediately by simply restoring the saved answer. We need to save answers for other types, e.g. a set of supplier-project pairs (x, y) such that x supplies all parts of type B to y. These answers can be put together if we associate each type to each answers. Let SUPPLYALL (SID, JID, PTYPE) be the relation built in such a way. From SUPPLYALL and PART relations, we can infer the "SUPPLY" fact, namely:

$$(\forall s/SID)(\forall j/JID)(\forall p/PID)$$
$$((\exists t/PTYPE)SUPPLYALL(s,t,j) \wedge PART(p,t)$$
$$\rightarrow SUPPLY(s,p,j)) \qquad (4.1)$$

Furthermore, we can remove all tuples which are infered in (4.1) from the original SUPPLY relation. We rename the reduced SUPPLY relation as SUPPLYSOME. The original SUPPLY relation is now decomposed into two relations: SUPPLYALL and SUPPLYSOME, as shown in Figure 3.

Now, let us consider the query Q22 in the previous example:

$$Q22 = [s/SID,j/JID:(\forall p/PID)(PART(p,A)$$
$$\rightarrow SUPPLY(s,p,j))]. \qquad (4.2)$$

The extension of SUPPLY(s,p,j) can be obtained from not only the SUPPLYSOME relation, but also the SUPPLYALL relation. Assume that there are no other relations which contain the information about SUPPLY(s,p,j). Then, the following equation holds:

$$
\begin{aligned}
&(\forall s/SID)(\forall p/PID)(\forall j/JID) \\
&\quad (SUPPLY(s,p,j) \equiv \\
&\qquad (\exists t/PTYPE)(SUPPLYALL(s,t,j) \wedge PART(p,t)) \\
&\qquad \vee SUPPLYSOME(s,p,j)). \qquad (4.3)
\end{aligned}
$$

This statement is a kind of query transformation axioms which give simple logical views of relations to the users (Furukawa [1977]). It is also considered to be a kind of definitional clauses (Reiter [1977]). The query Q22 is transformed to:

$$
\begin{aligned}
Q22 = [s/SID,j/JID :\ & \\
&(\forall p/PID)(PART(p,A) \\
&\quad \rightarrow (\exists t/PTYPE)(SUPPLYALL(s,t,j) \\
&\qquad\qquad \wedge PART(p,t)) \\
&\quad \vee SUPPLYSOME(s,p,j))] \qquad (4.4)
\end{aligned}
$$

by substituting the righthand expression of (4.3) for the consequent of (4.2). Considering the meaning of the predicate "SUPPLYALL", it is expected that !Q221! = ![s/SID,j/JID : SUPPLYALL(s,A,j)]! should be included in the answer set !Q22!. In fact, we can deduce !Q221! as a part of the answer (actually, !Q221! is the answer set itself. See Theorem 5). Since the antecedent of Q22 does not include any free variables, we can obtain the answer set !Q22! by division (Theorem 1). By substituting A for t in the first term of the antecedent of Q22, we get

SUPPLYALL(s,A,j) $\wedge$ PART(p,A). (4.5)

Since SUPPLYALL(s,A,j) and PART(p,A) does not share any variables, the extension R_{spj} of the expression (4.5) is equal to the direct product

!\[s/SID,j/JID : SUPPLYALL(s,A,j)]!
$\times$![p/PID : PART(p,A)]!.

The set R_{spj} is a part of the divident of the division. We cirtainly get !Q221! = ![s/SID,j/JID : SUPPLYALL(s,A,j)]! as a part of the quotient by symbolically dividing the set R_{spj} by the divisor ![p/PID : PART(p,A)]!.

We will give a more general description about what we have explained above.

Let REL ($\vec{X}$, Z) be an arbitrary relation and DIV (Z, T) a relation which defines the type t of each element z in $\pi_{\vec{X}}$ (REL). Suppose that there are n types T = {a1,...,an} of z. We select all z such that (z, ai) $\in$ DIV and make a unary relation DIVai (Z) for each ai in T. Let RELai ($\vec{X}$, Z) be a sub-relation of REL which consists of all and only elements ($\vec{c}$, d) such that d $\in$ DIVai. Then, it is obvious that

$$\underline{REL} = \bigcup_{ai \in T} \underline{RELai} \qquad (4.6)$$

Let QUOai ($\vec{X}$) and REMai ($\vec{X}$, Z) be the quotient and the remainder, respectively, which result from dividing RELai by DIVai; namely,

$$\underline{RELai} = \underline{QUOai} \times \underline{DIVai} \cup \underline{REMai}, \quad ai \in T. \qquad (4.7)$$

From (4.6) and (4.7), we get

$$\underline{REL} = \bigcup_{ai \in T} (\underline{QUOai} \times \underline{DIVai}) \cup \underline{REM}, \qquad (4.8)$$

where $\underline{REM} = \bigcup_{ai \in T} \underline{REMai}$.

The relation QUOai corresponds to the answer set in the previous example. In order to keep all answers QUOai, ai ϵ T, in a single relation, we associate the constant type information ai to each tuple of QUOai and make a new relation QUOai' ($\vec{X}$, T) for each ai ϵ T. The unified relation QUO ($\vec{X}$, T) is a union of QUOai' for all ai ϵ T. Since

$$\pi_t(\underline{QUOai'} *_t \underline{DIV}) = \underline{QUOai} \times \underline{DIVai} \qquad (4.9)$$

holds for all ai ϵ T, we get

$$\pi_t(\underline{QUO} *t \ \underline{DIV}) = \bigcup_{ai \in T} (\underline{QUOai} \times \underline{DIVai}). \qquad (4.10)$$

From (4.8) and (4.9), we conclude

$$\underline{REL} = \pi_t(\underline{QUO} *t \ \underline{DIV}) \cup \underline{REM}. \qquad (4.11)$$

In (4.11), the relation REL is decomposed into three relations; QUO, DIV, and REM. We shall refer to such a decomposition as a sum-of-product decomposition.

The corresponding logical statement to the equation (4.11) is

$$(\forall \vec{x}/\vec{\tau})(\forall z/\psi)(REL(\vec{x},z) \equiv (\exists t/T)(QUO(\vec{x},t) \wedge DIV(z,t)) \vee REM(\vec{x},z)). \qquad (4.12)$$

Some of the universally quantified queries to the decomposed relations are reduced to simpler quantifier-free queries by applying the following theorem:

Theorem 5. Suppose 1 REL $(\vec{X}, Z)$ is decompesed to QUO $(\vec{X}, T)$, DIV (Z, T) and REM $(\vec{X}, Z)$ as (4.12), and $Q = [\vec{x}/\vec{\tau} : (\forall z/\psi)(DIVai(z) \rightarrow REL(\vec{x},z))]$. If DB ∪ $\overline{EDB}$ is consistent, then

$$!Q! = ![\vec{x}/\vec{\tau} : QUO(\vec{x},ai)]!.$$

The relation QUO, unlike ordinary relations, is a set of intensional data. It is easily shown from Theorem 5 that $(\vec{c},ai) \in QUO$ iff $(\forall z/Z)(DIV(z,ai) \rightarrow REL(\vec{c},z))$; that is, any tuples in QUO represent corresponding intensional fact. For example, a tuple (s0,t0,j0) in SUPPLYALL means that the supplier s0 supplies all parts of type t0 to the project j0. On the other hand, "QUO($\vec{c}$,ai)" may sometimes be interpreted as an extensional fact by itself. (s0,t0,j0) becomes an extensional fact if it is not unusual that some suppliers supply to some projects all parts of some types. In that case, the query will be expressed directly in terms of "SUPPLYALL", instead of through "SUPPLY".

It has been suggested in Ohsuga [1979] that the information clustering by the division is an important mechanism to build a structure in the data base. The sum-of-product decomposition schema, together with the associated query transformation axiom, enables one to realize the above mechanism in the framework of deductive relational data bases.

We finally remark the evaluation of queries other than universally quantified ones to the decomposed relations. For example, consider the query Q = [p/PID,j/JID : SUPPLY(237,p,j)]. By applying the query transformation axiom (4.3) to Q, it is transformed to

$$Q = [p/PID,j/JID : (\exists t/PTYPE)(SUPPLYALL(237,t,j) \wedge PART(p,t)) \vee SUPPLYSOME(237,p,j)].$$

!Q! is converted to

$$!Q! = ![p/PID,j/JID : (\exists t/PTYPE)(SUPPLYALL(237,t,j) \wedge PART(p,t)]! \cup ![p/PID,j/JID : SUPPLYSOME(237,p,j)]! \quad (4.13)$$

by using the set of conversion rules in section 2.

We need to perform the costly join of the SUPPLYALL relation and the PART relation to evaluate the first term in (4.13). However, it may sometimes be adequate to answer in terms of SUPPLYALL and SUPPLYSOME instead of SUPPLY. In that case, we can avoid the join. It is related to the notion of approximate responses discussed in Joshi et. al. [1977].

5. CONCLUSION

Practical algorithms to evaluate universally quantified queries as well as numerically quantified ones are presented. They are formulated in the same way so that they can be realized compactly.

Furthermore, a new representation schema based on the sum-of-product decomposition of relations are introduced. A set of uniform intensional data of the form $(\forall\vec{x}/\vec{z})P(\vec{x})$ is expressed as a relation by using the schema. Universally quantified queries to the decomposed relations are converted to quantifier-free queries by the symbolic division.

The DBAP (Furukawa [1977]) are being expanded to realize the algorithms presented in this paper.

This research is considered to be a step toward a natural language QA system. It has been indicated in Furukawa [1977], Sacedoti [1977] and Harris [1977] that the most difficult problem to be solved to realize such a system is to mesh the user's conceptualization of data with the actual structure of the data. The query transformation mechanism developed here as well as in Furukawa [1977] will help to solve the problem.

ACKNOWLEDGEMENT

The auther thanks to Dr. Osamu Ishii and Dr. Akio Tojo for providing a chance of the present study, and to Professor Setsuo Ohsuga, Mr. Kazuhiro Fuchi and the staffs of the information systems section for their helpful discussions.

REFERENCES

1. Chamberlin, D. D. et. al. [1976] "SEQUEL 2: A Unified Approach to Data Definition, Manipulation and Control," IBM J. Res. Develop. Vol. 20, No. 6, 1976, 560-575.
2. Chang, C. L. [1978] "DEDUCE 2: Further Investigations of Deduction in Relational Data Bases," in Logic and Data

Bases, (ed. H. Gallaire and J. Minker), Plenum Press, 1978, 201-236.

3. Codd, E. F. [1972] "Relational Completeness of Data Base Sublanguages," in Data Base Systems, (Ed. R. Rustin), Prentice-Hall, 1972, 65-98.

4. Furukawa, K. [1977] "A Deductive Question Answering System on Relational Data Bases," Proc. Fifth IJCAI, 1977, 59-66.

5. Harris, L. R. [1977] "User Oriented Data Base Query with the ROBOT Natural Language Query System," Proc. Third VLDB, 1977, 303-311.

6. Joshi, A. K. et. al. [1977] "Approximate Responses from a Data Base Query System: An Application of Inferencing in Natural Language," Proc. Fifth IJCAI, 1977, 211-212.

7. Kellog, C. et. al. [1978] "Deductive Planning and Pathfinding for Relational Data Bases," in Logic and Data Bases, 1978, 179-200.

8. Minker, J. [1978] "An Experimental Relational Data Base Systems on Logic," in Logic and Data Bases, 1978, 107-148.

9. Ohsuga, S. [1979] "Toward Intelligent Interactive Systems," Proc. The IFIP W.G. 5.2 Workshop Seillac II on Methodology of Interaction, (Ed. P. Ten Hagen), North-Holland, 1979.

10. Palermo, F. P. [1974] "A Data Base Search Problem," in Information Systems, (Ed. J. T. Tou), Plenum Press, 1974.

11. Reiter, R. [1977] "An Approach to Deductive Question-Answering," BBN Report No. 3649, Bolt, Beranek and Newman, Inc., 1977.

12. Reiter, R. [1978a] "Deductive Question-Answering on Relational Data Bases," in Logic and Data Bases, 1978, 149-178.

13. Reiter, R. [1978b] "On Closed World Data Bases," in Logic and Data Bases, 1978, 55-76.

14. Sacerdoti, E. D. [1977] "Language Access to Distributed Data with Error Recovery," Proc. Fifth IJCAI, 1977, 196-202.

15. Whitney, V. K. [1974] "A Relational Data Management System," in Information Systems, (Ed. J. T. Tou), Plenum Press, 1974.

SUPPLY

SID	PID	JID
211	31	971
325	32	971
211	33	970
211	31	972
237	31	972
237	31	970
237	32	970
237	33	970
237	32	971
237	31	971

SUPPLIER

SID	SLOC	SNAME
211	NY	AA
325	SF	XX
237	LA	YY

PROJECT

JID	JLOC	JNAME
970	POK	A
971	SJ	X
972	SJ	Y

PART

PID	PTYPE
31	A
32	A
33	B

Figure 1. An example data base.

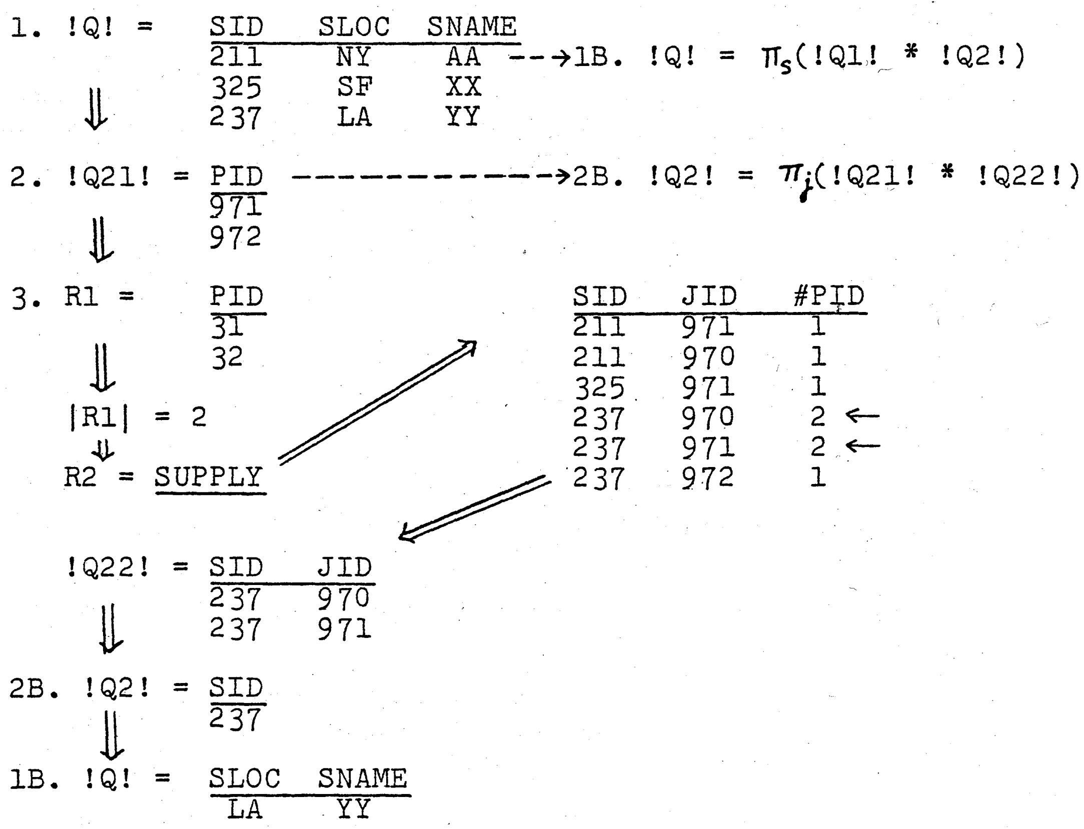

Figure 2. The process of the query evaluation.

SUPPLYALL	SID	PTYPE	JID
	237	A	971
	211	B	970
	237	B	971
	237	A	970

SUPPLYSOME	SID	PID	JID
	211	31	971
	325	32	971
	211	31	972
	237	31	972

Figure 3. The decomposed relations of SUPPLY.