STUDIES ON FUNCTIONAL AND MULTIVALUED DEPENDENCIES IN RELATIONAL DATABASES

Minoru Ito

Department of Information and Computer Sciences
Faculty of Engineering Science
Osaka University

September 1983

ABSTRACT

In the design theory of relational databases, functional dependencies (FDs) and multivalued dependencies (MVDs) are the most fundamental and important constraints. In this thesis, we deal with three topics on these dependencies.

(1) Database scheme design: Recently, the representative instance has been proposed as a suitable model for representing the "current" value of a database scheme under the weak universal instance assumption. Let $\underline{R} = \{\langle R_1,F_1\rangle, \dots, \langle R_n,F_n\rangle\}$ be a database scheme, where each R_i is a set of attributes and F_i is a set of FDs over R_i. We show that (a) it can be determined in $O(n|F|\|F\|)$ time whether $\underline{R}$ is consistent, where $F = F_1 \cup \dots \cup F_n$, $|F|$ is the number of FDs in F, and $\|F\|$ is the size of the description of F, and that (b) given a subset V of $R_1 \cup \dots \cup R_n$, we can construct in $O(n|F|\|F\|)$ time a relational expression whose value is the total projection of the representative instance onto V for every database instance of $\underline{R}$, provided that $\underline{R}$ is consistent.

(2) Dependency implication: Let R be a set of attributes and let $U_n \subsetneq \dots \subsetneq U_1 \subsetneq U_0 \subseteq R$. Let $D = F \cup M \cup M_1 \cup \dots \cup M_n$, where (a) F is a set of FDs $Z \rightarrow W$ satisfying $Z \subseteq U_0$ or $W \cap U_0 = \emptyset$, (b) M is a set of MVDs $Z \twoheadrightarrow W$ over T satisfying at least one of $Z \subseteq U_0 \subseteq T$, $W \cap U_0 = \emptyset$, and $(T - (Z \cup W)) \cap U_0 = \emptyset$, and (c) each M_i, $1 \leqq i \leqq n$, is a set of MVDs over U_i. Let d be an MVD $X \twoheadrightarrow Y$ over V (or an FD $X \rightarrow Y$). We present the following results on the problem of determining whether D implies d.

If $X,Y \subseteq V \subseteq U_0$, then the problem is solvable. If $X,Y \subseteq V \subseteq U_n$, then the problem can be solved in $O(\|D\| \cdot |Y - X|)$ time. If $X \subseteq U_n$ and $X,Y \subseteq V \subseteq U_0$, then the problem can be solved in $O(\|D\|^2 \cdot |U_n - X| \cdot \prod_{i=1}^{n-1}(|U_i - U_{i+1}| + 1))$ time.

(3) View dependency implication: A query can be formulated in terms of a relational algebra expression using projection, selection, restriction, cross product, and union. We show that it is NP-complete to determine whether given a database scheme $\underline{R}$, a database I of $\underline{R}$, and a relational expression E, view E(I) is not empty. And we show that (a) it is NP-complete to determine whether given a database scheme $\underline{R}$, a database I of $\underline{R}$, a relational expression E, and a tuple μ, view E(I) contains μ, but that (b) if E contains no projection, then it can be determined in polynomial time.

Next, we consider the problem of determining whether a given dependency d is valid in a given relational expression E over a given database scheme $\underline{R}$, and present the following results.

Case1: The case where each relation scheme in $\underline{R}$ is associated with FDs and d is an FD. Then the complement of the problem is NP-complete. If E contains no union, then the problem can be solved in polynomial time. Under the condition that at most two distinct values occur in any database instance of $\underline{R}$, the complement of the problem is NP-complete (even if E contains no union).

Case2: The case where each relation scheme in $\underline{R}$ is associated with FDs and full MVDs and d is an FD or a full MVD. Then the problem is solvable. Even if E consists only of selections and cross products, the problem is NP-hard. If E contains no union and each relation scheme name in $\underline{R}$ occurs at most once in E, then the problem can be solved in polynomial time.

ACKNOWLEDGMENTS

I would like to thank Professor Tadao Kasami for his continuous encouragements, sharp ideas and valuable suggestions.

I would also like to thank Associate Professor Kenichi Taniguchi for his helpful advice, comments and discussions.

I am grateful to Dr. Kenichi Hagihara for giving me many insights into the problems addressed in this thesis.

Finally, I thank Mr. Motoaki Iwasaki of Hitachi Ltd. for his useful assistance and discussions.

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

In relational databases, the notion of dependency, which is a constraint on relations, is central to the design of database schemes. Functional dependencies (FDs) [Codd 70] [Armstrong 74] and multivalued dependencies (MVDs) [Fagin 77] [Zaniolo 76] are the most fundamental and important dependencies. In this thesis, we deal with the following three topics on FDs and MVDs: (1) database scheme design, (2) dependency implication, and (3) view dependency implication. These are described below.

1.1 Database Scheme Design

In the design theory of relational databases, the "real world" is modeled by a single universal relation scheme <U,D>, where U is a set of attributes and D is a set of constraints over U. The database scheme representing the real world is defined by an ordered set $\underline{R} = \{<R_1,D_1>, \ldots, <R_n,D_n>\}$ of relation schemes, where $U = R_1 \cup \ldots \cup R_n$ and each D_i is a set of constraints that is "inherited" from D [Beeri et al 78]. Then an ordered set $I = \{r_1, \ldots, r_n\}$ of relations is called a database of $\underline{R}$ if each r_i is a relation over R_i. Furthermore, if each r_i satisfies D_i, then I is called a database instance of $\underline{R}$. It can be considered that a database instance of $\underline{R}$ represents the "current" value of the universal relation scheme <U,D> in some way. It has been often assumed that for a database instance $I = \{r_1, \ldots, r_n\}$ of $\underline{R}$, there must be a single relation r over U, called a pure universal instance for I, such that (1) r satisfies D and (2) each r_i coincides with the projection of r onto R_i. Then the relation r is considered as the "current" value of the universal relation scheme <U,D>.

However, the pure universal instance assumption is controversial and there are some criticisms [Beeri et al 78] [Kent 81]. Recently, a weakened version of this assumption has been proposed, which states that for a database instance $I = \{r_1, \ldots, r_n\}$ of $\underline{R}$, there must be a single relation r over U, called a <u>weak universal instance</u> for I, such that (1) r satisfies D and (2) each r_i is contained in the projection of r onto R_i [Honeyman 82] [Sagiv 81] [Ullman et al 82]. Under the weak universal instance assumption, the <u>representative instance</u> of I is a suitable model for representing the "current" value of <U,D> [Ullman et al 82]. It is known that there is a weak universal instance for a database instance I of $\underline{R}$ if and only if the representative instance of I satisfies D [Honeyman 82] [Ullman et al 82]. In this thesis, we assume a weak universal instance, but not a pure universal instance.

An important principle for designing a database scheme $\underline{R} = \{<R_1,D_1>, \ldots, <R_n,D_n>\}$ from a given universal relation scheme <U,D> is that for every database instance of $\underline{R}$, there is a weak universal instance; that is, the representative instance of every database instance of $\underline{R}$ always satisfies D. Because if so, then the global consistency of a database $I = \{r_1, \ldots, r_n\}$ of $\underline{R}$ depends only on the local consistency of each relation r_i; that is, there is no interrelational constraint among the relation scheme in $\underline{R}$ [Beeri et al 78] [Maier et al 80].

Consider the case where only FDs are given as constraints, and a cover of the FDs is embodied in the database scheme. That is, for given universal relation scheme <U,F> and database scheme $\underline{R} = \{<R_1,F_1>, \ldots, <R_n,F_n>\}$, a cover of F is equivalent to that of $F_1 \cup \ldots \cup F_n$, where F, $F_1, \ldots, F_n$ are sets of FDs. We say that a database scheme $\underline{R} = \{<R_1,F_1>, \ldots, <R_n,F_n>\}$ is <u>consistent</u> if the representative instance of every database instance $I = \{r_1, \ldots, r_n\}$ of $\underline{R}$ always satisfies $F_1 \cup \ldots \cup F_n$. The notion of consistency is equivalent to the notion that "local consistency implies global consistency" in [Sagiv 83] and the notion that "$\underline{R}$ is independent with respect to $F_1 \cup \ldots \cup F_n$" in [Graham and Yannakakis 82]. In this thesis, we consider the following two problems.

(1) Determine whether $\underline{R}$ is consistent.

(2) Given a subset V of $R_1 \cup \ldots \cup R_n$ and a database instance I of $\underline{R}$, how can we compute efficiently the <u>total projection</u> of the representative instance onto V?

The computation of the total projection is important for evaluating a query that refers to the set V with respect to the representative instance [Sagiv 83] [Ullman et al 82].

Sagiv [Sagiv 83] showed some results on these problems. As for problem (1), he showed a necessary and sufficient condition, called the uniqueness condition, for $\underline{R}$ to be consistent under the restriction that each $\langle R_i, F_i \rangle$ is a Boyce-Codd normal form scheme, that is, the left-hand side of every FD in F_i is the key of R_i. As for problem (2), he showed a quadratic algorithm for constructing a relational expression whose value is the total projection of the representative instance onto V for every database instance I of $\underline{R}$, provided that $\underline{R}$ satisfies the uniqueness condition. The relational expression consists of projections, extension joins [Honeyman 80], and unions, and thus its value for a database instance of $\underline{R}$ can be computed efficiently. And then he showed a quadratic algorithm for minimizing the number of unions and joins of the relational expression. The following negative results on a Boyce-Codd normal form are known [Beeri and Bernstein 79].

(a) There is a universal relation scheme that can not be transformed into any Boyce-Codd normal form database scheme. And it is NP-hard to determine whether a given universal relation scheme can be transformed into a Boyce-Codd normal form database scheme.

(b) It is NP-complete to determine whether a given relation scheme is not a Boyce-Codd normal form scheme.

As for problem (1), Graham and Yannakakis [Graham and Yannakakis 82] and, independently, the authors [Ito et al 83b] showed polynomial time algorithms for determining whether a given database scheme $\underline{R}$ is consistent with no restriction on $\underline{R}$. The algorithm of [Graham and Yannakakis 82] requires repeated tableau computations. The basic idea of the authors' algorithm is essentially the same as that of theirs, but the authors' algorithm is simpler and easier to implement, since no tableau computation is needed.

In this thesis, we show the following results, which are based on [Ito et al 83b] [Iwasaki et al 82].

(1) It can be determined in $O(n|F|\|F\|)$ time whether $\underline{R}$ is consistent, where $F = F_1 \cup \ldots \cup F_n$, $|F|$ is the number of FDs in F, and $\|F\|$ is the size of the description of F.

(2) We can construct in $O(n|F|\|F\|)$ time a relational expression whose value is the total projection of the representative instance onto V for every database instance of $\underline{R}$, provided that $\underline{R}$ is consistent. The relational expression consists of projections, extension joins, and unions.

(3) The relational expression above can be transformed in $O(n|F|\|F\|)$ time into a "simplified" relational expression in which (i) the relational expression contains neither redundant unions nor redundant joins and (ii) the projections of the relational expression are executed as early as possible when evaluating the relational expression for a database instance of $\underline{R}$. Note that the time and space for evaluating the relational expression can be saved by executing the projections as early as possible.

The topic of this section is discussed in Chapter 2. Result (1) above is shown in Section 2.2. Results (2) and (3) are shown in Sections 2.3.1 and 2.3.2, respectively.

1.2 Dependency Implication

Implication problem for dependencies is the problem of determining whether a given set D of dependencies implies another given dependency d, that is, whether whenever a relation satisfies D, it always satisfies d. The importance of this problem is summarized in [Ullman 81]. Implication problem for FDs can be solved in linear time [Beeri and Bernstein 79], and implication problem for FDs and full MVDs can be solved in polynomial time [Beeri 80] [Galil 82] [Hagihara et al 79] [Sagiv 80]. To the author's knowledge, however, it is open whether implication problem for embedded MVDs is solvable. The following possibly negative results on this problem are known, and it seems difficult to solve this problem completely.

(a) There is no finite set of inference rules that is complete for embedded MVDs [Parker and Parsaye-Ghomi 80] [Sagiv and Walecka 82]. Note that there is a complete set of inference rules for FDs and full MVDs [Beeri et al 77].

(b) Implication problem for template dependencies is unsolvable [Gurevich and Lewis 82] [Vardi 82]. Note that a template dependency is a generalization of embedded MVDs [Sadri and Ullman 82].

In this thesis, we show some restricted solutions on the implication problem for FDs and embedded MVDs, which are based on [Ito et al 80] [Ito et al 83a].

We denote an MVD over a set V of attributes by $X \twoheadrightarrow Y(V)$. Let R be a set of attributes and let $U_n \subsetneqq \ldots \subsetneqq U_1 \subsetneqq U_0 \subseteq R$. Let $D = F \cup M \cup M_1 \cup \ldots \cup M_n$, where

(i) F is a set of FDs $Z \rightarrow W$ satisfying $Z \subseteq U_0$ or $W \cap U_0 = \emptyset$,

(ii) M is a set of MVDs $Z \twoheadrightarrow W(V)$ satisfying at least one of $Z \subseteq U_0 \subseteq V$, $W \cap U_0 = \emptyset$, and $(V - (Z \cup W)) \cap U_0 = \emptyset$, and

(iii) each M_i for $1 \leqq i \leqq n$ is a set of MVDs over U_i.

Note that M may contain full MVDs. Then we have the following three results.

(1) Let $X,Y \subseteq V \subseteq U_0$. It is decidable whether $X \twoheadrightarrow Y(V)$ (or $X \rightarrow Y$) is implied by D.

(2) Let $X,Y \subseteq V \subseteq U_n$. It can be determined in $O(\|D\|.|Y - X|)$ time whether $X \twoheadrightarrow Y(V)$ (or $X \rightarrow Y$) is implied by D.

(3) Let $X \subseteq U_n$ and let $X,Y \subseteq V \subseteq U_0$. It can be determined in $O(\|D\|^2.|U_n - X|.\prod_{i=1}^{n-1}(|U_i - U_{i+1}| + 1))$ time whether $X \twoheadrightarrow Y(V)$ (or $X \rightarrow Y$) is implied by D.

The topic of this section is discussed in Chapter 3. Result (1) is shown in Section 3.2.1. Results (2) and (3) in the case of $X \twoheadrightarrow Y(V)$ are shown in Sections 3.2.2 and 3.2.3, respectively. Results (2) and (3) in the case of $X \rightarrow Y$ are shown in Section 3.2.4. Finally in Section 3.3.2, an extension of result (1) to a class of functional and template dependencies is shown, which is based on [Ito et al 81b].

1.3 View Dependency Implication

Relational algebra is known as a query language for relational databases. It has six operators on relations; projection, selection, restriction, cross product, union, and set difference [Codd 72]. A query can be formulated in terms of a relational expression consiting of the above six operators and relation scheme names in a given database scheme as operands [Ullman 80]. In this thesis, set difference is not considered. A relational expression E is considered as a mapping from databases I to relations E(I) called <u>views</u>.

We first consider the following two decision problems on views, whcih are the most fundamental problems in query processing.

(a) View nonemptiness problem: Given a database scheme $\underline{R}$, a database I of $\underline{R}$, and a relational expression E, determine whether view E(I) is not empty.

(b) Tuple membership problem: Given a database scheme $\underline{R}$, a database I of

$\underline{R}$, a relational expression E, and a tuple μ, determine whether μ is in E(I).

We show the following results, which are based on [Ito et al 82].

(1) Both problems are NP-complete in general.

(2) If E contains no projection, then the tuple membership problem can be solved in polynomial time.

Let $\underline{R}$ = $\{<R_1,D_1>, \ldots, <R_n,D_n>\}$ be a database scheme, let E be a relational expression, and let d be a dependency. Then d is said to be valid in E over $\underline{R}$ if for every database instance I = $\{r_1, \ldots, r_n\}$ of $\underline{R}$, view E(I) always satisfies d. We consider the problem, called the implication problem for view dependencies, of determining whether a given dependency d is valid in a given relational expression E over a given database scheme $\underline{R}$. This problem can be divided into the following two cases.

Case1: Each relation scheme in $\underline{R}$ is associated only with FDs (that is, $\underline{R}$ is of the form $\{<R_1,F_1>, \ldots, <R_n,F_n>\}$) and d is an FD.

Case2: Each relation scheme in $\underline{R}$ is associated with full MVDs as well as FDs (that is, $\underline{R}$ is of the form $\{<R_1,F_1 \cup M_1>, \ldots, <R_n,F_n \cup M_n>\}$) and d is an FD or a full MVD.

The importance of the implication problem for view dependencies is stated in [Klug 80] and [Klug and Price 82]. Klug [Klug 80] showed that in Case1, this problem is solvable (that is, it is decidable whether d is valid in E over $\underline{R}$). And then Klug and Price [Klug and Price 82] and, independently, the authors [Ito et al 81] showed that in Case2, this problem is still solvable. As for this problem, we show the following results, which are based on [Ito et al 81a] [Ito et al 82] [Ito et al 83c].

(3) In Case2, the implication problem for view dependencies is solvable.

(4) In Case1, the complement of the implication problem for view dependencies is NP-complete. That is, it is NP-complete to determine whether d is not valid in E over $\underline{R}$.

(5) In Case2, even if E consists only of selections and cross products, the implication problem for view dependencies is NP-hard. (The only known algorithm for solving this problem is exponential in time and space [Maier et al 79].)

(6) In Case1, if E contains no union, then the implication problem for view dependencies can be solved in polynomial time.

(7) In Case2, if E contains no union and each relation scheme name R_i in $\underline{R}$ occurs at most once in E, then the implication problem for view dependencies can be solved in polynomial time.

A scheme design method in relational databases is to decompose a given relation scheme <R,D> into a set $\{\langle R_1,D_1\rangle, \ldots, \langle R_n,D_n\rangle\}$ of smaller relation schemes (cf. [Beeri 79] [Beeri et al 78] [Rissanen 77]). It is important to examine whether the decomposition preserves the original constraints D [Beeri 79] [Maier et al 80]. This examination can be generalized as the problem of determining whether a given dependency d is valid in $R_1 \bowtie \ldots \bowtie R_n$, where $R_i \bowtie R_j$ is the natural join of R_i and R_j. It is known that in Case1 above, this problem can be solved in polynomial time [Rissanen 77] [Maier et al 80]. However, in Case2, it has not been known whether this problem can be solved in polynomial time. As a corollary of result (7) above, we show the following result.

(8) In Case2, it can be determined in polynomial time whether a given FD or a full MVD is valid in $R_1 \bowtie \ldots \bowtie R_n$.

When considering the implication problem for (view) dependencies, we usually assume that the domains of the values in databases are infinite. However, in practice, we must often consider finite domains (e.g., domain {male, female} or {sunday, monday, ..., saturday}). We say that d is k-valid in E over $\underline{R}$ if for every database instance I of $\underline{R}$ in which at most k distinct values occur, view E(I) satisfies d. It is possible that d is not valid but k-valid in E over $\underline{R}$. Theoretically, 2-validity is the simplest case in finite domains, since if only one value occurs in a database instance I of $\underline{R}$, then E(I) trivially satisfies any dependency d, and thus 1-validity is meaningless. Finally we show the following result.

(9) In Case1, even if E consists only of selections, restrictions, and cross products, the problem of determining whether d is not 2-valid in E over $\underline{R}$ is NP-complete. (Note result (6) above.)

The topic of this section is discussed in Chapter 4. Results (1) and (2) are shown in Sections 4.2.1 and 4.2.2, respectively. Results (3) through (6) are shown in Sections 4.3.1 through 4.3.4, respectively. Results (7) and (8) are shown in Section 4.3.5. Result (9) is shown in Section 4.3.6.

CHAPTER 2

SOME RESULTS ON THE REPRESENTATIVE INSTANCE

In this chapter, we discuss the topic of the representative instance. In Section 2.1, we provide basic definitions. In Section 2.2, we present a polynomial time algorithm for determining whether a given database scheme is consistent. In Section 2.3, we present a polynomial time algorithm for constructing a relational expression whose value is the total projection of the representative instance onto a given set of attributes, provided that the database scheme is consistent. And then a polynomial time simplification method of the relational expression is presented.

2.1 Definitions

A <u>relation</u> r over a set $R = \{A_1, \ldots, A_m\}$ of <u>attributes</u> is a finite set of <u>tuples</u> that are members of the Cartesian product $dom(A_1) \times \ldots \times dom(A_m)$, where $dom(A_i)$ is the domain of values of A_i. A relation can be viewed as a table such that each row is a tuple and each column is labeled by an attribute. Let μ be a tuple of r. For an attribute A in R, $\mu[A]$ denotes the value of μ in A column. For a subset X of R, $\mu[X]$ denotes the values of μ in X columns. We use A, B, C, ... for attributes, and ..., X, Y, Z for sets of attributes. We often write A for the singleton set $\{A\}$, and XY for the union $X \cup Y$.

A <u>functional dependency</u> (over R) [Armstrong 74] [Codd 70], abbreviated to FD, is a statement $X \rightarrow Y$, where X and Y are subsets of R. A relation r is said to <u>satisfy</u> $X \rightarrow Y$ if for all tuples μ and ν of r, $\mu[X] = \nu[X]$ implies $\mu[Y] = \nu[Y]$. A set F of FDs is said to <u>imply</u> an FD f if whenever a relation satisfies all FDs in F, it also satisfies FD f. For a set X of attributes, the <u>closure</u> of X with respect to F is a set of attributes defined by $\mathfrak{F}(X,F)$

= {A | F implies X → A}. We can compute $\mathcal{F}(X,F)$ in linear time [Beeri and Bernstein 79].

In the following we often consider a relation with variables. That is, a tuple of a relation may contain variables in some columns. For two tuples μ and ν, μ[A] = ν[A] if and only if μ and ν have either the same constant or the same variable in A column. We say that μ and ν agree in X columns if μ[X] = ν[X].

Let r be a relation that may contain variables and let F be a set of FDs. The chase of r under F is a relation obtained by applying FD-rules, which are defined below, for F to r until no rule can be applied anymore [Aho et al 79] [Maier et al 79]. An application sequence of FD-rules for F to r is called a chase process of r under F.

FD-rules: An FD X → Y in F has an associated rule as follows. Suppose that there are two tuples μ and ν that agree in X columns. FD-rule for X → Y executes the following for each attribute A in Y - X.

(1) If μ (or ν) has a variable v in A column and ν (or μ) has a constant c in that column, then replace all occurrences of v in A column with c.

(2) If μ and ν have different variables v_1 and v_2 in A column, then replace all occurrences of v_1 in A column with v_2.

If μ and ν have different constants in A column, then μ and ν are said to conflict (for X → Y). In this case, the chase of r under F does not satisfy F. By FD-rule for X → Y, μ and ν will be equated in Y columns unless μ and ν conflict. The chase of r under F satisfies F if and only if no confliction occurs by any chase process of r under F. If the chase satisfies F, then it is unique up to renaming of variables [Maier et al 79].

A relation scheme is a pair <R,F> of a set R of attributes and a set F of FDs over R. A database scheme over a set U of attributes is an ordered set $\underline{R}$ = $\{<R_1,F_1>, \ldots, <R_n,F_n>\}$ of relation schemes such that U = $R_1 \cup \ldots \cup R_n$. An ordered set I = $\{r_1, \ldots, r_n\}$ of relations is called a database of $\underline{R}$ if each r_i is a relation over R_i. Furthermore if each r_i satisfies F_i, then I is called a database instance of $\underline{R}$. In this chapter, we mainly consider database instances, and assume that no database of $\underline{R}$ contains any variable.

We assume that F (= $F_1 \cup \dots \cup F_n$) is a cover of all the FDs imposed on the database by the user, that is, a cover of the FDs is embodied in the database scheme. Given a universal relation scheme <U,F> and a decomposition $\{R_1, \dots, R_n\}$ of U, it can be determined in polynomial time whether a cover of F is embodied by the decomposition, that is, whether there is a database scheme $\underline{R} = \{<R_1,F_1>, \dots, <R_n,F_n>\}$ over U such that a cover of $F_1 \cup \dots \cup F_n$ is equivalent to that of F [Beeri and Honeyman 81]. This assumption implies Assumption2.1 below.

Assumption2.1: If an FD $X \rightarrow Y$ is implied by F and $XY \subseteq R_i$, then $X \rightarrow Y$ is also implied by F_i.

Let $I = \{r_1, \dots, r_n\}$ be a database instance of $\underline{R}$. Each r_i can be viewed as a relation over U, denoted $aug_U(r_i)$, by adding columns for the attributes in $U - R_i$ that contain distinct variables. That is, for each tuple μ of r_i, there is a tuple of $aug_U(r_i)$, denoted $aug_U(\mu)$, that agrees with μ in R_i columns and has distinct variables (that do not appear in any other tuple) for the attributes in $U - R_i$. We define $aug_U(I) = aug_U(r_1) \cup \dots \cup aug_U(r_n)$. We assume that each variable occurs once in only one tuple of $aug_U(I)$. The representative instance of the database I, denoted rep(I), is defined as the chase of $aug_U(I)$ under F [Honeyman 82] [Sagiv 81] [Vassiliou 80]. The database scheme $\underline{R}$ is said to be consistent if for every database instance I of $\underline{R}$, rep(I) satisfies F, that is, no confliction occurs by any chase process of $aug_U(I)$ under F.

For simplicity, we have the following two assumptions.

Assumption2.2: Each F_i satisfies the following conditions. For all $X \rightarrow Y$ in F_i,

(a) $Y = \mathcal{F}(X,F_i) - X$,

(b) $X \rightarrow Y$ is not implied by $F_i - \{X \rightarrow Y\}$, and

(c) for no proper subset X' of X, $X' \rightarrow Y$ is implied by F_i.

A quadratic algorithm for transforming a set of FDs into the set satisfying conditions (a), (b), and (c) above is known [Bernstein 76].

<u>Assumption2.3</u>: Let $X \rightarrow Y$ and $Z \rightarrow W$ be FDs in F. If $XY \subseteq ZW$, then $X \rightarrow Y$ and $Z \rightarrow W$ are in the same set (that is one of $F_1, \ldots, F_n$).

Note that it can be determined in $O(|F|\|F\|)$ time whether F satisfies Assumption2.3. If F does not satisfy Assumption2.3, then $\underline{R}$ is not consistent, as explained below. Suppose that there are two FDs $X \rightarrow Y$ and $Z \rightarrow W$ such that (1) $XY \subseteq ZW$ and (2) $X \rightarrow Y$ and $Z \rightarrow W$ are in different sets F_i and F_j, respectively. Then F_j as well as F_i imply $X \rightarrow Y$ by Assumption2.1. Consider a database instance $I = \{r_1, \ldots, r_n\}$ of $\underline{R}$ such that (1) r_i consists of only one tuple that has a constant c in all the columns, (2) r_j consists of only one tuple that has the constant c exactly in X columns (and another constants in $R_j - X$ columns), and (3) any other relation is empty. Then a confliction for $X \rightarrow Y$ occurs in $aug_U(I)$, and thus $\underline{R}$ is not consistent.

2.2 Testing Consistency of a Database Scheme

In this section we present an algorithm for determining whether a given database scheme is consistent.

2.2.1 Conditions for consistency of a database scheme

Let $\underline{R} = \{\langle R_1,F_1\rangle, \ldots, \langle R_n,F_n\rangle\}$ be a database scheme over U. In this section we present some conditions that are useful for developing an algorithm for determining whether $\underline{R}$ is consistent.

Let $I = \{r_1, \ldots, r_n\}$ be a database instance of $\underline{R}$. Consider a chase process of $\mathrm{aug}_U(I)$ under F. If a tuple μ of $\mathrm{aug}_U(I)$ is transformed into a tuple μ' by a number of applications of FD-rules for F, then μ is said to be *extended* to μ' by F, and μ' is called an *extension* of μ. An application of FD-rule for an FD $X \to Y$ in F_i to μ and ν that agree in X columns is said to be *restricted* if either μ or ν is an extension of a tuple of $\mathrm{aug}_U(r_i)$. If μ is the extension, then ν is equated to μ in Y columns by the restricted application unless μ and ν conflict, and μ remains unchanged. Let ν' be the resulting tuple. Then ν' agrees with ν in U - Y columns and agrees with μ in XY columns. We denote the restricted application by $\nu \xrightarrow[\mu]{X \to Y} \nu'$ (or simply $\nu \xrightarrow{X \to Y} \nu'$). If μ and ν conflict for $X \to Y$ in F_i (that is, μ and ν agree in X columns but have different constants in A column for an attribute A in Y) and if either μ or ν is an extension of a tuple of $\mathrm{aug}_U(r_i)$, then the confliction is said to be *restricted*. Then we have the following lemma, whose proof is given in Appendix 1.

[Lemma2.1] If $\underline{R}$ is not consistent, then there is a database instance I of $\underline{R}$ such that a restricted confliction occurs by extending only one tuple of $\mathrm{aug}_U(I)$ by only restricted applications of FD-rules for F and leaving all other tuples unchanged. []

For a relation scheme $\langle R_i, F_i \rangle$, a sequence $X_1 \to Y_1, \ldots, X_m \to Y_m$ of FDs in $F - F_i$ is called a *derivation* of a subset V of U from R_i if $X_k \subseteq R_i Y_1 \ldots Y_{k-1}$ for $1 \leqq k \leqq m$ and $V \subseteq R_i Y_1 \ldots Y_m$. If $X_1 \to Y_1, \ldots, X_m \to Y_m$ is a derivation of V from R_i, then the set $\{X_1 \to Y_1, \ldots, X_m \to Y_m\}$ implies $R_i \to Y_1 \ldots Y_m$. And then $R_i \to Y_1 \ldots Y_m$ implies $R_i \to V$. Thus it holds that $V \subseteq \mathcal{F}(R_i, F)$. In this section, we consider only the case where V is a singleton set $\{A\}$ and Y_m contains A. Such a derivation is called a derivation of A from R_i.

A derivation $X_1 \to Y_1, \ldots, X_m \to Y_m$ of A from R_i is said to be *close* if it satisfies the following property.

Property2.1: For each $X_k \to Y_k$ with $1 \leqq k \leqq m$, if there is an FD $X \to Y$ in F such that $XY \subsetneqq X_kY_k$ and $X \subseteq R_iY_1...Y_{k-1}$, then there is an FD $X_\ell \to Y_\ell$ such that $1 \leqq \ell \leqq k-1$ and $XY \subseteq X_\ell Y_\ell$.

Property2.1 is restated as follows. For an FD $X \to Y$ in F_j, we define $\text{cover}(X \to Y) = \{Z \to W \mid Z \to W$ is in F_j and $ZW \subseteq XY\}$. Suppose that $X_k \to Y_k$ is in F_j and let $H_j^{(k)}$ be the intersection of F_j and $\{X_1 \to Y_1, ..., X_{k-1} \to Y_{k-1}\}$. We define $\text{cover}(H_j^{(k)}) = \bigcup_{X \to Y \text{ in } H_j^{(k)}} \text{cover}(X \to Y)$. Then Property2.1 is equivalent to the condition that for each $X_k \to Y_k$ with $1 \leqq k \leqq m$, if there is an FD $X \to Y$ in F_j such that $XY \subsetneqq X_kY_k$ and $X \subseteq R_iY_1...Y_{k-1}$, then $X \to Y$ is in $\text{cover}(H_j^{(k)})$. That is, $X_k \to Y_k$ is a minimal FD in $F_j - \text{cover}(H_j^{(k)})$ satisfying $X_k \subseteq R_iY_1...Y_{k-1}$.

Let $X_1 \to Y_1, ..., X_m \to Y_m$ be a derivation of A from R_i. Suppose that the last FD $X_m \to Y_m$ is in F_j and let $H_j^{(m)}$ be the intersection of F_j and $\{X_1 \to Y_1, ..., X_{m-1} \to Y_{m-1}\}$. $X_m \to Y_m$ is said to be irreducible (with respect to the derivation $X_1 \to Y_1, ..., X_m \to Y_m$ of A from R_i) if $X_m \to A$ is not implied by $\text{cover}(H_j^{(m)})$.

[Lemma2.2] If $\underline{R}$ is not consistent, then one of the following holds.

(i) There is a close derivation $X_1 \to Y_1, ..., X_m \to Y_m$ of an attribute in R_i from R_i itself such that $X_m \to Y_m$ is irreducible.

(ii) There are two close derivations $Z_1 \to W_1, ..., Z_s \to W_s$ and $P_1 \to Q_1, ..., P_t \to Q_t$ of an attribute in U from R_i such that $Z_s \to W_s$ and $P_t \to Q_t$ are irreducible and different.

(Proof) If $\underline{R}$ is not consistent, then there is a database instance $I = \{r_1, ..., r_n\}$ of $\underline{R}$ that satisfies condition of Lemma2.1. Suppose that a tuple ν_1 of $\text{aug}_U(r_i)$ is extended to a tuple ν_m by restricted applications of FD-rules for $X_1 \to Y_1, ..., X_{m-1} \to Y_{m-1}$ in this order and suppose that ν_m

restrictedly conflicts with a tuple μ_m of $\mathrm{aug}_U(r_j)$ for an FD $X_m \to Y_m$ in F_j. That is, ν_m and μ_m agree in X_m columns but have different constants in A column for an attribute A in Y_m. The chase process is denoted $\nu_1 \xRightarrow[\mu_1]{X_1 \to Y_1} \nu_2 \xRightarrow[\mu_2]{X_2 \to Y_2} \ldots \xRightarrow[\mu_{m-1}]{X_{m-1} \to Y_{m-1}} \nu_m$. For $1 \leq k \leq m$, ν_k has constants exactly in $R_iY_1\ldots Y_{k-1}$ columns and $X_k \subseteq R_iY_1\ldots Y_{k-1}$, as explained next. Initially ν_1 has constants exactly in R_i columns. If ν_k has constants exactly in $R_iY_1\ldots Y_{k-1}$ columns, then ν_{k+1} has constants exactly in $R_iY_1\ldots Y_k$ columns and $X_k \subseteq R_iY_1\ldots Y_{k-1}$ by $\nu_k \xRightarrow[\mu_k]{X_k \to Y_k} \nu_{k+1}$. Thus it holds that $X_mA \subseteq R_iY_1\ldots Y_{m-1}$ and sequence $X_1 \to Y_1, \ldots, X_m \to Y_m$ is a derivation of A from R_i. In the following we show that the derivation can be assumed to be close without loss of generality.

For FD $X_k \to Y_k$, if there is an FD $X \to Y$ in F such that $X \subseteq R_iY_1\ldots Y_{k-1}$ and $XY \subsetneq X_kY_k$, then $X \to Y$ and $X_k \to Y_k$ are in the same set by Assumption2.3. If μ_k and ν_k do not conflict for $X \to Y$, then there is a chase process $\nu_k \xRightarrow[\mu_k]{X \to Y} \nu' \xRightarrow[\mu_k]{X_k \to Y_k} \nu''$. Clearly ν'' coincides with ν_{k+1}. Thus the original sequence $X_1 \to Y_1, \ldots, X_m \to Y_m$ can be replaced by the sequence $X_1 \to Y_1, \ldots, X_{k-1} \to Y_{k-1}, X \to Y, X_k \to Y_k, \ldots, X_m \to Y_m$. If μ_k and ν_k conflict for $X \to Y$, then the confliction is restricted, and thus the original sequence can be replaced by the sequence $X_1 \to Y_1, \ldots, X_{k-1} \to Y_{k-1}, X \to Y$. By repeating the process above, we assume without loss of generality that the sequence $X_1 \to Y_1, \ldots, X_m \to Y_m$ is close.

Let $H_j^{(m)}$ be the intersection of F_j and $\{X_1 \to Y_1, \ldots, X_{m-1} \to Y_{m-1}\}$. Then relation $r_j \cup \{\nu_m[R_j]\}$ satisfies $\mathrm{cover}(H_j^{(m)})$ (if each variable of $\nu_m[R_j]$ is considered as a constant), by the following reason. Let $Z \to W$ be an FD in $\mathrm{cover}(X_k \to Y_k)$, where $X_k \to Y_k$ is in $H_j^{(m)}$. By $\nu_k \xRightarrow[\mu_k]{X_k \to Y_k} \nu_{k+1}$, tuple μ_k agrees with ν_{k+1} (and also ν_m) in X_kY_k columns, especially in ZW columns. Since r_j satisfies $Z \to W$ and $\mu_k[R_j]$ is in r_j, relation $r_j \cup \{\nu_m[R_j]\}$ satisfies $Z \to W$.

Since (1) ν_m agrees with μ_m in X_m columns, (2) $\mu_m[R_j]$ is in r_j and (3)

$r_j \cup \{\nu_m[R_j]\}$ satisfies cover($H_j^{(m)}$), ν_m agrees with μ_m in $\mathfrak{F}(X_m$,cover($H_j^{(m)}$)) columns. Since ν_m and μ_m have different constants in A column, $\mathfrak{F}(X_m$,cover($H_j^{(m)}$)) does not contain A, that is, cover($H_j^{(m)}$) does not imply $X_m \rightarrow A$. Thus $X_m \rightarrow Y_m$ is irreducible. If R_i contains A, then condition (i) of Lemma2.2 follows. Suppose that R_i does not contain A and let $X_k \rightarrow Y_k$ be the first FD such that Y_k contains A. Since ν_m has constants exactly in $R_iY_1...Y_{m-1}$ columns, $R_iY_1...Y_{m-1}$ contains A, and thus it holds that $k \leqq m-1$. Then subsequence $X_1 \rightarrow Y_1$, ..., $X_k \rightarrow Y_k$ is a close derivation of A from R_i such that $X_k \rightarrow Y_k$ is irreducible by the fact that none of Y_1, ..., Y_{k-1} contains A. If $X_k \rightarrow Y_k$ is not in F_j, then $X_m \rightarrow Y_m$ and $X_k \rightarrow Y_k$ are different, and thus condition (ii) of Lemma2.2 follows. If $X_k \rightarrow Y_k$ is in F_j, then it is also in $H_j^{(m)}$. Since $X_m \rightarrow Y_m$ is not in cover($H_j^{(m)}$), $X_m \rightarrow Y_m$ and $X_k \rightarrow Y_k$ are different. Thus condition (ii) of Lemma2.2 follows. []

Conversely we have the following lemma.

[Lemma2.3] If there is a derivation $X_1 \rightarrow Y_1$, ..., $X_m \rightarrow Y_m$ of an attribute A in U from R_i such that (1) $R_iY_1...Y_{m-1}$ contains A and (2) the last FD $X_m \rightarrow Y_m$ is irreducible, then $\underline{R}$ is not consistent.

(Proof) Suppose that $X_m \rightarrow Y_m$ is in F_j. Let $H_j^{(m)}$ be the intersection of F_j and $\{X_1 \rightarrow Y_1$, ..., $X_{m-1} \rightarrow Y_{m-1}\}$. We denote $H_j^{(m)}$ by $\{Z_1 \rightarrow W_1$, ..., $Z_s \rightarrow W_s\}$. In the following we show that there is a database instance I of $\underline{R}$ such that a restricted confliction occurs by extending only one tuple by restricted applications of FD-rules for $X_1 \rightarrow Y_1$, ..., $X_{m-1} \rightarrow Y_{m-1}$. We define I = $\{r_1, ..., r_n\}$ as follows.

(1) Each r_k except r_j consists of only one tuple that has a constant c in all the columns.

(2) $r_j = \{\mu_1, ..., \mu_s, \mu\}$, where each tuple μ_k for $1 \leqq k \leqq s$ has the

constant c in Z_kW_k columns and distinct constants (that do not appear in any other tuple) in all other columns, and μ has the constant c in $\mathfrak{F}(X_m,\mathrm{cover}(H_j^{(m)}))$ columns and distinct constants in all other columns.

Then I is a database instance of $\underline{R}$ by the following reason. It suffices to show that r_j satisfies F_j. Suppose that for an FD $X \to Y$ in F_j, there are two tuples μ_k and ν of r_j that agree in X columns, where ν is one of $\mu_1, \ldots, \mu_{k-1}, \mu_{k+1}, \ldots, \mu_s, \mu$. Then μ_k and ν have the constant c in X columns, and thus $X \subseteq Z_kW_k$. Thus $Z_k \to W_k$ and $X \to Y$ imply $Z_k \to W_kXY$. It follows from Assumption2.2(a) that $XY \subseteq Z_kW_k$. If $\nu = \mu_t$, then it holds that $XY \subseteq Z_tW_t$ by the same reason. If $\nu = \mu$, then it holds that $XY \subseteq \mathfrak{F}(X_m,\mathrm{cover}(H_j^{(m)}))$, since (1) $X \subseteq \mathfrak{F}(X_m,\mathrm{cover}(H_j^{(m)}))$ and (2) $XY \subseteq Z_kW_k$ implies that $X \to Y$ is in $\mathrm{cover}(H_j^{(m)})$. Thus μ_k and ν have the constant c in Y columns, that is, μ_k and ν satisfy $X \to Y$.

Let τ_1 be a tuple of $\mathrm{aug}_U(r_i)$. Then there is a chase process $\tau_1 \xRightarrow[\nu_1]{X_1 \to Y_1} \tau_2 \xRightarrow[\nu_2]{X_2 \to Y_2} \cdots \xRightarrow[\nu_{m-1}]{X_{m-1} \to Y_{m-1}} \tau_m$ such that τ_k for $1 \leqq k \leqq m$ has the constant c exactly in $R_iY_1\ldots Y_{k-1}$ columns, as explained next. Since $i \neq j$, initially τ_1 has the constant c exactly in R_i columns. Suppose that $X_k \to Y_k$ is in F_{j_k} and that τ_k has the constant c exactly in $R_iY_1\ldots Y_{k-1}$ columns. If $j_k = j$, then we can choose a tuple of $\mathrm{aug}_U(r_j)$ that has the constant c exactly in X_kY_k columns as ν_k, and otherwise ν_k has the constant c exactly in R_{j_k} columns. Thus by $\tau_k \xRightarrow[\nu_k]{X_k \to Y_k} \tau_{k+1}$, tuple τ_{k+1} has the constant c exactly in $R_iY_1\ldots Y_k$ columns. Since $X_m \subseteq R_iY_1\ldots Y_{m-1}$, τ_m agrees with $\mathrm{aug}_U(\mu)$ in X_m columns. But since $\mathrm{cover}(H_j^{(m)})$ does not imply $X_m \to A$ by the irreducibility of $X_m \to Y_m$, $\mathrm{aug}_U(\mu)$ does not have the constant c in A column. Thus τ_m restrictedly conflicts with $\mathrm{aug}_U(\mu)$ for $X_m \to Y_m$, and thus $\underline{R}$ is not consistent. []

2.2.2 The method

[Algorithm2.1]

input: A database scheme $\underline{R} = \{\langle R_1,F_1\rangle, \ldots, \langle R_n,F_n\rangle\}$.

method: If there is a number i such that the following procedure EXAM(R_i) returns "no", then $\underline{R}$ is not consistent, and otherwise (that is, if for all i, EXAM(R_i) returns "yes") $\underline{R}$ is consistent.

```
procedure EXAM($R_i$)
begin
  (1) Let $S = R_i$ (that is, assign $R_i$ to S). For $1 \leq j \leq n$, let $G_j = \emptyset$.
  (2) while there is a number $j$ ($\neq i$) such that $F_j - G_j$ contains an FD $X \to Y$ with $X \subseteq S$
      do begin
        (2-i) Select an FD $X \to Y$ from $F_j - G_j$ such that $X \subseteq S$ and $X \to Y$ is minimal (that is, there is no FD $Z \to W$ in $F_j - G_j$ such that $Z \subseteq S$ and $ZW \subsetneq XY$).
        (2-ii) If the FD $X \to Y$ selected in step (2-i) satisfies the following condition, then return "no".
          Condition1: $S \cap Y - \mathcal{F}(X,G_j) \neq \emptyset$.
        (2-iii) If the FD $X \to Y$ does not satisfy Condition1, then let $S = S \cup Y$ and $G_j = G_j \cup \mathrm{cover}(X \to Y)$.
      end while
  (3) (The case where step (2) terminates without returning "no") return "yes".
end EXAM  []
```

We show that Algorithm2.1 correctly determines whether $\underline{R}$ is consistent. We denote the values of S, $G_1, \ldots, G_n$ at the p-th execution of step (2-i) by $S^{(p)}, G_1^{(p)}, \ldots, G_n^{(p)}$, respectively. We denote the FD selected at the p-th execution of step (2-i) by $X^{(p)} \rightarrow Y^{(p)}$. Then since $X^{(p)} \subseteq S^{(p)} = R_i Y^{(1)} \ldots Y^{(p-1)}$, sequence $X^{(1)} \rightarrow Y^{(1)}, \ldots, X^{(p)} \rightarrow Y^{(p)}$ is a derivation of each attribute in $Y^{(p)}$ from R_i. Suppose that $X^{(p)} \rightarrow Y^{(p)}$ is in F_j and let $H_j^{(p)}$ be the intersection of F_j and $\{X^{(1)} \rightarrow Y^{(1)}, \ldots, X^{(p-1)} \rightarrow Y^{(p-1)}\}$. Then it holds that $G_j^{(p)} = \text{cover}(H_j^{(p)})$. And since $X^{(p)} \rightarrow Y^{(p)}$ is minimal in $F_j - G_j^{(p)}$, the derivation is close.

Suppose that EXAM(R_i) returns "no" at the p-th execution of step (2-ii), that is, $S^{(p)} \cap Y^{(p)} - \mathcal{F}(X^{(p)},G_j^{(p)}) \neq \emptyset$. Let A be an attribute in $S^{(p)} \cap Y^{(p)} - \mathcal{F}(X^{(p)},G_j^{(p)})$. Then $X^{(1)} \rightarrow Y^{(1)}, \ldots, X^{(p)} \rightarrow Y^{(p)}$ is a close derivation of A from R_i. Since the fact that $\mathcal{F}(X^{(p)},G_j^{(p)})$ does not contain A implies that cover($H_j^{(p)}$) does not imply $X^{(p)} \rightarrow A$, FD $X^{(p)} \rightarrow Y^{(p)}$ is irreducible. Since $S^{(p)}$ (= $R_i Y^{(1)} \ldots Y^{(p-1)}$) contains A, $\underline{R}$ is not consistent by Lemma2.3.

In order to prove the converse, we present two lemmas below. The proofs are given in Appendix 1.

[Lemma2.4] Let $X_1 \rightarrow Y_1, \ldots, X_m \rightarrow Y_m$ be a close derivation of A from R_i such that the last FD $X_m \rightarrow Y_m$ in F_j is irreducible. For a subet G of F_j, if G does not contain $X_m \rightarrow Y_m$, then G does not imply $X_m \rightarrow A$, that is, A is not in $\mathcal{F}(X_m,G)$. [ ]

[Lemma2.5] Let $X_1 \rightarrow Y_1, \ldots, X_m \rightarrow Y_m$ be a close derivation of an attribute in U from R_i such that the last FD $X_m \rightarrow Y_m$ in F_j is irreducible. If $X_m \rightarrow Y_m$ is not selected in step (2-i) during the execution of EXAM(R_i), then EXAM(R_i) returns "no". [ ]

Suppose that $\underline{R}$ is not consistent. By Lemma2.2 there are two cases to be considered.

<u>Case1</u>: Suppose that there is a close derivation $X_1 \rightarrow Y_1, \ldots, X_m \rightarrow Y_m$ of

an attribute A in R_i from R_i itself such that the last FD $X_m \to Y_m$ in F_j is irreducible. By Lemma2.5, it suffices to consider the case where $X_m \to Y_m$ is selected in step (2-i). Suppose that $X_m \to Y_m$ is selected at the p-th execution of step (2-i). Since $X_m \to Y_m$ is irreducible and $G_j^{(p)}$ does not contain $X_m \to Y_m$, $\mathcal{F}(X_m,G_j^{(p)})$ does not contain A by Lemma2.4. Since R_i contains A and $R_i \subseteq S^{(p)}$, it holds that $S^{(p)} \cap Y_m - \mathcal{F}(X_m,G_j^{(p)}) \neq \emptyset$. Thus EXAM($R_i$) returns "no" by Condition1 in step (2-ii).

<u>Case2</u>: Suppose that there are two close derivations $Z_1 \to W_1$, ..., $Z_s \to W_s$ and $P_1 \to Q_1$, ..., $P_t \to Q_t$ of an attribute A in U from a relation scheme R_i such that $Z_s \to W_s$ and $P_t \to Q_t$ are irreducible and different. By Lemma2.5, it suffices to consider the case where both $Z_s \to W_s$ and $P_t \to Q_t$ are selected in step (2-i). We assume without loss of generality that $Z_s \to W_s$ is selected at the p-th execution of step (2-i) after $P_t \to Q_t$ has been selected. Suppose that $Z_s \to W_s$ is in F_j. Since $Z_s \to W_s$ is irreducible and $G_j^{(p)}$ does not contain $Z_s \to W_s$, $\mathcal{F}(Z_s,G_j^{(p)})$ does not contain A by Lemma2.4. Since Q_t contains A and $Q_t \subseteq S^{(p)}$, it holds that $S^{(p)} \cap W_s - \mathcal{F}(Z_s,G_j^{(p)}) \neq \emptyset$. Thus EXAM($R_i$) returns "no" by Condition1 in step (2-ii).

We estimate the time complexity of Algorithm2.1. We assume that as the input of Algorithm2.1, each attribute in U is represented by an integer and all given sets of attributes (e.g., R_1, ..., R_n and X, Y for $X \to Y$ in F) are represented by increasing sequences of integers. Before executing the procedure EXAM(R_i), we execute the following (a), (b), and (c). (These can be executed in $O(|F|\|F\|)$ time.)

(a) For each $X \to Y$ in F_j with $1 \leq j \leq n$, list all the FDs $Z \to W$ in F_j such that $ZW \subseteq XY$, that is, cover($X \to Y$).

(b) For each A in U, list all the FDs $X \to Y$ in F such that X contains A.

(c) For each $X \to Y$ in F, we introduce variable count($X \to Y$) and let the initial value of count($X \to Y$) be $|R_i - X|$. We use count($X \to Y$) for examining whether S contains X.

When EXAM(R_i) is executed, the loop of step (2) is most expensive. The loop is repeated at most $|F|$ times. Consider how to select an FD $X \rightarrow Y$ in step (2-i). For each execution of the loop, if an attribute A is added to S, then we decrease the value of count($X \rightarrow Y$) by one for each $X \rightarrow Y$ such that X contains A. This can be done in the time proportional to the number of FDs in F whose left-hand sides contain A (such FDs have been listed in step (b) above). If count($X \rightarrow Y$) = 0, then X is contained in S. Since for each attribute A in U and each $X \rightarrow Y$ in F whose left-hand side contains A, the value of count($X \rightarrow Y$) is decreased by one at most once, this process can be executed in $O(\|F\|)$ time as a whole. Since we have listed cover($X \rightarrow Y$) for each $X \rightarrow Y$ in F_j in step (a), we can test in $O(|F_j|) \leqq O(|F|)$ time whether $X \rightarrow Y$ is minimal in $F_j - G_j$. This process can be executed in $O(|F|^2)$ time as a whole. Next in step (2-ii), we can examine Condition1 in $O(\|F_j\|)$ time, since $\mathcal{F}(X,G_j)$ can be computed in $O(\|G_j\|) \leqq O(\|F_j\|)$ time [Beeri and Bernstein 79]. Note that we examine Condition1 for each FD at most once. By the discussions above, EXAM(R_i) can be executed in $O(|F|\|F\|)$ time. Thus we have the following theorem.

[Theorem2.1] Let $\underline{R} = \{\langle R_1,F_1\rangle, \ldots, \langle R_n,F_n\rangle\}$ be a database scheme. It can be determined in $O(n|F|\|F\|)$ time whether $\underline{R}$ is consistent, where $F = F_1 \cup \ldots \cup F_n$. []

2.3 Computing the Total Projection

Let r be a relation over R and let V be a subset of R. The <u>projection</u> of r onto V is a relation over V defined by $r[V] = \{\mu[V] \mid \mu \text{ is in } r\}$. If r contains variables, then the <u>total projection</u> of r onto V is defined by $r[V\text{-total}] = \{\mu[V] \mid \mu \text{ is in } r \text{ and contains no variable in } V \text{ columns}\}$. Let

r_1 and r_2 be relations over R_1 and R_2, respectively. The (natural) join of r_1 and r_2 is a relation over $R_1 \cup R_2$ defined by $r_1 \bowtie r_2 = \{\mu \mid \mu[R_1]$ is in r_1 and $\mu[R_2]$ is in $r_2\}$. If r_2 satisfies FD $R_2 \cap R_1 \rightarrow R_2 - R_1$, then the join $r_1 \bowtie r_2$ is called an extension join [Honeyman 80]. Unlike usual joins, extension joins can be computed efficiently [Honeyman 80]. In this section, only the extension joins are considered.

Let $\underline{R} = \{\langle R_1,F_1\rangle, \ldots, \langle R_n,F_n\rangle\}$ be a database scheme over U. In this section, we assume that $\underline{R}$ is consistent unless otherwise stated. A relational expression consists of $R_1, \ldots, R_n$ as operands and projection, union and join as operators. Formally a relational expression is defined as follows.

(1) R_i is a relational expression by itself.

(2) If E_1 and E_2 are relational expressions, then so are $E_1[V]$, $E_1 \bowtie E_2$, and $E_1 \cup E_2$.

The value of a relational expression E for a database (not necessarily database instance) $I = \{r_1, \ldots, r_n\}$ of $\underline{R}$, denoted E(I), is computed by substituting $r_1, \ldots, r_n$ for $R_1, \ldots, R_n$, respectively, and applying the operators according to the definitions. Two relational expressions E_1 and E_2 are said to be equivalent if $E_1(I) = E_2(I)$ for every database instance I of $\underline{R}$. And E_1 is said to include E_2 if $E_2(I) \subseteq E_1(I)$ for every database instance I of $\underline{R}$.

In Section 2.3.1, we show how to construct in $O(n|F|\|F\|)$ time a relational expression E whose value is the total projection of the representative instance onto V, that is, E(I) = rep(I)[V-total] for every databse instance I of $\underline{R}$, provided that the database scheme $\underline{R}$ is consistent. The expression E is of the form $\bigcup_i E_i[V]$, where each E_i is a sequence of extension joins, and thus rep(I)[V-total] can be computed efficiently. In Section 2.3.2, we show how to obtain a simplified relational expression from E in $O(n|F|\|F\|)$ time.

2.3.1 The method

Let $\underline{R} = \{\langle R_1,F_1\rangle, \ldots, \langle R_n,F_n\rangle\}$ be a consistent database scheme over U. Let $I = \{r_1, \ldots, r_n\}$ be a database instance of $\underline{R}$ and let V be a subset of U. We have the following lemma, whose proof is given in Appendix 1.

[Lemma2.6] Let $aug_U(I)^*$ be a relation obtained by only restricted applications of FD-rules for F to $aug_U(I)$ until no FD-rule can be restrictedly applied anymore. Then it holds that rep(I)[V-total] = $aug_U(I)^*$[V-total]. []

Let $\mu[V]$ be a tuple of rep(I)[V-total], where μ is an extension of a tuple μ_0 of $aug_U(r_i)$. By Lemma2.6 there is a chase process $\mu_0 \xRightarrow{X_1 \to Y_1} \mu_1 \xRightarrow{X_2 \to Y_2} \ldots \xRightarrow{X_m \to Y_m} \mu_m$ such that μ_m agrees with μ in V columns. Conversely, if there is a chase process $\mu_0 \xRightarrow{X_1 \to Y_1} \ldots \xRightarrow{X_m \to Y_m} \mu_m$ such that μ_m has constants in V columns, then $\mu_m[V]$ is in rep(I)[V-total]. Note that sequence $X_1 \to Y_1, \ldots, X_m \to Y_m$ is a derivation of V from R_i. However, the derivation $X_1 \to Y_1, \ldots, X_m \to Y_m$ may be dependent on $\mu[V]$. In the following we show that for a derivation $Z_1 \to W_1, \ldots, Z_s \to W_s$ of V from R_i that depends on only R_i and V, there is a chase process $\mu_0 \xRightarrow{Z_1 \to W_1} \mu_1' \xRightarrow{Z_2 \to W_2} \ldots \xRightarrow{Z_s \to W_s} \mu_s'$. By the consistency of $\underline{R}$, μ_s' agrees with μ_m (and also μ) in V columns. Suppose that $Z_t \to W_t$ is in F_{j_t} for $1 \leqq t \leqq s$. Then μ_s' is in relation $r_i \bowtie r_{j_1}[Z_1W_1] \bowtie \ldots \bowtie r_{j_s}[Z_sW_s]$. Thus a tuple $\mu[V]$, where μ is an extension of a tuple of $aug_U(r_i)$, is in rep(I)[V-total] if and only if $\mu[V]$ is in $(R_i \bowtie R_{j_1}[Z_1W_1] \bowtie \ldots \bowtie R_{j_s}[Z_sW_s])[V](I)$. Note that expression $R_i \bowtie R_{j_1}[Z_1W_1] \bowtie \ldots \bowtie R_{j_s}[Z_sW_s]$ is a sequence of extension joins.

Let $X_1 \to Y_1, \ldots, X_m \to Y_m$ be a derivation of V from R_i. We introduce three operations on derivations as follows.

(1) <u>Addition</u>: For an FD $X_k \to Y_k$ with $1 \leqq k \leqq m$, add an FD $X \to Y$ in

cover($X_k \to Y_k$) to the last of the derivation. Note that the resulting sequence $X_1 \to Y_1$, ..., $X_m \to Y_m$, $X \to Y$ is a derivation of V from R_i.

(2) Deletion: Delete an FD $X_k \to Y_k$ with $1 \leq k \leq m$ from the derivation under the condition that the resulting sequence $X_1 \to Y_1$, ..., $X_{k-1} \to Y_{k-1}$, $X_{k+1} \to Y_{k+1}$, ..., $X_m \to Y_m$ is a derivation of V from R_i.

(3) Exchange: Exchange $X_k \to Y_k$ for $X_{k+1} \to Y_{k+1}$ under the condition that the resulting sequence $X_1 \to Y_1$, ..., $X_{k+1} \to Y_{k+1}$, $X_k \to Y_k$, ..., $X_m \to Y_m$ is a derivation of V from R_i.

We have the following lemma, whose proof is given in Appendix 1.

[Lemma2.7] Let $Z_1 \to W_1$, ..., $Z_s \to W_s$ be a derivation of V from R_i that is obtained by a number of applications of the operations above. For a tuple μ_0 of $\mathrm{aug}_U(r_i)$, if there is a chase process $\mu_0 \xRightarrow{X_1 \to Y_1} \mu_1 \xRightarrow{X_2 \to Y_2} \cdots \xRightarrow{X_m \to Y_m} \mu_m$, then there is a chase process $\mu_0 \xRightarrow{Z_1 \to W_1} \mu_1' \xRightarrow{Z_2 \to W_2} \cdots \xRightarrow{Z_s \to W_s} \mu_s'$. Note that μ_s' agrees with μ_m in V columns by the consistency of $\underline{R}$. []

For an FD $X \to Y$ in F_j, we define proper-cover($X \to Y$) = $\{Z \to W \mid Z \to W$ is in F_j and $ZW \subsetneq XY\}$. A derivation $X_1 \to Y_1$, ..., $X_m \to Y_m$ of V from R_i is said to be minimal if there is no FD $X_k \to Y_k$ with $1 \leq k \leq m$ such that $R_i \to V$ is implied by $\{X_1 \to Y_1$, ..., $X_{k-1} \to Y_{k-1}$, $X_{k+1} \to Y_{k+1}$, ..., $X_m \to Y_m\} \cup$ proper-cover($X_m \to Y_m$). We have the following lemma, whose proof is given in Appendix 1.

[Lemma2.8] Let $Z_1 \to W_1$, ..., $Z_s \to W_s$ be a minimal derivation of V from R_i. Every derivation $X_1 \to Y_1$, ..., $X_m \to Y_m$ of V from R_i can be transformed into the minimal derivation $Z_1 \to W_1$, ..., $Z_s \to W_s$ by a number of applications of the operations: addition, deletion, and exchange. []

By Lemmas 2.6, 2.7, and 2.8, we can construct a relational expression E such that $E(I) = \mathrm{rep}(I)[V\text{-total}]$ for every database instance I of $\underline{R}$ by the following algorithm.

[Algorithm2.2]

(1) For each R_i such that $\mathcal{F}(R_i,F)$ contains V, construct the term E_i as follows. Compute a minimal derivation $Z_1 \rightarrow W_1, \ldots, Z_s \rightarrow W_s$ of V from R_i, where each FD $Z_t \rightarrow W_t$ for $1 \leq t \leq s$ is in F_{j_t}. Let $E_i = R_i \bowtie R_{j_1}[Z_1W_1] \bowtie \ldots \bowtie R_{j_s}[Z_sW_s]$.

(2) Let E be the union of all the terms $E_i[V]$, where E_i is constructed in step (1) above. []

Note that if $\mathcal{F}(R_i,F)$ does not contain V, then no extension of any tuple of $\mathrm{aug}_U(r_i)$ has constants in V columns, and thus there is no tuple $\mu[V]$ of $\mathrm{rep}(I)[V\text{-total}]$ such that μ is an extension of a tuple of $\mathrm{aug}_U(r_i)$.

We estimate the time complexity of Algorithm2.2. The key is how to find a minimal derivation of V from R_i for each R_i such that $\mathcal{F}(R_i,F)$ contains V.

Suppose that in the execution of procedure EXAM(R_i) of Algorithm2.1, the loop of step (2) is repeated p times and let $G = \{X^{(1)} \rightarrow Y^{(1)}, \ldots, X^{(p)} \rightarrow Y^{(p)}\}$. Note that if $k < \ell$, then $X^{(\ell)}Y^{(\ell)} - X^{(k)}Y^{(k)} \neq \emptyset$. A minimal derivation of V from R_i is computed by the following algorithm.

[Algorithm2.3]

```
(1) Let $G' = G$ (= $\{X^{(1)} \rightarrow Y^{(1)}, \ldots, X^{(p)} \rightarrow Y^{(p)}\}$).
(2) for $k = p$ step -1 until 1
    do begin
(3) If $G' - \{X^{(k)} \rightarrow Y^{(k)}\}$ implies $R_i \rightarrow V$, then delete $X^{(k)} \rightarrow Y^{(k)}$ from
$G'$. And otherwise, leave $X^{(k)} \rightarrow Y^{(k)}$ in $G'$.
```

```
    end
  (4) For the final value of $G'$ in step (2) that implies $R_i \to V$, construct
a derivation of V from $R_i$ by reordering the FDs in $G'$. []
```

We show that Algorithm2.3 correctly computes a minimal derivation of V from R_i. First we prove the following lemma.

[Lemma2.9] For a subset V of $\mathcal{F}(R_i,F)$, every minimal derivation of V from R_i consists of only some of the FDs in G.

(Proof) Let $Z_1 \to W_1$, ..., $Z_s \to W_s$ be a minimal derivation of V from R_i. Suppose that $Z_t \to W_t$ is in F_j. By Assumption2.2(b), there is an attribute A in W_t such that $F_j - \{Z_t \to W_t\}$ does not imply $Z_t \to A$. Since the derivation is minimal, there is no FD $Z_k \to W_k$ with $1 \leqq k \leqq t-1$ such that $Z_tW_t \subseteq Z_kW_k$. Thus subsequence $Z_1 \to W_1$, ..., $Z_t \to W_t$ is a derivation of A from R_i such that $Z_t \to W_t$ is irreducible. By inserting some of the FDs in $\cup_{1 \leqq k \leqq t}$proper-cover$(Z_k \to W_k)$, the derivation $Z_1 \to W_1$, ..., $Z_t \to W_t$ can be transformed into a close derivation of A from R_i. Since $F_j - \{Z_t \to W_t\}$ does not imply $Z_t \to A$, the last FD $Z_t \to W_t$ is still irreducible. By Lemma2.5, $Z_t \to W_t$ is selected in step (2-i) of EXAM(R_i). []

Let G'_{final} be the final value of G' in Algorithm2.3. Suppose that $X^{(\ell)} \to Y^{(\ell)}$ is in G'_{final} and that $R_i \to V$ is implied by $(G'_{final} - \{X^{(\ell)} \to Y^{(\ell)}\}) \cup$ proper-cover$(X^{(\ell)} \to Y^{(\ell)})$. By Lemma2.9 we assume without loss of generality that $R_i \to V$ is implied by $(G'_{final} - \{X^{(\ell)} \to Y^{(\ell)}\}) \cup (G \cap$ proper-cover$(X^{(\ell)} \to Y^{(\ell)}))$. Since there is no FD $X^{(k)} \to Y^{(k)}$ with $\ell+1 \leqq k \leqq p$ such that $X^{(k)}Y^{(k)} \subseteq X^{(\ell)}Y^{(\ell)}$, G' contains $G \cap$ proper-cover$(X^{(\ell)} \to Y^{(\ell)})$ when $k = \ell$ in step (2). Thus when $k = \ell$, $G' - \{X^{(\ell)} \to Y^{(\ell)}\}$ would imply $R_i \to V$, and thus $X^{(\ell)} \to Y^{(\ell)}$ must be deleted from

G'. Contradiction. Thus Algorithm2.3 correctly computes a minimal derivation of V from R_i.

The set G is obtained in $O(|F|\|F\|)$ time by EXAM(R_i). Thus a minimal derivation of V from R_i can be computed in $O(p\|G\|) \leq O(|F|\|F\|)$ time by Algorithm2.3. Thus we have the following theorem.

[Theorem2.2] Let $\underline{R} = \{\langle R_1,F_1\rangle, \ldots, \langle R_n,F_n\rangle\}$ be a consistent database scheme over U and let V be a subset of U. We can construct a relational expression E such that E(I) = rep(I)[V-total] for every database instance I of $\underline{R}$ in $O(n|F|\|F\|)$ time. []

2.3.2 Simplification of the relational expression

Let E be the relational expression of Theorem2.2.

<u>Stage1</u>: We consider how to remove a redundant term from E. Suppose that E contains a term $E_i[V]$ and that E_i is of the form $R_i \bowtie R_{j_1}[Z_1W_1] \bowtie \ldots \bowtie R_{j_s}[Z_sW_s]$, where sequence $Z_1 \rightarrow W_1, \ldots, Z_s \rightarrow W_s$ is a minimal derivation of V from R_i. Let $H = \{Z_1 \rightarrow W_1, \ldots, Z_s \rightarrow W_s\}$. Then we have the following lemma, whose proof is given in Appendix 1.

[Lemma2.10] If there is an FD $Z_t \rightarrow W_t$ in H such that $Z_tW_t \rightarrow V$ is implied by cover(H) $\cup$ F_i, then E is equivalent to the expression obtained by removing the term $E_i[V]$ from E. []

By the following algorithm, redundant terms can be removed from E.

```
[Algorithm2.4]
(1) Let E' = E.
(2) for i = 1 until n
  do begin
```

(3) If E' contains term $E_i[V]$ (= $(R_i \bowtie R_{j_1}[Z_1W_1] \bowtie \dots \bowtie R_{j_s}[Z_sW_s])[V]$) and there is an FD $Z_t \to W_t$ with $1 \leq t \leq s$ such that (i) $Z_tW_t \to V$ is implied by cover(H) $\cup$ F_i, where $H = \{Z_1 \to W_1, \dots, Z_s \to W_s\}$, and (ii) E' contains term $E_{j_t}[V]$, then remove the term $E_i[V]$ from E'.

<u>end</u> []

We have the following lemma, whose proof is given in Appendix 1.

[Lemma2.11] Let E'_{final} be the final value of E' by Algorithm2.4. Then E'_{final} contains neither redundant union nor redundant join. []

<u>Stage2</u>: After executing Algorithm2.4, we can remove redundant attributes from each term in E'_{final} as follows.

Suppose that E'_{final} contains a term $E_i[V]$ and that E_i is of the form $R_i \bowtie R_{j_1}[Z_1W_1] \bowtie \dots \bowtie R_{j_s}[Z_sW_s]$. For $1 \leq t \leq s$, let W'_t = $W_t \cap (Z_{t+1} \dots Z_sV)$, and let $E'_i = R_i \bowtie R_{j_1}[Z_1W'_1] \bowtie \dots \bowtie R_{j_s}[Z_sW'_s]$. Then $E_i[V]$ is equivalent to $E'_i[V]$, as explained below.

Let ν_0 be a tuple of $aug_U(r_i)$. Suppose that there is a chase process $\nu_0 \xRightarrow{Z_1 \to W_1} \dots \xRightarrow{Z_s \to W_s} \nu_s$. Then for $0 \leq t \leq s$, tuple ν_t has constants exactly in $R_iW_1 \dots W_t$ columns. But in order to execute $\nu_t \xRightarrow{Z_{t+1} \to W_{t+1}} \dots \xRightarrow{Z_s \to W_s} \nu_s$, the values of ν_t in $(R_iW_1 \dots W_t) \cap (Z_{t+1} \dots Z_sV)$ columns are sufficient.

<u>Stage3</u>: The reason above also implies that the expression $E'_i[V]$ can be transformed into an equivalent expression $E''_i[V]$ without changing the order of join sequence of E'_i in such a way that the projections are executed as early as possible, as follows. Let $P_0 = R_i \cap (Z_1 \dots Z_sV)$ and P_t = $(R_iW'_1 \dots W'_t) \cap (Z_{t+1} \dots Z_sV)$ for $1 \leq t \leq s$. Note that $P_s = V$. Let e_0 = $R_i[P_0]$ and $e_t = (e_{t-1} \bowtie R_{j_t}[Z_tW'_t])[P_t]$ for $1 \leq t \leq s$. Then $E''_i[V]$ is defined as the expression e_s, that is, $E''_i[V]$ is of the form

$((\ldots(R_i[P_0] \bowtie R_{j_1}[Z_1W_1'])[P_1] \bowtie \ldots)[P_{s-1}] \bowtie R_{j_s}[Z_sW_s'])[P_s]$.

We estimate the time for the simplification of E. In Stage1, it can be determined in $O(s\|cover(H) \cup F_i\|) \leqq O(|F|\|F\|)$ time whether there is an FD $Z_t \rightarrow W_t$ in H such that $Z_tW_t \rightarrow V$ is implied by cover(H) $\cup F_i$. Thus Algorithm2.4 can be executed in $O(n|F|\|F\|)$ time. In Stage2, each term $E_i[V]$ in E'_{final} can be transformed into $E_i'[V]$ in $O(s\|H\|)$ time. In Stage3, $E_i'[V]$ can be transformed into $E_i''[V]$ in $O(s\|H\|)$ time. Thus we have the following corollary of Theorem2.2.

[Corollary2.1] The relational expression E of Theorem2.2 can be transformed in $O(n|F|\|F\|)$ time into an equivalent relational expression E' such that (1) E' contains neither redundant union nor redundant join and (2) the projections are executed as early as possible when evaluating $E'(I)$ for a database instance I of $\underline{R}$. []

CHAPTER 3

IMPLICATION PROBLEM FOR FUNCTIONAL AND EMBEDDED MULTIVALUED DEPENDENCIES

In this chapter, we consider implication problem for functional and embedded multivalued dependencies. In Section 3.1, we provide basic definitions and a result from [Sadri and Ullman 80], which is useful for this problem. In Section 3.2, we show some results on this problem. In Section 3.3, we give some extensions of these results, especially an extension of a decidability result of this problem to a class of functional and template dependencies.

3.1 Definitions

Let R be a set of attributes and let V be a subset of R. A <u>multivalued dependency</u> over V [Fagin 77] [Zaniolo 76], abbreviated to MVD, is a statement $X \twoheadrightarrow Y(V)$, where X and Y are subsets of V. A relation r over R is said to be <u>satisfy</u> $X \twoheadrightarrow Y(V)$ if whenever r contains two tuples μ and ν such that $\mu[X] = \nu[X]$, r also contains a tuple τ such that $\tau[XY] = \mu[XY]$ and $\tau[X(V-Y)] = \nu[X(V-Y)]$. It is easy to see that r satisfies $X \twoheadrightarrow Y(V)$ if and only if $r[V] = r[XY] \bowtie r[X(V-Y)]$. If V coincides with R, then $X \twoheadrightarrow Y(V)$ is said to be <u>full</u>. (If V is a proper subset of R, then $X \twoheadrightarrow Y(V)$ is usually called an <u>embedded</u> multivalued dependency.)

Let <R,D> be a relation scheme, where D is a set of FDs and MVDs (possibly containing full MVDs). Let $X \subseteq V \subseteq R$. The <u>dependency basis</u> of X over V with respect to D, denoted $\mathcal{M}(X,V,D)$, is a partition $\{P_1, \ldots, P_\ell\}$ of V such that (1) D implies $X \twoheadrightarrow P_i$ for $1 \leq i \leq \ell$ and (2) D implies an MVD $X \twoheadrightarrow Y(V)$ if and only if the right-hand side Y coincides with a union of some of the blocks P_i. Thus if $\mathcal{M}(X,V,D)$ is known, then it is easy to determine whether a given MVD $X \twoheadrightarrow Y(V)$ is implied by D. If D consists of

FDs and only full MVDs, then there are several polynomial time algorithms for computing $\mathcal{M}(X,R,D)$ for any X [Beeri 80] [Galil 82] [Hagihara et al 79] [Sagiv 80].

Let r be a relation over R consisting of only variables, which is called a tableau over R. FD-rule for each FD in D can be applied to r. Furthermore, MVD-rule for each MVD in D, which is defined below, can be also applied to r.

MVD-rule: An MVD $X \twoheadrightarrow Y(V)$ in D has associated rule as follows. Suppose that r does not satisfy $X \twoheadrightarrow Y(V)$. Then there are tuples μ and ν of r and τ not of r such that $\tau[X] = \mu[X] = \nu[X]$, $\tau[XY] = \mu[XY]$ ($\neq \nu[XY]$), and $\tau[V - XY] = \nu[V - XY]$ ($\neq \mu[V - XY]$). MVD-rule for $X \twoheadrightarrow Y(V)$ adds to r a tuple τ' that agrees with τ in V columns and has distinct variables (that do not appear in r) in R - V columns. Each variable of τ' in R - V columns is said to be unique.

It is known that if a chase process of r under D terminates (that is, a tableau satisfying D is obtained by the chase process), then any chase process of r under D always terminates and the resulting tableau is unique up to renaming of variables [Maier et al 79]. The resulting tableau is called the chase of r under D and denoted chase(r,D). Note that if D consists of FDs and only full MVDs, then any chase process of r under D always terminates [Maier et al 79]. However, if D contains two or more MVDs, then there may be an infinite chase process of r under D, that is, we may not obtain a finite tableau satisfying D by any chase process of r under D.

In the following we often consider a tableau consisting of two tuples that agree exactly in X columns, which is called an X-agreed tableau. The following lemma is obtained from Theorems 1 and 4 of [Sadri and Ullman 80].

[Lemma3.1] Let <R,D> be a relation scheme and let $r = \{\mu, \nu\}$ be an

X-agreed tableau over R.

(1) D implies an MVD $X \twoheadrightarrow Y(V)$ if and only if we obtain a tableau containing a tuple τ such that $\tau[X] = \mu[X] = \nu[X]$, $\tau[Y] = \mu[Y]$, and $\tau[V - XY] = \nu[V - XY]$ by a chase process of r under D. The tuple τ is said to <u>witness</u> $X \twoheadrightarrow Y(V)$.

(2) D implies an FD $X \rightarrow Y$ if and only if we obtain a tableau in which $\mu[Y]$ and $\nu[Y]$ are identified by a chase process of r under D. [ ]

Lemma3.1 implies that if a chase process of r under D terminates, then chase(r,D) contains all the information about FDs and MVDs with the left-hand side X that are implied by D. We have the following two corollaries of Lemma3.1.

[Corollary3.1] Let $\langle R,D' \rangle$ be another relation scheme. If chase(r,D) and chase(r,D′) are the same (up to renaming of variables), then $\mathcal{M}(X,V,D) = \mathcal{M}(X,V,D')$. [ ]

[Corollary3.2] Let P be a block in $\mathcal{M}(X,V,D)$ and let τ be a tuple of chase(r,D). If for all attribute A in V, τ agrees with either μ or ν in A column, then τ agrees with either μ or ν in V columns. [ ]

3.2 Implication Problem

Let R be a set of attributes and let $U_n \subsetneq \ldots \subsetneq U_1 \subsetneq U_0 \subseteq R$. In this section, we consider a relation scheme $\langle R, F \cup M \cup M_1 \cup \ldots \cup M_n \rangle$ such that

(1) F is a set of FDs $Z \rightarrow W$ satisfying $Z \subseteq U_0$ or $W \cap U_0 = \emptyset$,

(2) M is a set of MVDs $Z \twoheadrightarrow W(V)$ satisfying at least one of $Z \subseteq U_0 \subseteq V$, $W \cap U_0 = \emptyset$, and $(V - ZW) \cap U_0 = \emptyset$, and

(3) each M_i for $1 \leq i \leq n$ is a set of MVDs over U_i.

Note that M may contain full MVDs. For simplicity, we denote $F \cup M \cup M_1 \cup \ldots \cup M_n$ by D_n.

3.2.1 A decidability result

For a relation scheme <R,D> and a subset U of R, we define

$$D[U] = \{Z \rightarrow W \cap U \mid Z \rightarrow W \text{ is in } D \text{ and } Z \subseteq U\}$$
$$\cup \{Z \twoheadrightarrow W \cap U(V \cap U) \mid Z \twoheadrightarrow W(V) \text{ is in } D \text{ and } Z \subseteq U\}.$$

Note that if a relation r over R satisfies D, then r[U] satisfies D[U], and thus D implies D[U]. D[U] can be considered as the "projection" of D onto U. Then we have the following lemma.

[Lemma3.2] Let <R,D> be a relation scheme. Suppose that a subset U of R satisfies

(1) $Z \subseteq U$ or $W \cap U = \emptyset$ for all $Z \rightarrow W$ in D, and

(2) at least one of $Z \subseteq U$, $W \cap U = \emptyset$, and $(V - ZW) \cap U = \emptyset$ for all $Z \twoheadrightarrow W(V)$ in D.

Then an MVD $X \twoheadrightarrow Y(V)$ (or an FD $X \rightarrow Y$) over a subset V of U is implied by D if and only if it is implied by D[U].

(Proof) Since D implies D[U], the "if" part is trivial. For the "only if" part, suppose that $X \twoheadrightarrow Y(V)$ is not implied by D[U]. Then there is a relation r over U that satisfies D[U] but does not satisfy $X \twoheadrightarrow Y(V)$. The relation r can be extended to a relation over R by adding columns for the attributes in R - U such that all tuples of the new relation agree in R - U columns. Since the new relation satisfies FD $\emptyset \rightarrow R - U$, it is easy to show that the new relation satisfies D but does not satisfy $X \twoheadrightarrow Y(V)$. Thus $X \twoheadrightarrow Y(V)$ is not implied by D. The same argument applies also to FDs. []

Consider the relation scheme $<R,D_n>$ defined above. Since U_0 satisfies condition of Lemma3.2, it follows that an MVD $X \twoheadrightarrow Y(V)$ (or an FD $X \rightarrow Y$) over a subset V of U_0 is implied by D_n if and only if it is implied by $D_n[U_0]$. For the relation scheme $<R,D_n[U_0]>$, we have the following lemma.

[Lemma3.3] Let r_0 be a tableau over R. Then any chase process of r_0 under $D_n[U_0]$ finally terminates.

(Proof) Suppose that there is an infinite chase process of r_0 under $D_n[U_0]$. Then it can be considered that an "infinite" tableau r is obtained by the chase process. There is at least one attribute A in U_0 such that r has infinite distinct variables in A column. However, we show that any tableau r obtained by any chase process of r_0 under $D_n[U_0]$ has a finite number of variables in all columns by induction on the order U_n, ..., U_1, U_0, R. Thus Lemma3.3 will follow. For convenience, let $U_{-1} = R$ and let $M_0 = M[U_0]$. Note that M_0 is a set of MVDs over U_0.

Basis: Since $U_n \subseteq V$ for all $Z \twoheadrightarrow W(V)$ in $D_n[U_0]$, r has no unique variable in U_n columns. Thus for all attribute A in U_n, the number of distinct variables of r in A column is at most that of r_0 in A column.

Induction: Suppose that for all attribute A in U_i, r has at most p_i variables in A column. Since $U_i \subseteq U_{i-1}$, it suffices to show that for all attribute A in $U_{i-1} - U_i$, r has finite variables in A column.

Since $U_n \subseteq \ldots \subseteq U_0$, it follows that $V \subseteq U_i$ for all $Z \twoheadrightarrow W(V)$ in $M_n \cup \ldots \cup M_i$ and that $U_i \subseteq V$ for all $Z \twoheadrightarrow W(V)$ in $M_{i-1} \cup \ldots \cup M_0$. Thus each unique variable of r in $U_{i-1} - U_i$ columns has been added by MVD-rule for an MVD in $M_n \cup \ldots \cup M_i$. Since the number of distinct tuples of $r[U_i]$ is at most $p_i^{|U_i|}$ by the induction hypothesis, the number of applications of MVD-rules for the MVDs in $M_n \cup \ldots \cup M_i$ is also at most $p_i^{|U_i|}$. Thus for all attribute A in $U_{i-1} - U_i$, the number of unique variables of r in A column is at most $p_i^{|U_i|}$. []

By Lemmas 3.1, 3.2, and 3.3, we have the following theorem.

[Theorem3.1] Consider the relation scheme $<R,D_n>$. It is decidable whether a given MVD $X \twoheadrightarrow Y(V)$ (or a given FD $X \rightarrow Y$) over a subset V of U_0 is implied by D_n. []

By Theorem3.1, $\mathcal{M}(X,V,D_n)$ for $X \subseteq V \subseteq U_0$ can be, in principle, obtained but the theorem suggests no efficient procedure for computing $\mathcal{M}(X,V,D_n)$. However, if $X \subseteq U_n$, then $\mathcal{M}(X,V,D_n)$ can be computed efficiently. In the following we will present a procedure for computing $\mathcal{M}(X,V,D_n)$ in two cases: (1) $X \subseteq V \subseteq U_n$ and (2) $X \subseteq U_n$ and $X \subseteq V \subseteq U_0$.

3.2.2 The case where $X \subseteq V \subseteq U_n$

For a partition Π of U and a subset V of U, we define $\Pi[V] = \{B \cap V \mid B$ is in Π and $B \cap V \neq \emptyset\}$. Note that $\Pi[V]$ is a partition of V. Then we have the following lemma.

[Lemma3.4] Let $<R,D>$ be a relation scheme. Suppose that a subset U of R satisfies the condition that $U \subseteq W$ for all $Y \twoheadrightarrow Z(W)$ in D. Then $\mathcal{M}(X,V,D) = \mathcal{M}(X,VU,D)[V]$ for any $X \subseteq V \subseteq R$.

(Proof) Let P and Q be blocks in $\mathcal{M}(X,V,D)$ and $\mathcal{M}(X,VU,D)$, respectively, such that $P \subseteq Q$. It suffices to show that $P = Q \cap V$. Let $r = \{\mu, \nu\}$ be an X-agreed tableau over R. Consider a chase process of r under D and let τ be a tuple that witness $X \twoheadrightarrow P(V)$. For all attribute A in U, τ agrees with either μ or ν in A column, since no unique variable is introduced in U columns by the assumption. Thus τ actually witness $X \twoheadrightarrow PS(VU)$, where $S \subseteq U - V$. But $Q \subseteq PS$, and thus $Q \cap V = P$. []

For the relation scheme $\langle R, D_n[U_0]\rangle$, since U_n satisfies condition of Lemma3.4, in order to obtain $\mathcal{M}(X,V,D_n[U_0])$ for $X \subseteq V \subseteq U_0$, it suffices to obtain $\mathcal{M}(X,VU_n,D_n[U_0])$. The following lemma shows that $\mathcal{M}(X,U_n,D_n[U_0])$ can be computed using a technique for full MVDs. Thus $\mathcal{M}(X,U_n,D_n[U_0])$ can be computed efficiently. For an attribute A in U_i and M_i with $1 \leq i \leq n$, we define

$$f(A,M_i) = \{Z \twoheadrightarrow W(U_0) \mid Z \twoheadrightarrow W(U_i) \text{ is in } M_i \text{ and } A \text{ is not in } W\}$$
$$\cup \{Z \twoheadrightarrow U_i - W(U_0) \mid Z \twoheadrightarrow W(U_i) \text{ is in } M_i \text{ and } A \text{ is in } W\}.$$

Note after the definition of $f(A,M_i)$ that $f(A,M_i)$ implies M_i.

[Lemma3.5] Consider the relation scheme $\langle R,D_n\rangle$. Let A be in U_n and let P be a subset of U_n that contains A. Then P is in $\mathcal{M}(X,U_n,D_n[U_0])$ if and only if P is in $\mathcal{M}(X,U_n,F[U_0] \cup M[U_0] \cup f(A,M_1) \cup \ldots \cup f(A,M_n))$.

(Proof) Induction on the number n.

Basis: If n = 0, then this lemma holds trivially.

Induction: Let P be a block in $\mathcal{M}(X,U_n,D_n[U_0])$ that contains A. Let $r = \{\mu, \nu\}$ be a $(U_n - P)$-agreed tableau over U_0. The fact that P is in $\mathcal{M}(X,U_n,D_n[U_0])$ implies that P is also in $\mathcal{M}(U_n - P,U_n,D_n[U_0])$, and thus it follows from Corollary3.2 that every tuple of $\text{chase}(r,D_n[U_0])$ agrees with either μ or ν in W_n columns. Thus $\text{chase}(r,D_n[U_0])$ satisfies $f(A,M_n)$, and so P is in $\mathcal{M}(U_n - P,U_n,D_{n-1}[U_0] \cup f(A,M_n))$, where $D_{n-1}[U_0] = D_n[U_0] - M_n$. Since $D_n[U_0]$ implies $X \twoheadrightarrow P(U_n)$, $D_{n-1}[U_0] \cup f(A,M_n)$ also implies $X \twoheadrightarrow P(U_n)$. Thus the fact that P is in $\mathcal{M}(U_n - P,U_n,D_{n-1}[U_0] \cup f(A,M_n))$ implies that P is also in $\mathcal{M}(X,U_n,D_{n-1}[U_0] \cup f(A,M_n))$. For the relation scheme $\langle R,D_{n-1}[U_0] \cup f(A,M_n)\rangle$, since U_{n-1} satisifes condition of Lemma3.4, it follows that $\mathcal{M}(X,U_n,D_{n-1}[U_0] \cup f(A,M_n)) = \mathcal{M}(X,U_{n-1},D_{n-1}[U_0] \cup f(A,M_n))[U_n]$. Thus there is a block Q in

$\mathcal{M}(X,U_{n-1},D_{n-1}[U_0] \cup f(A,M_n))$ such that $P = Q \cap U_n$. By the induction hypothesis, Q is in $\mathcal{M}(X,U_{n-1},(F[U_0] \cup M[U_0] \cup f(A,M_1) \cup \ldots \cup f(A,M_{n-1})) \cup f(A,M_n))$. Thus P is in $\mathcal{M}(X,U_n,F[U_0] \cup M[U_0] \cup f(A,M_1) \cup \ldots \cup f(A,M_n))$ (= $\mathcal{M}(X,U_{n-1},F[U_0] \cup M[U_0] \cup f(A,M_1) \cup \ldots \cup f(A,M_n))[U_n]$). []

For simplicity, let $D_A = F[U_0] \cup M[U_0] \cup f(A,M_1) \cup \ldots \cup f(A,M_n)$. Consider the relation scheme $\langle U_0,D_A \rangle$. Since D_A consists of FDs and only full MVDs, $\mathcal{M}(X,U_0,D_A)$ can be computed by a known algorithm for full MVDs. Furthermore since U_0 satisfies condition of Lemma3.4, it follows that $\mathcal{M}(X,U_n,D_A) = \mathcal{M}(X,U_0,D_A)[U_n]$. Thus we have the following algorithm.

[Algorithm3.1]

input: $\langle R,D_n \rangle$, and X and V such that $X \subseteq V \subseteq U_n$.

output: $\pi = \mathcal{M}(X,V,D_n) - \{\{A\} \mid A \text{ is in } X\}$.

method:

```
procedure FIND(A)
(A is an attribute in $U_n - X$. This procedure computes a subset P of $U_n$,
such that A is in P and P is in $\mathcal{M}(X,U_n,D_n)$.)
begin
   (1) Make QUEUE empty.
   (2) Let $P = U_0 - X$.
   (3) For each dependency in $D_n[U_0]$, if its left-hand side is disjoint from
P, then put the dependency on QUEUE.
   (4) while QUEUE is not empty
         do begin
   (4-i) Remove a dependency from QUEUE. There are two cases to be
considered.
      Case1: Suppose that the dependency is an FD $Z \rightarrow W$. If A is in W, then
```

```
return $\{A\}$ and terminate FIND(A). Otherwise, let $P = P - W$.
        Case2: Suppose that the dependency is an MVD $Z \twoheadrightarrow W(U_i)$ with
    $0 \leq i \leq n$. If A is in W, then let $P = P - (U_i - W)$. Otherwise, let $P = P - W$.
    (4-ii) For each dependency in $D_n[U_0]$, if its left-hand side is disjoint
    from P and the dependency has not been on QUEUE, then put the dependency on
    QUEUE.
        end while
    (5) Return $P \cap U_n$.
end FIND.

begin (main procedure)
    (6) Compute $D_n[U_0]$ from $D_n$.
    (7) Make $\pi$ empty.
    (8) Let $Q = W_n - X$.
    (9) while Q is not empty
        do begin
    (9-i) Select an attribute A in P.
    (9-ii) Execute FIND(A) and let the result P be a new block in $\pi$.
    (9-iii) Let $Q = Q - P$.
        end while
end main procedure. []
```

After Algorithm3.1 terminates, the value of π coincides with $\mathcal{M}(X,U_n,D_n) - \{\{A\} \mid A \text{ is in } X\}$. Note that each attribute A in X itself constitutes a block in $\mathcal{M}(X,U_n,D_n)$. The procedure FIND(A) of Algorithm3.1 is essentially an extended version of the procedure "FIND(B)" of [Sagiv 80]. The differences between them are as follows.

(1) In Sagiv's procedure, only full MVDs (and FDs) are considered. This

corresponds to the case where every MVD is of the form $Z \twoheadrightarrow W(U_0)$.

(2) Sagiv's procedure does not consider the projection. Thus, Sagiv's one has return statement "Return P" in step (5) of Algorithm3.1.

The reason we adopt Sagiv's procedure is that given an attribute A, we do not have to construct the whole dependency basis $\mathcal{M}(X,U_0,D_A)$ but we need only a block in $\mathcal{M}(X,U_0,D_A)[U_n]$ that contains A.

We estimate the running time of Algorithm3.1. Let s be the number of blocks in the output π and let k be the number of dependencies in $D_n[U_0]$. Then Algorithm3.1 terminates in $O(\|D_n[U_0]\|.\min\{k,\log_2 s\})$ time [Galil 82]. If we assume that each attribute in R is represented by an integer, then $D_n[U_0]$ can be computed from D_n in $O(\|D_n\|)$ time. And $\mathcal{M}(X,U_n,D_n[U_0])[V]$ can be computed from $\mathcal{M}(X,U_n,D_n[U_0])$ in $O(|U_n|) \leqq O(\|D_n[U_0]\|)$ time. Thus we have the following theorem.

[Theorem3.2] Consider the relation scheme $\langle R,D_n \rangle$ and let $X \subseteq V \subseteq U_n$. Then $\mathcal{M}(X,V,D_n)$ can be computed in $O(\|D_n[U_0]\|.\min\{k,\log_2 s\} + \|D_n\|)$ time, where s is the number of blocks in $\mathcal{M}(X,U_n,D_n) - \{\{A\} \mid A \text{ is in } X\}$ and k is the number of dependencies in $D_n[U_0]$. [ ]

Consider the implication problem. In order to determine whether a given MVD $X \twoheadrightarrow Y(V)$ over a subset V of U_n is implied by D_n, we need only blocks in $\mathcal{M}(X,V,D_n)$ that intersect Y. If there is a block P in $\mathcal{M}(X,V,D_n)$ such that $P \cap Y \neq \emptyset$ and $P - Y \neq \emptyset$, then $X \twoheadrightarrow Y(V)$ is not implied by D_n, and otherwise it is implied by D_n. Thus it can be determined in $O(\|D_n[U_0]\|.\min\{k,\log_2 s'\} + \|D_n\|)$ time whether $X \twoheadrightarrow Y(V)$ is implied by D_n, where s' is the number of blocks in $\mathcal{M}(X,V,D_n) - \{\{A\} \mid A \text{ is in } X\}$ that intersect Y (cf. [Galil 82]).

3.2.3 The case where $X \subseteq U_n$ and $X \subseteq V \subseteq U_0$

In the following we often write $X \twoheadrightarrow P_1|\ldots|P_s$ for a set $\{X \twoheadrightarrow P_1(U), \ldots, X \twoheadrightarrow P_s(U)\}$ of MVDs such that $\{X, P_1, \ldots, P_s\}$ is a partition of U. We have the following lemma, whose proof is given in Appendix 2.

[Lemma3.6] Let <R,D> be a relation scheme and let $X \subseteq U \subseteq V \subseteq R$. Suppose that D implies $X \twoheadrightarrow P_1|\ldots|P_s$, where $\{X, P_1, \ldots, P_s\}$ is a partition of U. Then a subset Q of V is in $\mathcal{M}(X,V,D)$ if and only if Q is a minimal set (in the sense of set inclusion: $\subseteq$) such that for all i with $1 \leqq i \leqq s$, Q is a union of some of the blocks in $\mathcal{M}(U - P_i,V,D)$. []

Lemma3.6 shows that $\mathcal{M}(X,V,D)$ can be obtained from $\mathcal{M}(U - P_1,V,D), \ldots, \mathcal{M}(U - P_s,V,D)$ by the following algorithm.

[Algorithm3.2]

```
input: $\Pi_1 = \mathcal{M}(U - P_1,V,D), \ldots, \Pi_s = \mathcal{M}(U - P_s,V,D)$.
comment: D implies $X \twoheadrightarrow P_1|\ldots|P_s$, where $\{X, P_1, \ldots, P_s\}$ is a partition of U.
output: $\Pi = \mathcal{M}(X,V,D)$.
method:
procedure MIN(A)
(This procedure computes a minimal set Q such that (i) Q contains A and (ii) for all i with $1 \leqq i \leqq s$, Q is a union of some of the blocks in $\Pi_i$.)
begin
  (1) Let Q = {A}.
  (2) while there is a block in $\Pi_1 \cup \ldots \cup \Pi_s$ that intersects Q
        do begin
  (2-i) Select and delete all blocks from $\Pi_1 \cup \ldots \cup \Pi_s$ that intersects Q.
  (2-ii) Let S be the union of all blocks selected in step (2-i).
```

```
(2-iii) Let $Q = Q \cup S$.
        end while
    (3) Return Q.
end MIN.

begin (main procedure)
    (4) Make $\Pi$ empty.
    (5) Let $T = V$.
    (6) while T is not empty
        do begin
    (6-i) Select an attribute A in T.
   (6-ii) Execute MIN(A) and let the result Q be a new block in $\Pi$.
  (6-iii) Let $T = T - Q$.
        end while
end main procedure. []
```

Since $\Pi_1, \ldots, \Pi_s$ are partitions of the same set V and each block in $\Pi_1 \cup \ldots \cup \Pi_s$ is used at most once, Algorithm3.2 terminates in $O(s.|V|)$ time.

The following lemma implies that $\mathcal{M}(X,V,D_n[U_0])$ for $X \subseteq U_n$ and $X \subseteq V \subseteq U_0$ can be computed by a recursive procedure.

[Lemma3.7] Consider the relation scheme $\langle R,D_n \rangle$. Let $X \subseteq U_n$ and $X \subseteq V \subseteq U_0$. Let P be a block in $\mathcal{M}(X,U_n,D_n[U_0]) - \{\{A\} \mid A \text{ is in } X\}$. Then it holds that $\mathcal{M}(U_n - P,VU_n,D_n[U_0]) = \mathcal{M}(U_n - P,VU_n,D_{n-1}[U_0])$, where $D_{n-1}[U_0] = D_n[U_0] - M_n$.

(Proof) Let $r = \{\mu, \nu\}$ be a $(U_n - P)$-agreed tableau. Since $D_{n-1}[U_0] \subseteq D_n[U_0]$ and the fact that P is in $\mathcal{M}(X,U_n,D_n[U_0])$ implies that P is also in $\mathcal{M}(U_n - P, U_n, D_n[U_0])$, it follows from Corollary3.2 that every tuple of chase$(r, D_{n-1}[U_0])$ agrees with either μ or ν in U_n columns. Let $Z \twoheadrightarrow W(U_n)$ be in M_n. If $P \cap Z = \emptyset$, then either $P \subseteq W$ or $P \cap W = \emptyset$ (otherwise P would not be in $\mathcal{M}(X, U_n, D_n[U_0])$), and thus chase$(r, D_{n-1}[U_0])$ satisfies $Z \twoheadrightarrow W(U_n)$. Thus chase$(r, D_{n-1}[U_0])$ = chase$(r, D_n[U_0])$ and it follows from Corollary3.1 that $\mathcal{M}(U_n - P, VU_n, D_n[U_0]) \doteq \mathcal{M}(U_n - P, VU_n, D_{n-1}[U_0])$. []

For convenience, we consider $D_0 = F \cup M$. Let $\{P_1, \ldots, P_s\} = \mathcal{M}(X, U_n, D_n[U_0]) - \{\{A\} \mid A \text{ is in } X\}$. Then $\mathcal{M}(X, VU_n, D_n[U_0])$ is obtained from $\mathcal{M}(U_n - P_1, VU_n, D_{n-1}[U_0])$, ..., $\mathcal{M}(U_n - P_s, VU_n, D_{n-1}[U_0])$ by Lemma3.7 and Algorithm3.2. If $n = 0$, then it follows from Lemma3.4 that $\mathcal{M}(S, T, D_0[U_0]) = \mathcal{M}(S, U_0, D_0[U_0])[T]$ for any $S \subseteq T \subseteq U_0$. Furthermore $\mathcal{M}(S, U_0, D_0[U_0])$ can be computed by a known algorithm for full MVDs. Thus we have the following algorithm.

[Algorithm3.3]

input: $\langle R, D_n \rangle$, and X and V such that $X \subseteq U_n$ and $X \subseteq V \subseteq U_0$.

output: $\mathcal{M}(X, V, D_n[U_0])$ (= $\mathcal{M}(X, V, D_n)$ by Lemma3.2).

method:

```
procedure COMPUTE-$\mathcal{M}(S,T,D_i[U_0])$
(This procedure computes $\mathcal{M}(S,T,D_i[U_0])$ for $S \subseteq U_i$ and $S \subseteq T \subseteq U_0$.)
begin
  (1) If $i = 0$, then compute $\mathcal{M}(S,U_0,D_0[U_0])$ by a known algorithm for full
  MVDs and return $\mathcal{M}(S,U_0,D_0[U_0])[T]$.
  (Consider the case where $i \geq 1$.)
  (2) Compute $\mathcal{M}(S,U_i,D_i[U_0]) - \{\{A\} \mid A$ is in $S\}$ by applying Algorithm3.1
  to $\langle R,D_i[U_0] \rangle$ and S. Let $\{P_1, \ldots, P_s\} = \mathcal{M}(S,U_i,D_i[U_0]) - \{\{A\} \mid A$ is in
  $S\}$.
```

(3) For each P_j with $1 \leqq j \leqq s$, execute COMPUTE-$\mathcal{M}(U_i - P_j, TU_i, D_{i-1}[U_0])$ and assign the result to Π_j. (By Lemma3.7, the value of Π_j coincides with $\mathcal{M}(U_i - P_j, TU_i, D_i[U_0])$.)

(4) Compute $\mathcal{M}(S, TU_i, D_i[U_0])$ by applying Algorithm3.2 to $\Pi_1, \ldots, \Pi_s$.

(5) Return $\mathcal{M}(S, TU_i, D_i[U_0])[T]$. (It follows from Lemma3.4 that $\mathcal{M}(S, TU_i, D_i[U_0])[T] = \mathcal{M}(S, T, D_i[U_0])$.)

<u>end</u> COMPUTE-$\mathcal{M}$.

<u>begin</u> (main procedure)

(6) Compute $D_n[U_0]$ from D_n.

(7) Execute COMPUTE-$\mathcal{M}(X, V, D_n[U_0])$.

<u>end</u> main procedure. []

We estimate the running time of Algorithm3.3. Let TIME(n) be the time for step (7) of Algorithm3.3 and let TIME(i) for $0 \leqq i \leqq n-1$ be the time for executing a recursive call in step (3) of Algorithm3.3 to the procedure call COMPUTE-$\mathcal{M}$ with $D_i[U_0]$ as the third argument. Let s_i be the maximum number of blocks obtained in step (2) for U_i, $D_i[U_0]$, and some subset S of U_i. It follows from the following four facts that TIME(i) = $O(s_i . \| D_i[U_0] \|)$ + s_i.TIME(i-1) + $O(s_i . |U_0|)$ for $i \geqq 1$.

(a) By Theorem3.2, step (2) can be executed in $O(s_i . \| D_i[U_0] \|)$ time. Note that $\min\{k, \log_2 s\} \leqq s_i$.

(b) In step (3), COMPUTE-$\mathcal{M}$ is called s_i times and each call needs TIME(i-1) time. Thus step (3) can be executed in s_i.TIME(i-1) time.

(c) Since each Π_j is a partition of TU_i, step (4) can be executed in $O(s_i . |TU_i|) \leqq O(s_i . |U_0|)$ time by Algorithm3.2.

(d) Step (5) can be executed in $O(|T|) \leqq O(|U_0|)$ time.

It is clear that $s_n \leqq |U_n - X|$, because $S = X$ when step (2) is executed for U_n and $D_n[U_0]$. Consider a call COMPUTE-$\mathcal{M}(U_i - P_j, TU_i, D_{i-1}[U_0])$ in step

(3), where P_j is in $\mathcal{M}(S,U_i,D_i[U_0]) - \{\{A\} \mid A \text{ is in } S\}$. This call results in the computation of $\mathcal{M}(U_i - P_j,U_{i-1},D_{i-1}[U_0]) - \{\{A\} \mid A \text{ is in } U_i - P_j\}$ in step (2). P_j must be contained in one block of $\mathcal{M}(U_i - P_j,U_{i-1},D_{i-1}[U_0])$ (otherwise P_j would not be in $\mathcal{M}(S,U_i,D_i[U_0])$). Thus the number of blocks in $\mathcal{M}(U_i - P_j,U_{i-1},D_{i-1}[U_0]) - \{\{A\} \mid A \text{ is in } U_i - P_j\}$ is at most $|U_{i-1} - U_i| + 1$, that is, $s_i \leqq |U_{i-1} - U_i| + 1$ for $1 \leqq i \leqq n-1$.

If $i = 0$, then $\mathcal{M}(Y,U_0,D_0[U_0])$ for any $Y \subseteq U_0$ can be executed in $O(\|D_0[U_0]\|.\min\{k,\log_2 s\})$ time by the algorithm of [Galil 82], where s is the number of blocks in $\mathcal{M}(Y,U_0,D_0[U_0])$ and k is the number of dependencies in $D_0[U_0]$. Thus $\mathrm{TIME}(0) \leqq O(\|D_0[U_0]\|.\min\{k,\log_2 s\})$.

By the discussions above, we have TIME(n) = $O((\|D_n[U_0]\| + \mathrm{TIME}(0) + |U_0|).|U_n - X|.\prod_{i=1}^{n-1}(|U_i - U_{i+1}| + 1))$. Since $D_n[U_0]$ is computed from D_n in $O(\|D_n\|)$ time and $|U_0| \leqq \|D_n[U_0]\|$, we have the following theorem.

[Theorem3.3] Consider the relation scheme $\langle R,D_n\rangle$. Let $X \subseteq U_n$ and $X \subseteq V \subseteq U_0$. Then $\mathcal{M}(X,V,D_n)$ can be computed in $O((\|D_n[U_0]\| + \mathrm{TIME}(0)).|U_n - X|.\prod_{i=1}^{n-1}(|U_i - U_{i+1}| + 1) + \|D_n\|)$ time, where TIME(0) is the time for computing $\mathcal{M}(Y,U_0,D_0[U_0])$ for a subset Y of U_0. []

3.2.4 Treatments of functional dependencies

The following lemma can be proved in the same way as Lemma3.5.

[Lemma3.8] Consider the relation scheme $\langle R,D_n\rangle$. Let X be a subset of U_n and let A be an attribute in U_n. Then $X \to A$ is implied by $D_n[U_0]$ if and only if it is implied by $F[U_0] \cup M[U_0] \cup f(A,M_1) \cup \ldots \cup f(A,M_n)$. []

By the proof of Theorem11 of [Sagiv 80], $X \to A$ is implied by $F[U_0] \cup M[U_0] \cup f(A,M_1) \cup \ldots \cup f(A,M_n)$ if and only if a call FIND(A) of

Algorithm3.1 returns {A} in Case1 of step (2-i). Let X and Y be subsets of U_n. Then testing whether $X \rightarrow Y$ is implied by D_n can be done in $O(|Y - X|.\|D_n[U_0]\| + \|D_n\|)$ time by checking whether for each attribute A in $Y - X$, a call FIND(A) returns {A} in Case1 of step (2-i).

The following lemma follows from Lemma3.6 and an inference rule for FDs and MVDs [Beeri et al 77].

[Lemma3.9] Let <R,D> be a relation scheme and let $X \subseteq U \subseteq R$. Suppose that D implies $X \twoheadrightarrow P_1|\ldots|P_s$, where $\{X,P_1, \ldots, P_s\}$ is a partition of U. Then D implies $X \rightarrow A$ if and only if D implies $\{U - P_1 \rightarrow A, \ldots, U - P_s \rightarrow A\}$. []

The follwing lemma can be proved in the same way as Lemma3.7.

[Lemma3.10] Consider the relation scheme $<R,D_n>$. Let X U_n and let P be a block in $\mathcal{M}(X,U_n,D_n[U_0]) - \{\{A\} \mid A \text{ is in } X\}$. For an attribute A in U_0, FD $U_n - P \rightarrow A$ is implied by $D_n[U_0]$ if and only if it is implied by $D_{n-1}[U_0]$ []

Let $X \subseteq U_n$. By Lemmas 3.2, 3.9, and 3.10, $\mathcal{F}(X,D_n) \cap U_0$ can be computed in the time for computing $\mathcal{M}(X,U_0,D_n)$ by modifying Algorithm3.3, as shown below. Thus if $X \subseteq U_n$ and $Y \subseteq U_0$, then it can be determined in $O((\|D_n[U_0]\| + TIME(0)).|U_n - X|.\prod_{i=1}^{n-1}(|U_i - U_{i+1}| + 1) + \|D_n\|)$ time whether $X \rightarrow Y$ is implied by D_n.

[Algorithm3.4]

input: $<R,D_n>$ and $X \subseteq U_n$.

output: $\mathcal{F}(X,D_n[U_0])$ (= $\mathcal{F}(X,D_n) \cap U_0$ by Lemma3.2).

method:

```
procedure COMPUTE-$\mathcal{F}(S,D_i[U_0])$
(This procedure computes $\mathcal{F}(S,D_i[U_0])$ for $S \subseteq U_i$.)
begin
    (1) If $i = 0$, then compute $\mathcal{F}(S,D_0[U_0])$ by a known algorithm for FDs and full MVDs and return $\mathcal{F}(S,D_0[U_0])$.
(Consider the case where $i \geq 1$.)
    (2) Compute $\mathcal{M}(S,U_i,D_i[U_0]) - \{\{A\} \mid A$ is in $S\}$ by applying Algorithm3.1 to $\langle R,D_i[U_0]\rangle$ and $S$. Let $\{P_1, \ldots, P_s\} = \mathcal{M}(S,U_i,D_i[U_0]) - \{\{A\} \mid A$ is in $S\}$.
    (3) For each $P_j$ with $1 \leq j \leq s$, execute COMPUTE-$\mathcal{F}(U_i - P_j,D_{i-1}[U_0])$ and assign the result to $Q_j$. (By Lemma3.10, $Q_j$ coincides with $\mathcal{F}(U_i - P_j,D_i[U_0])$.)
    (4) Return $Q_1 \cap \ldots \cap Q_s$. (By Lemma3.9, $Q_1 \cap \ldots \cap Q_s$ coincides with $(S,D_i[U_0])$.)
end COMPUTE-$\mathcal{F}$.

begin (main procedure)
    (5) Compute $D_n[U_0]$ from $D_n$.
    (6) Execute COMPUTE-$\mathcal{F}(X,D_n[U_0])$.
end main procedure. []
```

3.3 Some Extensions

3.3.1 Extensions of Theorems 3.2 and 3.3

Consider the relation scheme $\langle R,D_n\rangle$ and let $U_{i+1} \subseteq X \subseteq U_i$. In order to obtain $\mathcal{M}(X,V,D_n)$, every MVD $Z \twoheadrightarrow W(U)$ in $D_n - D_i$ ($= M_{i+1} \cup \ldots \cup M_n$) can be ignored, because $U \subseteq U_{i+1} \subseteq X$. That is, it follows that $\mathcal{M}(X,V,D_n) = \mathcal{M}(X,V,D_i)$. Consequently, we have the following two facts as corollaries of Theorems 3.2 and 3.3, respectively.

(1) For $0 \leq i \leq n-1$, if $U_{i+1} \subseteq X \subseteq V \subseteq U_i$, then $\mathcal{M}(X,V,D_n)$ can be computed in $O(\|D_i[U_0]\| . \min\{k, \log_2 s\} + \|D_i\|)$ time, where s is the number of blocks in $\mathcal{M}(X,U_i,D_n) - \{\{A\} \mid A \text{ is in } X\}$ and k is the number of dependencies in $D_i[U_0]$.

(2) For $0 \leq i \leq n-1$, if $U_{i+1} \subseteq X \subseteq U_i$ and $X \subseteq V \subseteq U_0$, then $\mathcal{M}(X,V,D_n)$ can be computed in $O((\|D_i[U_0]\| + \text{TIME}(0)) . |U_i - X| . \prod_{j=1}^{i-1} (|U_j - U_{j+1}| + 1) + \|D_i\|)$ time.

3.3.2 An extension of Theorem3.1 to functional and template dependencies

Theorem3.1 can be extended to a class of functional and template dependencies. In the following we present the result. The detailed proof is found in [Ito et al 81b], and will be omitted here.

A template dependency over R [Sadri and Ullman 80], abbreviated to TD, is a statement $(t_1,\dots,t_p)/t$, where the t_i's and t are tuples of variables over R. No variable may occur in two distinct columns among the t_i's and t, but one variable may occur in the same column of some of the t_i's or t. The t_i's are called the hypothesis rows, and t is the conclusion row. A variable of the conclusion row is said to be unique if it does not occur in the hypothesis rows. A variable of the hypothesis rows is said to be repeated if it occurs in two or more of the hypothesis rows. For a TD d over R, non-unique(d) is defined as the set of attributes for which the conclusion row contains non-unique variables, and repeated(d) is the set of attributes for which at least one repeated variable occurs. Note that an MVD $X \twoheadrightarrow Y(Z)$ can be represented by a TD $d:(t_1,t_2)/t$ such that (1) $\{t_1, t_2\}$ is an X-agreed tableau, (2) $\{t, t_1\}$ is an XY-agreed tableau, and (3) $\{t, t_2\}$ is an $X(Z - Y)$-agreed tableau. Thus non-unique(d) = Z and repeated(d) = X.

A relation r over R is said to satisfy a TD $d:(t_1,\dots,t_p)/t$ over R if whenever there is a mapping h from variables of the hypothesis rows to entries of r such that $h(t_i)$ is a tuple of r for all i, r[non-unique(d)] contains tuple h(t[non-unique(d)]), where $h(v_1v_2\ldots v_n)$ is defined to be $h(v_1)h(v_2)\ldots h(v_n)$. Intuitively, r satisfies TD d if whenever we find tuples $\mu_1, \ldots, \mu_p$ of r with certain specific equalities among the entries of these tuples, we can find a tuple μ that has certain of its entries equal to certain of the entries in $\mu_1, \ldots, \mu_p$, and other entries of μ may be arbitrary.

We generalize the relation scheme $\langle R,D_n\rangle$ defined in Section 3.2 to a class of FDs and TDs as follows. Let $U_n \subsetneq \ldots \subsetneq U_1 \subsetneq U_0 \subseteq R$ and let $C_n = F \cup T \cup T_1 \cup \ldots \cup T_n$, where

(1) F is a set of FDs $Z \rightarrow W$ such that $Z \subseteq U_0$ or $W \cap U_0 = \emptyset$,

(2) T is a set of TDs $d:(t_1,\ldots,t_p)/t$ such that repeated(d) $\subseteq U_0 \subseteq$ non-unique(d) or there is a tuple t_j of the hypothesis rows that agrees with t in $U_0 \cap$ non-unique(d) columns, and

(3) each T_i for $1 \leq i \leq n$ is a set of TDs d such that non-unique(d) = U_i and repeated(d) $\subseteq U_0$.

Then we have the following theorem, which is a generalization of Theorem3.1.

[Theorem3.4] Consider the relation scheme $\langle R,C_n\rangle$ above. It is decidable whether a given TD d over R with non-unique(d) $\subseteq U_0$ (or a given FD $X \rightarrow Y$ with $XY \subseteq U_0$) is implied by C_n. []

We give a brief proof of this theorem below. Let $d:(t_1,\ldots,t_p)/t$ be a TD over R and let U be a subset of R. Then d[U] is defined to be $(t_1[U],\ldots,t_p[U])/t[U]$. If repeated(d) $\subseteq$ U, then d implies d[U] by an inference rule for TDs (weakening) [Sadri and Ullman 82]. Let $\langle R,C\rangle$ be a relation scheme, where C is a set of FDs and TDs. We define

$$C[U] = \{Z \rightarrow W \cap U \mid Z \rightarrow W \text{ is in } C \text{ and } Z \subseteq U\}$$
$$\cup \{d[U] \mid d \text{ is in } C \text{ and repeated}(d) \subseteq U\}.$$

Then we have the following lemma, which is a generalization of Lemma3.2.

[Lemma3.11] Let <R,C> be a relation scheme. Suppose that a subset U of R satisfies

(1) for all FD $Z \rightarrow W$ in C, $Z \subseteq U$ or $W \cap U = \emptyset$, and

(2) for all TD $d:(t_1,\ldots,t_p)/t$ in C, repeated(d) $\subseteq$ U or the conclusion row t agrees with a tuple of the hypothesis rows in U $\cap$ non-unique(d) columns.

Then a TD d over R with non-unique(d) $\subseteq$ U (or an FD $X \rightarrow Y$ with $XY \subseteq U$) is implied by C if and only if it is implied by C[U]. []

Note that for a TD $d:(t_1,\ldots,t_p)/t$ over R, if the conclusion row t agrees with a tuple of the hypothesis rows in U $\cap$ non-unique(d) columns, then d[U $\cap$ non-unique(d)] is a trivial TD, that is, any relation satisfies it [Sadri and Ullman 82]. Since U_0 satisfies condition of Lemma3.11, it follows that a TD d over R with non-unique(d) $\subseteq U_0$ (or an FD $X \rightarrow Y$ with $XY \subseteq U_0$) is implied by C_n if and only if it is implied by $C_n[U_0]$.

For a relation scheme <R,C> and a tableau r over R, a chase process of r under C can be defined [Sadri and Ullman 80]. Let d be a TD $(t_1,\ldots,t_p)/t$ over R. It is known that C implies d if and only if we can obtain a tuple t' that agrees with t in non-unique(d) columns by a chase process of $\{t_1, \ldots, t_p\}$ under C [Sadri and Ullman 80]. Thus if the chase process terminates, then it can be determined whether C implies d (and also a given FD). For the relation scheme $<R,C_n[U_0]>$, we have the following lemma, which is a generalization of Lemma3.3. Thus Theorem3.4 will follow from the discussions above, Lemmas 3.11 and 3.12.

[Lemma3.12] Let r_0 be a tableau over R. Then any chase process of r_0 under $C_n[U_0]$ finally terminates. []

CHAPTER 4

IMPLICATION PROBLEM FOR VIEW DEPENDENCIES

In this chapter, we consider implication problem for view dependencies. In Section 4.1, we briefly provide some definitions. In Section 4.2, we consider two decision problems on views: view nonemptiness problem and tuple membership problem. In Section 4.3, we consider implication problem for view dependencies.

4.1 Definitions

In this chapter, we often consider cross products or unions of relations. Thus it is convenient to refer to columns of relations (or tuples) by integers, called column numbers, instead of attributes. For example, $\mu[i]$ denotes the value of μ in the i-th column. If a relation r consists of m columns, then r is said to have degree m.

A value equality [Klug 80], abbreviated to VEQ, is a statement $A \equiv c$, where A is a column number and c is a constant. A tuple μ is said to satisfy $A \equiv c$ if $\mu[A] = c$. A relation r is said to satisfy $A \equiv c$ if every tuple of r satisfies $A \equiv c$. The selection of a relation r by $A \equiv c$ is a relation defined by $r[A \equiv c] = \{\mu \mid \mu$ is in r and satisfies $A \equiv c\}$. We often write $A_1 \ldots A_n \equiv c_1 \ldots c_n$ for a set $\{A_1 \equiv c_1, \ldots, A_n \equiv c_n\}$ of VEQs.

A domain equality [Klug 80], abbreviated to DEQ, is a statement $A = B$, where A and B are column numbers. A tuple μ is said to satisfy $A = B$ if $\mu[A] = \mu[B]$. A relation r is said to satisfy $A = B$ if every tuple of r satisfies $A = B$. The restriction of a relation r by $A = B$ is a relation defined by $r[A = B] = \{\mu \mid \mu$ is in r and satisifes $A = B\}$. We often write $A_1 \ldots A_n = B_1 \ldots B_n$ for a set $\{A_1 = B_1, \ldots, A_n = B_n\}$ of DEQs.

Let $r_1, \ldots, r_n$ be relations of degrees $m_1, \ldots, m_n$, respectively. The cross product of $r_1, \ldots, r_n$ is a relation defined by $r_1 \times \ldots \times r_n = \{\mu_1 \times \ldots \times \mu_n \mid \mu_i$ is in r_i for $1 \leq i \leq n\}$, where $\mu_1 \times \ldots \times \mu_n$ is the concatenation of $\mu_1, \ldots, \mu_n$. Note that $r_1 \times \ldots \times r_n$ has degree $\sum_{i=1}^{n} m_i$. If a column number A is in $\{1, \ldots, m_k\}$, then column number $\sum_{i=1}^{k-1} m_i + A$ for $r_1 \times \ldots \times r_n$ corresponds to column number A for r_k, that is, for each tuple μ of $r_1 \times \ldots \times r_n$, there is a tuple μ_k of r_k such that $\mu[\sum_{i=1}^{k-1} m_i + A] = \mu_k[A]$, and vice versa. For simplicity, we denote $\sum_{i=1}^{k-1} m_i + A$ by $A^{(k)}$. Similarly, for a subset $X = \{A_1, \ldots, A_\ell\}$ of $\{1, \ldots, m_k\}$, we denote $\{\sum_{i=1}^{k-1} m_i + A_1, \ldots, \sum_{i=1}^{k-1} m_i + A_\ell\}$ by $X^{(k)}$ $(= \{A_1^{(k)}, \ldots, A_\ell^{(k)}\})$.

Let $\underline{R} = \{\langle R_1, F_1 \cup M_1\rangle, \ldots, \langle R_n, F_n \cup M_n\rangle\}$ be a database scheme, where each F_i and M_i are sets of FDs and full MVDs, respectively. In this chapter, we consider only FDs and full MVDs as constraints. Thus we assume that "MVD" means "full MVD", and we simply write $X \twoheadrightarrow Y$ for $X \twoheadrightarrow Y(V)$. With each R_i is associated a degree, denoted $\deg(R_i)$. For a database $I = \{r_1, \ldots, r_n\}$ of $\underline{R}$, each r_i has degree $\deg(R_i)$. Let E be a relational expression consisting of $R_1, \ldots, R_n$ and five operations: projection, selection, restriction, cross product, and union. For every database I of $\underline{R}$, relation E(I) has the same degree. Thus we define degree of E, denoted deg(E), to be degree of E(I) for a database I of $\underline{R}$. In the following, E(I) is often called a view (of I with respect to E). Two relational expression E_1 and E_2 are said to be strongly equivalent if $E_1(I) = E_2(I)$ for every database I of $\underline{R}$. (In Section 2.3 before, we have defined that E_1 and E_2 are equivalent if $E_1(I) = E_2(I)$ for every database instance (not database) I of $\underline{R}$.)

4.2 Decision Problems on Views

In this chapter we consider the following two problems on views.

(1) View nonemptiness problem: Given a database scheme $\underline{R}$, a database I of $\underline{R}$, and a relational expression E, determine whether E(I) is not empty.

(2) Tuple membership problem: Given a database scheme $\underline{R}$, a database I of $\underline{R}$, a relational expression E, and a tuple μ, determine whether μ is in E(I).

We show that both problems are NP-complete in general, but if E contains no projection, then the tuple-membership problem can be solved in polynomial time.

4.2.1 NP-completeness results

[Lemma4.1] Let $\underline{R}$ be a database scheme, let I be a database of $\underline{R}$, and let E be a relational expression consisting only of restrictions and cross products. It is NP-hard to determine whether E(I) is not empty.

(Proof) We transform the 3-satisfiability problem [Garey and Johnson 79] into this problem. Let $P = Q_1 \wedge \dots \wedge Q_m$ be a conjunctive normal form Boolean expression, where each clause Q_i contains exactly three literals. Let $x_1, \dots, x_n$ be all variables occurring in P. We construct a database scheme $\underline{R}$, a database I of $\underline{R}$, and a relational expression E consisting of restrictions and cross products such that E(I) is not empty if and only if P is satisfiable.

Let x_{i1}, x_{i2}, x_{i3} be three variables occurring in Q_i. Let $\{\delta_{i1}^{(1)}, \delta_{i2}^{(1)}, \delta_{i3}^{(1)}\}, \dots, \{\delta_{i1}^{(7)}, \delta_{i2}^{(7)}, \delta_{i3}^{(7)}\}$ be the seven truth assignments to $\{x_{i1}, x_{i2}, x_{i3}\}$ that make Q_i true. Then we define $r_i = \{\delta_{i1}^{(j)}\delta_{i2}^{(j)}\delta_{i3}^{(j)} \mid 1 \leq j \leq 7\}$. Let μ $(= \delta_{11}\delta_{12}\delta_{13}\dots\delta_{m1}\delta_{m2}\delta_{m3})$ be a tuple of the cross product $r_1 \times \dots \times r_m$. Then μ can be considered to be a truth assignment to $\{x_{11}, x_{12}, x_{13}, \dots, x_{m1}, x_{m2}, x_{m3}\}$ that makes $Q_1, \dots, Q_m$ true "independently". In order for μ to represent a truth assignment to $\{x_1, \dots, x_n\}$, it is necessary and sufficient that for each variable x_i with $1 \leq i \leq n$, μ has the same value in all the columns, each which corresponds to a position of x_i occurring in P. That is, if the k-th variable of Q_s and the ℓ-th variable of Q_t are the same, then δ_{sk} and $\delta_{t\ell}$ must be equal. If μ satisfies the property, then μ is said to be <u>assignable</u> to $\{x_1, \ldots, x_n\}$. Clearly, P is satisfiable if and only if $r_1 \times \ldots \times r_m$ contains a tuple that is assignable to $\{x_1, \ldots, x_n\}$.

Let the 3m positions 1, 2, ..., 3m of variables occurring in P correspond to column numbers 1, 2, ..., 3m, respectively. We can choose all tuples that are assignable to $\{x_1, \ldots, x_n\}$ from $r_1 \times \ldots \times r_m$ by restrictions as follows. Let $\psi = \psi_1 \cup \ldots \cup \psi_n$ be a set of DEQs such that for each x_i, if $p_1, \ldots, p_j$ are all the positions of x_i occurring in P, then $\psi_i = \{p_1 = p_2, \ldots, p_1 = p_j\}$. Then relation $(r_1 \times \ldots \times r_m)[\psi]$ consists of exactly all tuples that are assignable to $\{x_1, \ldots, x_n\}$.

Let $\underline{R} = \{\langle R_1,\emptyset\rangle, \ldots, \langle R_m,\emptyset\rangle\}$, where $\deg(R_i) = 3$ for all R_i. Let $I = \{r_1, \ldots, r_m\}$ and let $E = (R_1 \times \ldots \times R_m)[\psi]$. Since $E(I) = (r_1 \times \ldots \times r_m)[\psi]$, relation E(I) is not empty if and only if $r_1 \times \ldots \times r_m$ contains a tuple that is assignable to $\{x_1, \ldots, x_n\}$. Thus P is satisfiable if and only if E(I) is not empty. Since $\underline{R}$, I, and E can be constructed from P in polynomial time, Lemma4.1 follows. []

[Lemma4.2] Let $\underline{R}$ be a database scheme, let I be a database of $\underline{R}$, let E be a relational expression, and let μ be a tuple. It can be determined in nondeterministic polynomial time whether μ is in E(I).

(Proof) Let $\underline{R} = \{\langle R_1,\emptyset\rangle, \ldots, \langle R_n,\emptyset\rangle\}$ and let $I = \{r_1, \ldots, r_n\}$. Suppose that μ is in E(I). Since E can be transformed into a strongly equivalent relational expression $E_1 \cup \ldots \cup E_p$ such that each term E_i contains no union, we assume that μ is in $E_i(I)$. Such an expression E_i can be obtained in nondeterministic polynomial time by repeating the operation of choosing nondeterministically either E_s or E_t for each union $E_s \cup E_t$ appearing in E.

Since E_i contains no union, E_i can be transformed into a strongly equivalent relational expression $(R_{k_1} \times \ldots \times R_{k_s})[Z \equiv \Delta][P = Q][V]$ in

$O(|E_i|^2 + \deg(E_i))$ time [Klug 80] [Smith and Chang 75] [Ullman 80]. Let $E_i' = (R_{k_1} \times \ldots \times R_{k_s})[Z \equiv \Delta][P = Q]$. Since μ is in $E_i(I)$, there is a tuple μ' of $E_i'(I)$ that agrees with μ in V columns and satisfies $Z \equiv \Delta$ and $P = Q$. Such a tuple μ' can be obtained in nondeterministic polynomial time by choosing nondeterministically $\deg(E_i') - \deg(E_i)$ elements from the set of constants in I as deleted columns. Since E_i' contains no projection, it can be determined in $O(|E_i'|.(|E_i'| + |I| + \deg(E_i')))$ time whether μ' is in $E_i'(I)$ by Theorem4.3 described below. Thus Lemma4.2 follows. []

[Lemma4.3] The view nonemptiness problem can be transformed into the tuple membership problem in polynomial time.

(Proof) Let $\underline{R} = \{\langle R_1,\emptyset\rangle, \ldots, \langle R_n,\emptyset\rangle\}$ be a database scheme, let $I = \{r_1, \ldots, r_n\}$ be a database of $\underline{R}$, and let E be a relational expression. Let $\langle R_0,\emptyset\rangle$ be an additional relation scheme and let $\underline{R}' = \{\langle R_0,\emptyset\rangle, \langle R_1,\emptyset\rangle, \ldots, \langle R_n,\emptyset\rangle\}$. We assume that $\deg(R_0) = 1$. Then let μ be a unary tuple and let $I' = \{\{\mu\}, r_1, \ldots, r_n\}$ be a database of $\underline{R}'$. Let $E' = (R_0 \times E)[1]$. Since $E'(I') = (\{\mu\} \times E(I))[1]$, if E(I) is empty, then so is $E'(I')$, and if E(I) is not empty, then $E'(I') = \{\mu\}$. That is, E(I) is not empty if and only if $E'(I')$ contains μ. Since $\underline{R}'$, I', E', and μ can be constructed from $\underline{R}$, I, and E in polynomial time, Lemma4.3 follows. Here, we note that E' can be obtained from E by adding one cross product and one projection. []

By Lemmas 4.1, 4.2, and 4.3, we have the following two theorems.

[Theorem4.1] Let $\underline{R}$ be a database scheme, let I be a database of $\underline{R}$, and let E be a relational expression. It is NP-complete to determine whether E(I) is not empty. Even if E consists only of restrictions and cross products, the problem is still NP-complete. []

[Theorem4.2] Let $\underline{R}$ be a database scheme, let I be a database of $\underline{R}$, let E be a relational expression, and let μ be a tuple. It is NP-complete to determine whether μ is in E(I). Even if E consists only of restrictions, cross products, and one projection, the problem is still NP-complete. []

4.2.2 A polynomial time algorithm

[Theorem4.3] Let $\underline{R} = \{\langle R_1,\emptyset\rangle, \ldots, \langle R_n,\emptyset\rangle\}$ be a database scheme, let $I = \{r_1, \ldots, r_n\}$ be a database of $\underline{R}$, let E be a relational expression with no projection, and let μ be a tuple. It can be determined in $O(|E|.(|E| + |I| + \deg(E)))$ time whether μ is in E(I).

(Proof) The problem can be solved by the following recursive way.

(1) If $E = R_i$, then examine whether μ is in r_i.

(2) If $E = E_1[\psi]$, where ψ is a VEQ or a DEQ, then examine whether μ is in $E_1(I)$ and satisfies ψ.

(3) If $E = E_1 \times E_2$, then examine whether $\mu[12\ldots\deg(E_1)]$ is in $E_1(I)$ and $\mu[\deg(E_1)+1\ldots\deg(E_1)+\deg(E_2)]$ is in $E_2(I)$.

(4) If $E = E_1 \cup E_2$, then examine whether μ is in $E_1(I)$ or μ is in $E_2(I)$.

We estimate the time for solving the problem. The problem can be divided into one or two subproblems by one of cases (1) through (4) in $O(|E| + |\mu|) \leq O(|E| + \deg(E))$ time, and then a solution of the problem can be obtained from solutions of subproblems in a constant time. Since for each operation in E, the problem is devided into at most two subproblems, the total number of subproblems is $O(|E|)$. In case (1), it can be examined in $O(|r_i|) \leq O(|I|)$ time whether μ is in r_i. In case (2), it can be examined in $O(|\mu|) \leq O(\deg(E))$ time whether μ satisfies ψ. Thus the problem can be solved in $O(|E|.(|E| + |I| + \deg(E)))$ time. []

4.3 Implication Problem for View Dependencies

Let $\underline{R} = \{\langle R_1, F_1 \cup M_1\rangle, \ldots, \langle R_n, F_n \cup M_n\rangle\}$ be a database scheme, let E be a relational expression, and let d be one of FD, MVD, VEQ, and DEQ. Then d is said to be *valid* in E over $\underline{R}$ if for every database instance I of $\underline{R}$, E(I) satisfies d. In a special case where $E = R_i$, d is valid in E over $\underline{R}$ if and only if d is implied by $F_i \cup M_i$. In this section, we often consider a chase process of a relation r under a set M of MVDs. Then chase(r,M) is defined as the unique minimum relation that contains r and satisfies M [Maier et al 79]. That is, if a relation r' contains r and satisfies M, then r' also contains chase(r,M).

4.3.1 A decidability result

In this section we show that it is decidable whether a given FD or MVD d is valid in a given relational expression E over a given database scheme $\underline{R}$. The basic idea is that in order to determine whether d is not valid in E over $\underline{R}$, it suffices to examine whether there is a database instance I of $\underline{R}$ such that E(I) does not satisfy d in a finite set of database instances of $\underline{R}$ defined by $\underline{R}$ and E.

[Lemma4.4] Let $\underline{R} = \{\langle R_1, F_1 \cup M_1\rangle, \ldots, \langle R_n, F_n \cup M_n\rangle\}$ be a database scheme, let E be a relational expression, and let d be an FD or MVD. If d is not valid in E over $\underline{R}$, then there is a database instance I of $\underline{R}$ such that (1) E(I) does not satisfy d and (2) the number of distinct constants of I is at most $2 \times c_E \times \max_{1 \leq \ell \leq n}\{\deg(R_\ell)\}$, where c_E is the number of occurrences of $R_1, \ldots, R_n$ in E.

(Proof) For simplicity, we denote $U_i = \{1, \ldots, \deg(R_i)\}$ for $1 \leq i \leq n$. We shall consider the case where d is an MVD $X \twoheadrightarrow Y$. The same argument applies also to FDs. Suppose that $X \twoheadrightarrow Y$ is not valid in E over $\underline{R}$. Then there is a database instance $I = \{r_1, \ldots, r_n\}$ of $\underline{R}$ such that $E(I)$ does not satisfy $X \twoheadrightarrow Y$. Let $U = \{1, \ldots, \deg(E)\}$ and let $Z = U - XY$. Since $E(I)$ does not satisfy $X \twoheadrightarrow Y$, there are tuples μ and ν of $E(I)$ and τ not of $E(I)$ such that $\mu[X] = \nu[X] = \tau[X]$, $\mu[Y] = \tau[Y]$ ($\neq \nu[Y]$), and $\nu[Z] = \tau[Z]$ ($\neq \mu[Z]$). In the following we shall show that there is a database instance I' of $\underline{R}$ defined only by μ and ν such that $E(I')$ contains μ and ν but does not contain τ.

E can be transformed into a strongly equivalent relational expression $E_1 \cup \ldots \cup E_p$ such that each term E_i contains no union. Since $E(I) = E_1(I) \cup \ldots \cup E_p(I)$, we assume that μ is in $E_i(I)$ and ν is in $E_j(I)$ (possibly, $i = j$). Since E_i contains no union, E_i can be transformed into a strongly equivalent relational expression $(R_{k_1} X \ldots X R_{k_s})[Z \equiv \Delta][P = Q][V]$, where $Z \equiv \Delta$ and $P = Q$ are sets of VEQs and DEQs, respectively. For simplicity, let $E_i' = (R_{k_1} X \ldots X R_{k_s})[Z \equiv \Delta][P = Q]$. There is a tuple μ' of $E_i'(I)$ that agrees with μ in V columns. We define that the <u>projection</u> <u>mapping</u> of μ' with respect to E_i' is the minimum database $I_{\mu'} = \{r_1', \ldots, r_n'\}$ of $\underline{R}$ such that r_{k_t}' contains $\mu'[U_{k_t}^{(t)}]$ for $1 \leq t \leq s$. That is, if $R_{k_{t1}}, \ldots, R_{k_{tq}}$ are all occurrences of R_ℓ in E_i', then $r_\ell' = \{\mu'[U_\ell^{(t1)}], \ldots, \mu'[U_\ell^{(tq)}]\}$. Since μ' is the concatenation $\mu'[U_{k_1}^{(1)}] X \ldots X \mu'[U_{k_s}^{(s)}]$, $E_i'(I_{\mu'})$ contains μ'. Similarly, E_j can be transformed into a strongly equivalent relational expression $E_j'[V']$, and there is a tuple ν' of $E_j'(I)$ that agrees with ν in V' columns. Let $I_{\nu'} = \{r_1'', \ldots, r_n''\}$ be the projection mapping of ν' with respect to E_j'. Then it holds that (1) r_ℓ contains $r_\ell' \cup r_\ell''$ for $1 \leq \ell \leq n$, (2) $E_i(I')$ contains μ, and (3) $E_j(I')$ contains ν.

We define $I' = \{\mathrm{chase}(r_1' \cup r_1'', M_1), \ldots, \mathrm{chase}(r_n' \cup r_n'', M_n)\}$. Then we have the following fact, whose proof is given in Appendix 3.

[Fact4.1] I' is a database instance of $\underline{R}$, and $E(I')$ does not satisfy

$X \twoheadrightarrow Y$. []

The number of distinct constants of I' is equal to that of μ' and ν', and thus at most $\deg(E_i') + \deg(E_j')$. The number s of occurrences of $R_1, \ldots, R_n$ in E_i' is at most the number c_E of occurrences of $R_1, \ldots, R_n$ in E. Thus, $\deg(E_i') = \sum_{t=1}^{s} \deg(R_{k_t}) \leqq c_E \times \max_{1 \leqq \ell \leqq n}\{\deg(R_\ell)\}$. Similarly, $\deg(E_j') \leqq c_E \times \max_{1 \leqq \ell \leqq n}\{\deg(R_\ell)\}$. Thus, $\deg(E_i') + \deg(E_j') \leqq 2 \times c_E \times \max_{1 \leqq \ell \leqq n}\{\deg(R_\ell)\}$. Thus Lemma4.4 follows. []

By Lemma4.4, we have the following theorem.

[Theorem4.4] Let $\underline{R} = \{\langle R_1,F_1 \cup M_1\rangle, \ldots, \langle R_n,F_n \cup M_n\rangle\}$ be a database scheme, let E be a relational expression, and let d be an FD or MVD. It is decidable whether d is valid in E over $\underline{R}$.

(Proof) We shall consider the case where d is an MVD $X \twoheadrightarrow Y$. Suppose that $X \twoheadrightarrow Y$ is not valid in E over $\underline{R}$. For simplicity, let $k = 2 \times c_E \times \max_{1 \leqq \ell \leqq n}\{\deg(R_\ell)\}$. By Lemma4.4, there is a database instance I of $\underline{R}$ such that E(I) does not satisfy $X \twoheadrightarrow Y$ and the number of distinct constants of I is at most k. Each constant of I either appears in E (as the right-hand side of a VEQ) or does not appear in E. Here, the number of distinct constants not appearing in E is at most k.

Let S be the union of the set of constants appearing in E and another k constants. By the discussions above, if $X \twoheadrightarrow Y$ is not valid in E over $\underline{R}$, then there is a database instance I of $\underline{R}$ such that each constant of I is in S and E(I) does not satisfy $X \twoheadrightarrow Y$. Conversely, if there is such a database instance I of $\underline{R}$, then $X \twoheadrightarrow Y$ is not valid in E over $\underline{R}$. Since S is finite, the number of database instances of $\underline{R}$ consisting of S is also finite. Thus Theorem4.4 follows. []

4.3.2 An NP-completeness result

In this section we show that if $\underline{R}$ contains only FDs as constraints, then it is NP-complete to determine whether a given FD is not valid in a given relational expression over $\underline{R}$.

[Lemma4.5] Let $\underline{R} = \{\langle R_1,F_1\rangle, \ldots, \langle R_n,F_n\rangle\}$ be a database scheme, let E be a relational expression, and let d be an FD $X \rightarrow Y$. It can be determined in nondeterministic polynomial time whether d is not valid in E over $\underline{R}$.

(Proof) Suppose that $X \rightarrow Y$ is not valid in E over $\underline{R}$. In the proof of Lemma4.4, it is shown that $I' = \{chase(r_1' \cup r_1'',M_1), \ldots, chase(r_n' \cup r_n'',M_n)\}$ is a database instance of $\underline{R}$ such that $E(I')$ does not satisfy $X \rightarrow\rightarrow Y$ (and thus, does not satisfy $X \rightarrow Y$). In this case, since $\underline{R}$ contains no MVD, it follows that $I'' = \{r_1' \cup r_1'', \ldots, r_n' \cup r_n''\}$ is a database instance of $\underline{R}$ such that $E(I'')$ does not satisfy $X \rightarrow Y$. Here, the size of the description of I" is bounded by $2 \times c_E \times \max_{1 \leqq \ell \leqq n}\{deg(R_\ell)\}$. Thus we can determine in nondeterministic polynomial time whether $X \rightarrow Y$ is not valid in E as follows.

(1) Let $k = 2 \times c_E \times \max_{1 \leqq \ell \leqq n}\{deg(R_\ell)\}$. Let S be the union of the set of constants appearing in E and another k constants.

(2) Guess a number k' within k. Construct a database instance I of $\underline{R}$ by choosing nondeterministically k' elements from S. Note that given a database $I = \{r_1, \ldots, r_n\}$ of $\underline{R}$, we can examine in polynomial time whether I is a database instance of $\underline{R}$ by checking whether each relation r_ℓ satisfies F_ℓ.

(3) Construct two deg(E)-tuples μ and ν such that $\mu[X] = \nu[X]$ and $\mu[Y] \neq \nu[Y]$ by choosing nondeterministically $2 \times deg(E)$ elements from S.

(4) Examine whether both μ and ν are in E(I). This can be done in nondeterministic polynomial time by Lemma4.2. [ ]

The difficulty of implication problems is mainly caused by the fact that if a given relational expression E is transformed into a strongly equivalent relational expression $E_1 \cup \ldots \cup E_p$ such that each term E_i contains no union, then the size of $E_1 \cup \ldots \cup E_p$ may be exponential to that of E. In fact, we have the following lemma.

[Lemma4.6] Let $\underline{R} = \{\langle R_1,F_1\rangle, \ldots, \langle R_n,F_n\rangle\}$ be a database scheme, let E be a relational expression consisting only of selections, cross products, and unions, and let d be an FD. It is NP-hard to determine whether d is not valid in E over $\underline{R}$.

(Proof) E is said to be sound under $\underline{R}$ if there is a database instance I of $\underline{R}$ such that E(I) is not empty. First we transform the 3-satisfiability problem into the problem of determining whether E is sound under $\underline{R}$, called the soundness problem for E. Let $P = Q_1 \wedge \ldots \wedge Q_m$ be a conjunctive normal form Boolean expression, where each clause Q_i contains exactly three literals. Let $x_1, \ldots, x_n$ be all variables occurring in P. We denote $Q_i = x_{i1}^{\omega_{i1}} \vee x_{i2}^{\omega_{i2}} \vee x_{i3}^{\omega_{i3}}$, where ω_{ik} is either zero or one and $x_{ik}^1 = x_{ik}$ and $x_{ik}^0 = \overline{x}_{ik}$. We construct a database scheme $\underline{R}$ containing only FDs and a relational expression E_P consisting of selections, cross products, and unions such that E_P is sound under $\underline{R}$ if and only if P is satisfiable.

Let $\underline{R} = \{\langle R_1,\{1 \rightarrow 2\}\rangle, \ldots, \langle R_n,\{1 \rightarrow 2\}\rangle\}$, where $\deg(R_i) = 2$ for $1 \leq i \leq n$. Let $E_P = G_1 \times \ldots \times G_m$, where each G_i corresponds to clause Q_i and $G_i = R_{i1}[12 \equiv c_2c_{\omega_{i1}}] \cup R_{i2}[12 \equiv c_2c_{\omega_{i2}}] \cup R_{i3}[12 \equiv c_2c_{\omega_{i3}}]$. For example, if $P = (x_1 \vee \overline{x}_2 \vee x_3) \wedge (x_1 \vee \overline{x}_3 \vee \overline{x}_4) \wedge (\overline{x}_1 \vee x_2 \vee x_4)$,

$$\begin{aligned}\text{then } E_P = {} & (R_1[12 \equiv c_2c_1] \cup R_2[12 \equiv c_2c_0] \cup R_3[12 \equiv c_2c_1]) \\ & \times (R_1[12 \equiv c_2c_1] \cup R_3[12 \equiv c_2c_0] \cup R_4[12 \equiv c_2c_0]) \\ & \times (R_1[12 \equiv c_2c_0] \cup R_2[12 \equiv c_2c_1] \cup R_4[12 \equiv c_2c_1]).\end{aligned}$$

Let σ be a truth assignment to $\{x_1, \ldots, x_n\}$. We define a database instance

$I_\sigma = \{r_1, \ldots, r_n\}$ of $\underline{R}$ such that under the truth assignment σ, if $x_j = 1$, then $r_j = \{c_2c_1\}$, and otherwise $r_j = \{c_2c_0\}$. Clearly if P is true under the truth assignment σ, then $E_P(I_\sigma)$ is not empty, and otherwise $E_P(I_\sigma)$ is empty. Conversely, let $I = \{r_1, \ldots, r_n\}$ be a database instance of $\underline{R}$ such that each r_j contains c_2c_1 or c_2c_0. Since r_j can not contain both c_2c_1 and c_2c_0 by FD $1 \rightarrow 2$, we define a truth assignment $\sigma = \{\delta_1, \ldots, \delta_n\}$ to $\{x_1, \ldots, x_n\}$ such that if r_j contains c_2c_1, then $\delta_j = 1$, and if r_j contains c_2c_0, then $\delta_j = 0$. Clearly if $E_P(I)$ is not empty, then P is true under the truth assignment σ, and otherwise P is not true. Thus the soundness problem for E is NP-hard.

The soundness problem for E can be transformed into the complement of the implication problem as follows. Let $<R_0,\emptyset>$ be an additional relation scheme, where $\deg(R_0) = 2$, and let $\underline{R}' = \{<R_0,\emptyset>, <R_1,\{1 \rightarrow 2\}>, \ldots, <R_n,\{1 \rightarrow 2\}>\}$. Consider relational expression $E = R_0 \times E_P$ and FD $1 \rightarrow 2$. Since $1 \rightarrow 2$ is not valid in R_0 over $\underline{R}'$, if there is a database instance I' of $\underline{R}'$ such that $E(I')$ is not empty, then $1 \rightarrow 2$ is not valid in E over $\underline{R}'$. Conversely, if there is no such database instance I' of $\underline{R}'$, then $1 \rightarrow 2$ is trivially valid in E over $\underline{R}'$. Clearly, there is a database instance I' of $\underline{R}'$ such that $E(I')$ is not empty if and only if there is a database instance I of $\underline{R}$ such that $E_P(I)$ is not empty. Thus, FD $1 \rightarrow 2$ is <u>not</u> valid in E over $\underline{R}'$ if and only if E_P is sound under $\underline{R}$. Thus Lemma4.6 follows. [ ]

By Lemmas 4.5 and 4.6, we have the following theorem.

[Theorem4.5] Let $\underline{R} = \{<R_1,F_1>, \ldots, <R_nF_n>\}$ be a database scheme, let E be a relational expression and let d be an FD. It is NP-complete to determine whether d is not valid in E over $\underline{R}$. Even if E consists only of selections, cross products, and unions, the problem is still NP-complete. [ ]

4.3.3 An NP-hardness result

In this section we show that even if a given relational expression E consists only of selections and cross products, it is NP-hard to determine whether a given FD or MVD is valid in E over a given database scheme.

[Lemma4.7] Let r be a relation and let M be a set of MVDs. The problem: "Determine whether chase(r,M) does not satisfy a given FD d" is NP-hard.

(Proof) We transform the 3-satisfiability problem into this problem. Let $P = Q_1 \wedge \ldots \wedge Q_m$, where $Q_i = x_{i1}^{\omega_{i1}} \vee x_{i2}^{\omega_{i2}} \vee x_{i3}^{\omega_{i3}}$. Let $x_1, \ldots, x_n$ be all variables occurring in P. We construct a relation r, an FD d, and a set M of MVDs such that chase(r,M) does not satisfy d if and only if P is satisfiable.

Let $U = \{1, \ldots, n + 2m + 1\}$ be a set of column numbers. For simplicity, we denote $1, \ldots, n$; $n + 1, \ldots, n + m$; $n + m + 1, \ldots, n + 2m$; $n + 2m + 1$ by $X_1, \ldots, X_n$; $Y_1, \ldots, Y_m$; $Z_1, \ldots, Z_m$; W, respectively. Each X_j for $1 \leqq j \leqq n$ corresponds to variable x_j, and each Y_i and Z_i for $1 \leqq i \leqq m$ correspond to clause Q_i. Let r be a relation of degree $n + 2m + 1$ with $(7m + 3)$ tuples as follows.

(i) $r = \{\mu_0, \mu_1, \mu_2\} \cup T_1 \cup T_2 \cup \ldots \cup T_m$.

(ii)

	$X_1 \ldots X_n$	$Y_1 \ldots Y_m$	$Z_1 \ldots Z_m$	W
μ_0:	c...c	1...1	c...c	v
μ_1:	1...1	b...b	b...b	u
μ_2:	0...0	b...b	b...b	u

(iii) Each T_i for $1 \leqq i \leqq m$ corresponds to clause $Q_i = x_{i1}^{\omega_{i1}} \vee x_{i2}^{\omega_{i2}} \vee x_{i3}^{\omega_{i3}}$. Let $\{\delta_{i1}^{(1)}, \delta_{i2}^{(1)}, \delta_{i3}^{(1)}\}, \ldots, \{\delta_{i1}^{(7)}, \delta_{i2}^{(7)}, \delta_{i3}^{(7)}\}$ be the seven truth assignments to $\{x_{i1}, x_{i2}, x_{i3}\}$ that make Q_i true. Then T_i has seven tuples $\nu_{i1}, \ldots, \nu_{i7}$ that correspond to the seven truth assignments, respectively, as follows.

(a) $\nu_{ik}[X_{i1}X_{i2}X_{i3}] = \delta_{i1}^{(k)}\delta_{i2}^{(k)}\delta_{i3}^{(k)}$, and ν_{ik} has constant b in all other X columns.

(b) $\nu_{ik}[Y_i] = 1$, and ν_{ik} has constant b in all other Y columns.

(c) $\nu_{ik}[Z_i] = a_{ik}$, and ν_{ik} has constant b in all other Z columns.

(d) $\nu_{ik}[W] = u$.

The following table illustrates T_i.

	...	X_{i1}	...	X_{i2}	...	X_{i3}	...	...	Y_i	...	...	Z_i	...	W
ν_{i1}:	b	$\delta_{i1}^{(1)}$	b	$\delta_{i2}^{(1)}$	b	$\delta_{i3}^{(1)}$	b	b	1	b	b	a_{i1}	b	u
ν_{i2}:	b	$\delta_{i1}^{(2)}$	b	$\delta_{i2}^{(2)}$	b	$\delta_{i3}^{(2)}$	b	b	1	b	b	a_{i2}	b	u
...			...						...			...		
ν_{i7}:	b	$\delta_{i1}^{(7)}$	b	$\delta_{i2}^{(7)}$	b	$\delta_{i3}^{(7)}$	b	b	1	b	b	a_{i7}	b	u

Here we assume that 0, 1, b, c, u, v, a_{11}, ..., a_{m7} are distinct constants. Let d be FD $Y_1 \ldots Y_m \rightarrow W$. Let M be a set of MVDs consisting of the following (n + m) MVDs.

$$\mathrm{MVD}_1\colon Y_1 \ldots Y_m Z_1 \ldots Z_m \twoheadrightarrow X_1$$

...

$$\mathrm{MVD}_n\colon Y_1 \ldots Y_m Z_1 \ldots Z_m \twoheadrightarrow X_n$$

$$\mathrm{MVD}_{n+1}\colon X_{11}X_{12}X_{13} \twoheadrightarrow Y_1 Z_1$$

...

$$\mathrm{MVD}_{n+m}\colon X_{m1}X_{m2}X_{m3} \twoheadrightarrow Y_m Z_m$$

Note that r, d, and M can be constructed from P in polynomial time.

If part: We show that if P is satisfiable, then chase(r,M) does not satisfy FD d. Suppose that a truth assignment $\{\delta_1, \ldots, \delta_n\}$ to $\{x_1, \ldots, x_n\}$ makes P true.

Since (1) $\mu_1[Y_1 \ldots Y_m Z_1 \ldots Z_m] = \mu_2[Y_1 \ldots Y_m Z_1 \ldots Z_m] = b \ldots b$, (2) $\mu_1[X_1 \ldots X_n] = 1 \ldots 1$ and (3) $\mu_2[X_1 \ldots X_n] = 0 \ldots 0$, the chase of $\{\mu_1, \mu_2\}$ under $\{\mathrm{MVD}_1, \ldots, \mathrm{MVD}_n\}$ consists of 2^n tuples as follows.

$X_1 \ldots X_n$	$Y_1 \ldots Y_m$	$Z_1 \ldots Z_m$	W
0...00	b...b	b...b	u
0...01	b...b	b...b	u
0...10	b...b	b...b	u
...	...	...	
1...11	b...b	b...b	u

That is, all possible truth assignments to $\{x_1, \ldots, x_n\}$ appear in $X_1 \ldots X_n$ columns of the chase. Thus the chase contains tuple $\delta_1 \ldots \delta_n b \ldots bu$, denoted τ. Thus chase(r,M) also contains tuple τ. For $1 \leqq i \leqq m$, since $\{x_{i1}, x_{i2}, x_{i3}\} = \{\delta_{i1}, \delta_{i2}, \delta_{i3}\}$ makes Q_i true, there is a tuple ν_{ik_i} of T_i such that $\nu_{ik_i}[X_{i1}X_{i2}X_{i3}] = \tau[X_{i1}X_{i2}X_{i3}]$, where $1 \leqq k_i \leqq 7$. Noting that $\nu_{1k_1}[Y_1Z_1] = 1a_{1k_1}, \ldots, \nu_{mk_m}[Y_mZ_m] = 1a_{mk_m}$, we have tuple $\delta_1 \ldots \delta_n 1 \ldots 1 a_{1k_1} \ldots a_{mk_m} u$, denoted τ', by the chase of $\{\tau, \nu_{1k_1}, \ldots, \nu_{mk_m}\}$ under $\{MVD_{n+1}, \ldots, MVD_{n+m}\}$. Thus chase(r,M) also contains tuple τ'. Since $\mu_0[Y_1 \ldots Y_m] = \tau'[Y_1 \ldots Y_m]$, $\mu_0[W] = v$, and $\tau'[W] = u$, we conclude that chase(r,M) does not satisfy $Y_1 \ldots Y_m \rightarrow W$.

<u>Only if part</u>: We show that if chase(r,M) does not satisfy FD d, then P is satisfiable. Let $\overline{r} = r - \{\mu_0\}$. Then we have the following two facts, whose proofs are given in Appendix 3.

[Fact4.2] If chase(r,M) does not satisfy FD d, then chase($\overline{r}$,M) contains a tuple τ such that $\tau[Y_1 \ldots Y_m] = 1 \ldots 1$. [ ]

[Fact4.3] Let τ' be a tuple of chase($\overline{r}$,M). If $\tau'[Y_i] = 1$, then $\{\tau'[X_{i1}], \tau'[X_{i2}], \tau'[X_{i3}]\}$ is a truth assignment to $\{x_{i1}, x_{i2}, x_{i3}\}$ that makes Q_i true. [ ]

Suppose that chase(r,M) does not satisfy FD d. By Fact4.2, chase($\overline{r}$,M) contains a tuple τ such that $\tau[Y_1 \ldots Y_m] = 1 \ldots 1$. By Fact4.3, for $1 \leqq i \leqq m$,

$\{x_{i1}, x_{i2}, x_{i3}\} = \{\tau[X_{i1}], \tau[X_{i2}], \tau[X_{i3}]\}$ makes Q_i true, and thus $\{x_1, \ldots, x_n\} = \{\tau[X_1], \ldots, \tau[X_n]\}$ makes P true. []

By Lemma4.7, we have the following theorem.

[Theorem4.6] Let $\underline{R} = \{\langle R_1, F_1 \cup M_1\rangle, \ldots, \langle R_n, F_n \cup M_n\rangle\}$ be a database scheme, let E be a relational expression consisting only of selections and cross products, and let d be an FD or MVD. It is NP-hard to determine whether d is valid in E over $\underline{R}$.

(Proof) We transform the problem of Lemma4.7 (i.e., given a relation $r = \{\nu_1, \ldots, \nu_k\}$, an FD d, and a set M of MVDs, to determine whether chase(r,M) does not satisfy d) into the complement of the soundness problem.

Let $\underline{R} = \{\langle R, \{d\} \cup M\rangle\}$, where deg(R) is equal to degree of r, and let $E = R[U \equiv \nu_1] \times R[U \equiv \nu_2] \times \ldots \times R[U \equiv \nu_k]$, where $U = \{1, \ldots, \deg(R)\}$. Then chase(r,M) does not satisfy FD d if and only if E is <u>not</u> sound under $\underline{R}$, as explained below.

Suppose that chase(r,M) satisfies FD d. Then chase(r,M) satisfies $\{d\} \cup M$, and thus I = {chase(r,M)} is a database instance of $\underline{R}$ such that E(I) is not empty. Thus E is sound under $\underline{R}$. Conversely, suppose that there is a database instance $I = \{r'\}$ of $\underline{R}$ such that E(I) is not empty. Since r' must contain the k tuples $\nu_1, \ldots, \nu_k$, r' contains r. Furthermore since r' satisfies $\{d\} \cup M$, it follows from the definition of the chase that r' contains chase(r,M). Since r' satisfies FD d, chase(r,M) also satisfies FD d.

By Lemma4.7, the complement of the soundness problem for E is NP-hard. The soundness problem can be transformed into the complement of the implication problem as presented in the proof of Lemma4.6. Thus Theorem4.6 follows. []

Let E be a relational expression that may contain unions. By Lemma4.6, it is NP-hard to determine whether a given FD or MVD is <u>not</u> valid in E over $\underline{R}$. Thus by Theorem4.6, it is NP-hard and co-NP-hard to determine whether a given FD or MVD is valid in E over $\underline{R}$. This fact suggests that it is not likely that there is a nondeterministic polynomial time algorithm for determining whether a given FD or MVD is valid (or not valid) in E over $\underline{R}$ (cf. [Garey and Johnson 79]). It is not known whether there is such a nondeterministic polynomial time algorithm if E contains no union.

4.3.4 A polynomial time algorithm (1)

Let $\underline{R} = \{\langle R_1,F_1\rangle, \ldots, \langle R_n,F_n\rangle\}$ be a database scheme. For simplicity, we denote $U_i = \{1, \ldots, \deg(R_i)\}$ for $1 \leq i \leq n$. Let E be a relational expression and let $U = \{1, \ldots, \deg(E)\}$. Let X be a subset of U. Then we can define the closure $\mathcal{F}(X,E)$ of X with respect to E over $\underline{R}$, that is, $\mathcal{F}(X,E) = \{A \mid X \rightarrow A \text{ is valid in E over } \underline{R}\}$. Note that $\mathcal{F}(X,R_i) = \mathcal{F}(X,F_i)$. In this section, we consider the case where E contains no union.

Suppose that E contains no union. Then E can be transformed into a strongly equivalent relational expression $(R_{k_1} \times \ldots \times R_{k_s})[Z \equiv \Delta][P = Q][V]$ in $O(|E|^2 + \deg(E))$ time. Let X be a subset of V. It follows from the definition of FDs that $\mathcal{F}(X,E) = \mathcal{F}(X,(R_{k_1} \times \ldots \times R_{k_s})[Z \equiv \Delta][P = Q]) \cap V$. Thus without loss of generality we consider only the case where E is of the form $(R_{k_1} \times \ldots \times R_{k_s})[Z \equiv \Delta][P = Q]$.

Let $\mathcal{E}(E)$ be a partition of U such that for all A and B in U, DEQ $A = B$ is valid in E if and only if A and B are in the same block in $\mathcal{E}(E)$. That is, $\mathcal{E}(E)$ is the <u>equivalence class</u> with respect to all valid DEQs in E. If E is not sound under $\underline{R}$, then any dependency is trivially valid in E, and thus for any subset X of U, $\mathcal{F}(X,E)$ coincides with U. First, we show that the soundness problem for E can be solved in polynomial time and $\mathcal{E}(E)$ can be computed in polynomial time. Next, we show that $\mathcal{F}(X,E)$ can be computed in polynomial time in the case where E is sound under $\underline{R}$. (In order to solve the soundness problem for E and compute $\mathcal{F}(X,E)$, it is useful to compute $\mathcal{E}(E)$.)

Soundness test and computation of the equivalence class

Considering the following two facts (a) and (b), it is easy to see that Algorithm4.1 described below generates a refinement of $\mathcal{E}(E)$ (that is, if A and B are in the same block in the partition obtained by Algorithm4.1, then DEQ $A = B$ is valid in E). By fact (c) below, the soundness test for E can be done. We shall show in Lemma4.8 that the partition obtained by Algorithm4.1 coincides with $\mathcal{E}(E)$ and that the soundness test is correct.

(a) If $A \equiv c$ and $B \equiv c$ with the same constant c are in $Z \equiv \Delta$, then DEQ $A = B$ is valid in E.

(b) Let r be a relation and let μ be a tuple of $r \times r$. If r satisfies $X \rightarrow A$ and $\mu[X^{(1)}] = \mu[X^{(2)}]$, then it follows from the definition of FDs that $\mu[A^{(1)}] = \mu[A^{(2)}]$. Generally for relational expression E, if (1) $R_{k_i} = R_{k_j} = R_\ell$ (that is, R_{k_i} and R_{k_j} are occurrences of the same R_ℓ), (2) DEQ $X^{(i)} = X^{(j)}$ is valid in E, and (3) F_ℓ implies $X \rightarrow A$, then so is DEQ $A^{(i)} = A^{(j)}$.

(c) Let PL be a refinement of $\mathcal{E}(E)$. PL is said to be compatible with respect to $Z \equiv \Delta$ if there are no VEQs $A \equiv c$ and $B \equiv c'$ with $c \neq c'$ in $Z \equiv \Delta$ such that A and B are in the same block in PL. If PL is not compatible with respect to $Z \equiv \Delta$, then E is not sound under $\underline{R}$.

[Algorithm4.1]

input: $\underline{R} = \{<R_1,F_1>, \ldots, <R_n,F_n>\}$, and

$E = (R_{k_1} \times \ldots \times R_{k_s})[Z \equiv \Delta][P = Q]$.

output: $\mathcal{E}(E)$.

method:

(1) Let PL = $\{\{1\}, \ldots, \{\deg(E)\}\}$.

(2) For each DEQ $A = B$ in $P = Q$, merge the block L_A containing A and the one L_B containing B. That is, delete L_A and L_B from PL and add $L_A \cup L_B$ to PL. Repeat this step until no blocks can be merged anymore.

(3) For each pair $A \equiv c$ and $B \equiv c$ with the same constant c in $Z \equiv \Delta$, merge the block containing A and the one containing B. (DEQ $A = B$ is valid in E by fact (a).) Repeat this step until no blocks can be merged anymore.

(4) Repeat the following steps (i) through (iii) for each R_{k_i} and R_{k_j} such that $R_{k_i} = R_{k_j} = R_\ell$ $(i \neq j)$ until no blocks can be merged anymore.

(i) Let $W = \{A \mid A$ is in U_ℓ, and $A^{(i)}$ and $A^{(j)}$ are in the same block in PL$\}$. (For such a set W, DEQ $W^{(i)} = W^{(j)}$ is valid in E.)

(ii) Compute $\mathcal{F}(W,F_\ell)$.

(iii) For each A in $\mathcal{F}(W,F_\ell)$, merge the block containing $A^{(i)}$ and the one containing $A^{(j)}$. (DEQ $A^{(i)} = A^{(j)}$ is valid in E by fact (b).)

(5) If PL is compatible with respect to $Z \equiv \Delta$, then output PL as $\mathcal{E}(E)$. (In this case, E is sound under $\underline{R}$ as will be shown in Lemma4.8.) Otherwise, E is not sound under $\underline{R}$. (This decision is correct by fact (c).) []

[Lemma4.8] Let PL_{final} be the final partition obtained by Algorithm4.1. If PL_{final} is compatible with respect to $Z \equiv \Delta$, then E is sound under $\underline{R}$ and PL_{final} coincides with $\mathcal{E}(E)$.

(Proof) By the disccussions above, it suffices to show that there is a database instance I of $\underline{R}$ such that if A and B are in different blocks in PL_{final}, then E(I) does not satisfy DEQ $A = B$. Let μ be a deg(E)-tuple satisfying the following two conditions. (There is such a tuple μ, since PL_{final} is compatible with respect to $Z \equiv \Delta$.)

(1) For all A and B in U, if A and B are in the same block in PL_{final}, then $\mu[A] = \mu[B]$, and otherwise $\mu[A] \neq \mu[B]$.

(2) $\mu[Z] = \Delta$.

Let $I_\mu = \{r_1, ..., r_n\}$ be the projection mapping of μ with respect to E. Since μ is the concatenation $\mu[U^{(1)}_{k_1}] \times ... \times \mu[U^{(s)}_{k_s}]$ and each relation r_{k_i} in I_μ contains tuple $\mu[U^{(i)}_{k_i}]$, relation $E(I_\mu)$ contains tuple μ. Condition (1) means that if A and B are in different blocks in PL_{final}, then $E(I_\mu)$ does not satisfy DEQ A = B. Moreover, I_μ is a database instance of $\underline{R}$ by the following fact, whose proof is given in Appendix 3. Thus Lemma4.8 follows.

[Fact4.4] Each relation r_ℓ in I_μ satisfies F_ℓ. []

We estimate the time complexity of Algorithm4.1. Steps (1) through (3) can be executed in $O(|E|^2 + deg(E))$ time. Step (5) can be executed in $O(|Z|^2) \leq O(|E|^2)$ time. Considering the following three facts, the loop of step (4) can be executed in $O(deg(E).|E|^2.\|\underline{R}\|)$ time as a whole.

(i) Since mergings of blocks in PL are executed at most deg(E) times, the loop repeats at most deg(E) times.

(ii) The number of pairs of the same occurrences in $R_{k_1}, ..., R_{k_s}$ is at most $\frac{1}{2}s(s - 1)$, that is, $O(s^2)$ ($\leq O(|E|^2)$).

(iii) For a subset W of U_ℓ, $\mathcal{F}(W,F_\ell)$ can be computed in $O(\|F_\ell\|) \leq O(\|\underline{R}\|)$ time [Beeri and Bernstein 79].

<u>Computation of the closure</u>

In the following we consider the case where E is sound under $\underline{R}$. For relational expression $E = (R_{k_1} \times ... \times R_{k_s})[Z \equiv \Delta][P = Q]$, let $F = F^{(1)}_{k_1} \cup ... \cup F^{(s)}_{k_s} \cup \{\emptyset \rightarrow Z\} \cup \{A \rightarrow L_A \mid L_A$ is in $\mathcal{E}(E)$ and A is in $L_A\}$, where $F^{(i)}_{k_i} = \{X^{(i)} \rightarrow Y^{(i)} \mid X \rightarrow Y$ is in $F_{k_i}\}$ for $1 \leq i \leq s$. Then we have the following lemma.

[Lemma4.9] $\mathcal{F}(X,E) = \mathcal{F}(X,F)$.

(Proof) Clearly all FDs in F are valid in E. Thus it holds that $\mathcal{F}(X,F) \subseteq \mathcal{F}(X,E)$. In order to prove the converse, we shall show that there is a database instance I of $\underline{R}$ such that E(I) does not satisfy X → A for any A in U - $\mathcal{F}(X,F)$. For simplicity, let S = $\mathcal{F}(X,F)$. Let μ_1 and μ_2 be deg(E)-tuples satisfying the following three conditions.

(1) μ_1 and μ_2 agree exactly in S columns.

(2) For all A and B in U, if A and B are in the same block in $\mathcal{E}(E)$, then $\mu_1[A] = \mu_1[B]$ and $\mu_2[A] = \mu_2[B]$, and otherwise $\mu_1[A] \neq \mu_1[B]$ and $\mu_2[A] \neq \mu_2[B]$.

(3) $\mu_1[Z] = \mu_2[Z] = \Delta$.

Since (a) S is a union of some of the blocks in $\mathcal{E}(E)$ by FDs $\{A \to L_A \mid L_A$ is in $\mathcal{E}(E)$ and A is in $L_A\}$ and (b) $\mathcal{E}(E)$ is compatible with respect to $Z \equiv \Delta$, there are such tuples μ_1 and μ_2. Let $I_{\mu_1} = \{r_1, \ldots, r_n\}$ and $I_{\mu_2} = \{r_1', \ldots, r_n'\}$ be the projection mappings of μ_1 and μ_2 with respect to E, respectively. Let $I = \{r_1 \cup r_1', \ldots, r_n \cup r_n'\}$. Then relation E(I) contains both μ_1 and μ_2 by conditions (2) and (3) above. Condition (1) means that E(I) does not satisfy X → A for any A in U - S. Moreover, I is a database instance of $\underline{R}$ by the following fact, whose proof is given in Appendix 3. Thus Lemma4.9 follows.

[Fact4.5] Each relation $r_\ell \cup r_\ell'$ in I satisfies F_ℓ. []

We have polynomial time algorithms for solving the soundness problem for E and for computing $\mathcal{F}(X,F)$. Thus we have the following theorem.

[Theorem4.7] Let $\underline{R} = \{\langle R_1,F_1\rangle, \ldots, \langle R_n,F_n\rangle\}$ be a database scheme, let E be a relational expression with no union, and let d be an FD. It can be determined in polynomial time whether d is valid in E over $\underline{R}$. []

4.3.5 A polynomial time algorithm (2)

Let $\underline{R} = \{\langle R_1, F_1 \cup M_1\rangle, \ldots, \langle R_n, F_n \cup M_n\rangle\}$ be a database scheme. For simplicity, we denote $U_i = \{1, \ldots, \deg(R_i)\}$ for $1 \leq i \leq n$. Let E be a relational expression and let $U = \{1, \ldots, \deg(E)\}$. Let X be a subset of U. Then we can define the dependency basis $\mathcal{M}(X,E)$ of X with respect to E over $\underline{R}$, that is, $\mathcal{M}(X,E)$ is a partition $\{L_1, \ldots, L_q\}$ of U such that (1) $X \twoheadrightarrow L_i$ is valid in E over $\underline{R}$ for $1 \leq i \leq q$ and (2) $X \twoheadrightarrow Y$ is valid in E over $\underline{R}$ if and only if the right-hand side Y coincides with a union of some of the blocks L_i.

In the following we show how to compute efficiently $\mathcal{F}(X,E)$ and $\mathcal{M}(X,E)$ for a subset X of U and a relational expression E such that (1) E contains neither union nor projection and (2) each R_i occurs at most once in E. And then we extend the result to the case where E contains projections. We assume without loss of generality that E is of the form $(R_1 \times \ldots \times R_n)[Z \equiv \Delta][P = Q]$.

Soundness test and computations of the equivalence class and the closure

Algorithm4.1 correctly computes $\mathcal{E}(E)$ by itself, as explained below. Let PL_{final} be the final partition and suppose that PL_{final} is compatible with respect to $Z \equiv \Delta$. Clearly if A and B are in the same block in PL_{final}, then DEQ A = B is valid in E over $\underline{R}$. Conversely, consider the projection mapping $I_\mu = \{r_1, \ldots, r_n\}$ defined in the proof of Lemma4.8. Since each R_i occurs once in E, each relation r_i consists of exactly one tuple, and thus r_i trivially satisfies $F_i \cup M_i$. Thus I_μ is a database instance of $\underline{R}$ such that $E(I_\mu)$ contains the tuple μ. Thus if A and B are in different blocks in PL_{final}, then $E(I_\mu)$ does not satisfy DEQ A = B, that is, DEQ A = B is not valid in E over $\underline{R}$. The soundness problem for E can be also solved by Algorithm4.1.

We consider the case where E is sound under $\underline{R}$. For relational expression $E = (R_1 \times \ldots \times R_n)[Z \equiv \Delta][P = Q]$, let $F = F_1^{(1)} \cup \ldots \cup F_n^{(n)} \cup \{\emptyset \rightarrow Z\} \cup \{A \rightarrow L_A \mid L_A$ is in $\mathcal{E}(E)$ and A is in $L_A\}$ and $M = M_1^{(1)} \cup \ldots \cup M_n^{(n)} \cup \{\emptyset \twoheadrightarrow U_1^{(1)}, \ldots, \emptyset \twoheadrightarrow U_n^{(n)}\}$. Then we have the following lemma.

[Lemma4.10] $\mathcal{F}(X,E) = \mathcal{F}(X,F \cup M)$.

(Proof) $\underline{\mathcal{F}(X,F \cup M) \subseteq \mathcal{F}(X,E)}$: Let $E' = (R_1 \times \ldots \times R_n)[Z \equiv \Delta]$ and $F' = F_1^{(1)} \cup \ldots \cup F_n^{(n)} \cup \{\emptyset \rightarrow Z\}$. It is easy to see that $F' \cup M$ is valid in E' over $\underline{R}$. Since $E(I) = E'[P = Q](I) \subseteq E'(I)$ for any database I of $\underline{R}$, if $E'(I)$ satisfies an FD d, then $E(I)$ also satisfies d. That is, all valid FDs in E' are also valid in E over $\underline{R}$. Furthermore, all FDs in $\{A \rightarrow L_A \mid L_A$ is in $\mathcal{E}(E)$ and A is in $L_A\}$ are valid in E over $\underline{R}$ by the definition of $\mathcal{E}(E)$. Thus all FDs implied by $F \cup M$ are valid in E over $\underline{R}$. Thus, $\mathcal{F}(X,F \cup M) \subseteq \mathcal{F}(X,E)$.

$\underline{\mathcal{F}(X,E) \subseteq \mathcal{F}(X,F \cup M)}$: We shall show that there is a database instance I of $\underline{R}$ such that $E(I)$ does not satisfy $X \rightarrow A$ for any A in $U - \mathcal{F}(X,F \cup M)$. For simplicity, let $S = \mathcal{F}(X,F \cup M)$. Consider the database $I = \{r_1 \cup r_1', \ldots, r_n \cup r_n'\}$ of $\underline{R}$ defined in the proof of Lemma4.9. We can show that each relation $r_i \cup r_i'$ satisfies all FDs implied by $F_i \cup M_i$ in the same way as the proof of Fact4.5. (Since each R_i occurs once in E, it suffices to consider only Case2 in the proof of Fact4.5.) Since each R_i occurs once in E, $r_i \cup r_i'$ consists of exactly two tuples. Here, by the completeness proof of a set of inference rules for FDs and MVDs in [Beeri et al 77], we can show that for any relation r consisting of two tuples and any sets F and M of FDs and MVDs, respectively, if r satisfies all FDs implied by $F \cup M$, then chase(r,M) satisfies $F \cup M$. Thus chase($r_i \cup r_i', M_i$) satisfies $F_i \cup M_i$, and thus $I' = \{$chase($r_1 \cup r_1', M_1$), ..., chase($r_n \cup r_n', M_n$)$\}$ is a database instance of $\underline{R}$. Since $E(I)$ contains the two tuples μ_1 and μ_2, $E(I')$ also contains these tuples. By the definition of μ_1 and μ_2, E(I′) does not satisfy $X \rightarrow A$ for any A in U - S. []

Computation of the dependency basis

We consider the case where E is sound under $\underline{R}$. Let $S = \mathcal{F}(X,E)$. Then it holds that $\mathcal{M}(X,E) = \mathcal{M}(S,E)$ and each A in S is a block in $\mathcal{M}(S,E)$ by itself. For relational expression $E = (R_1 \times \ldots \times R_n)[Z \equiv \Delta][P = Q]$, let $F' = F_1^{(1)} \cup \ldots \cup F_n^{(n)} \cup \{\emptyset \rightarrow Z\}$ and $M = M_1^{(1)} \cup \ldots \cup M_n^{(n)} \cup \{\emptyset \twoheadrightarrow U_1^{(1)}, \ldots, \emptyset \twoheadrightarrow U_n^{(n)}\}$. The following lemma states a simple process for computing $\mathcal{M}(S,E)$ from $\mathcal{M}(S,F' \cup M)$ (there is a known algorithm for computing the latter). The same process does not work for computing $\mathcal{M}(X,E)$ from $\mathcal{M}(X,F' \cup M)$. The latter does not have a sufficient information for FDs.

[Lemma4.11] $\mathcal{M}(S,E) - \{\{A\} \mid A \text{ is in } S\}$ can be obtained from $\mathcal{M}(S,F' \cup M) - \{\{A\} \mid A \text{ is in } S\}$ by the following process.

Let $DB = \mathcal{M}(S,F' \cup M) - \{\{A\} \mid A \text{ is in } S\}$. For each DEQ A = B in P = Q, merge two blocks L_A and L_B in DB such that L_A and L_B contain A and B, respectively. Repeat this step until no blocks can be merged anymore.

(Proof) Let $\mathcal{M}(S,F' \cup M) - \{\{A\} \mid A \text{ is in } S\} = \{L_1', \ldots, L_q'\}$. Let $DB_{final} = \{L_1, \ldots, L_p\}$ be the final partition obtained by the process above. It suffices to show that (i) $S \twoheadrightarrow L_i$ is valid in E over $\underline{R}$ for $1 \leq i \leq p$ and (ii) $S \twoheadrightarrow \overline{L_i}$ is not valid in E over $\underline{R}$ for any $\overline{L_i}$ such that $\emptyset \neq \overline{L_i} \subsetneq L_i$.

(i) Let $E' = (R_1 \times \ldots \times R_n)[Z \equiv \Delta]$. It is easy to see that $F' \cup M$ is valid in E′ over $\underline{R}$. Thus $S \twoheadrightarrow L_j'$ is valid in E′ over $\underline{R}$ for $1 \leq j \leq q$. We can show that $S \twoheadrightarrow L_i$ is valid in $E'[P = Q]$ (= E) over $\underline{R}$ for $1 \leq i \leq p$ by repeated applications of the following fact, whose proof is given in Appendix 3.

[Fact4.6] Let E_0 be a relational expression. If $X \twoheadrightarrow Y$ and $X \twoheadrightarrow W$ are valid in E_0 over $\underline{R}$, then $X \twoheadrightarrow YW$ is valid in $E_0[A = B]$ over $\underline{R}$, where Y and W contain A and B, respectively. []

(ii) Consider the database $I = \{r_1 \cup r_1', \ldots, r_n \cup r_n'\}$ of $\underline{R}$ defined in the proof of Lemma4.9. As stated in the proof of Lemma4.10, $I' = \{\mathrm{chase}(r_1 \cup r_1', M_1), \ldots, \mathrm{chase}(r_n \cup r_n', M_n)\}$ is a database instance of $\underline{R}$. For the database instance I', relation $E'(I')$ is of the form

$$\mu_1[S] \text{ X } \begin{Bmatrix} \mu_1[L_1'] \\ \mu_2[L_1'] \end{Bmatrix} \text{ X } \ldots \text{ X } \begin{Bmatrix} \mu_1[L_q'] \\ \mu_2[L_q'] \end{Bmatrix} .$$

Here the order of columns of μ_1 and μ_2 are rearranged. Suppose that there is a DEQ $A_1 = A_2$ in $P = Q$ such that L_1' contains A_1 and L_2' contains A_2. Then since $\mu_1[A] \neq \mu_2[A]$ for all A in $U - S$ by the definition of μ_1 and μ_2, relation $E'(I')[A_1 = A_2]$ (= $E'[A_1 = A_2](I')$) is of the form

$$\mu_1[S] \text{ X } \begin{Bmatrix} \mu_1[L_1'L_2'] \\ \mu_2[L_1'L_2'] \end{Bmatrix} \text{ X } \begin{Bmatrix} \mu_1[L_3'] \\ \mu_2[L_3'] \end{Bmatrix} \text{ X } \ldots \text{ X } \begin{Bmatrix} \mu_1[L_q'] \\ \mu_2[L_q'] \end{Bmatrix} .$$

Thus $E'[P = Q](I')$ (= $E(I')$) is of the form

$$\mu_1[S] \text{ X } \begin{Bmatrix} \mu_1[L_1] \\ \mu_2[L_1] \end{Bmatrix} \text{ X } \ldots \text{ X } \begin{Bmatrix} \mu_1[L_p] \\ \mu_2[L_p] \end{Bmatrix} .$$

Using the technique of [Beeri et al 77], we can show that $E(I')$ does not satisfy MVD $S \twoheadrightarrow \overline{L_i}$ for any $\overline{L_i}$ such that $\emptyset \neq \overline{L_i} \subsetneq L_i$. Thus DB_{final} coincides with $\mathcal{M}(S,E) - \{\{A\} \mid A \text{ is in } S\}$. []

Suppose that $X \subseteq V \subseteq U$. It is easy to see that $\mathcal{M}(X,E[V]) = \{L \cap V \mid L \text{ is in } \mathcal{M}(X,E)\}$. Evidently, $X \twoheadrightarrow L_i \cap V$ is valid in $E[V]$ over $\underline{R}$ for every block in $\mathcal{M}(X,E) - \{\{A\} \mid A \text{ is in } \mathcal{J}(X,E)\}$ [Fagin 77]. The database instance I′ of $\underline{R}$ in the proof of Lemma4.11 is an example which shows that E(I′)[V] does not satisfy $X \twoheadrightarrow \overline{L_i}$ for any $\overline{L_i}$ such that $\emptyset \neq \overline{L_i} \subsetneq L_i \cap V$.

As a summary of this section, we present an algorithm for computing $\mathcal{F}(X,E)$ and $\mathcal{M}(X,E)$ for a relational expression E (= $(R_1 \times \dots \times R_n)[Z \equiv \Delta][P = Q][V]$) and a subset X of V, which can be executed in polynomial time.

[Algorithm4.2]

input: $\underline{R} = \{R_1\langle m_1, F_1 \cup M_1\rangle, \dots, R_n\langle m_n, F_n \cup M_n\rangle\}$,

$E = (R_1 \times \dots \times R_n)[Z \equiv \Delta][P = Q][V]$, and

$X\ (\subseteq V)$.

output: $\mathcal{F}(X,E)$,

$\mathcal{M}(X,E) - \{\{A\} \mid A \text{ is in } \mathcal{F}(X,E)\}$.

method:

(1) For simplicity, let $E' = (R_1 \times \dots \times R_n)[Z \equiv \Delta][P = Q]$. Compute $\mathcal{E}(E')$ by the following steps (i) through (iii). Note that step (4) of Algorithm4.1 is not executed for E′, since each R_i occurs once in E′.

(i) Let PL = $\{\{1\}, \dots, \{\deg(E')\}\}$.

(ii) For each DEQ A = B in P = Q, merge two blocks L_A and L_B in PL such that L_A and L_B contain A and B, respectively. Repeat this step until no blocks can be merged anymore.

(iii) For each pair $A \equiv c$ and $B \equiv c$ with the same constant c in $Z \equiv \Delta$, merge two blocks L_A and L_B in PL such that L_A and L_B contain A and B, respectively. Repeat this step until no blocks can be merged anymore.

(2) Suppose that E′ (and also E) is sound under $\underline{R}$. Compute $\mathcal{F}(X, F \cup M)$ by a known method, where $F = F_1^{(1)} \cup \dots \cup F_n^{(n)} \cup \{\emptyset \rightarrow Z\} \cup \{A \rightarrow L_A \mid L_A \text{ is in } \mathcal{E}(E') \text{ and } A \text{ is in } L_A\}$ and $M = M_1^{(1)} \cup \dots \cup M_n^{(n)} \cup \{\emptyset \twoheadrightarrow U_1^{(1)}, \dots, \emptyset \twoheadrightarrow U_n^{(n)}\}$. Then $\mathcal{F}(X, F \cup M) \cap V$ coincides with $\mathcal{F}(X,E)$.

(3) Let $S = \mathcal{F}(X, F \cup M)$. Compute $\mathcal{M}(S, F' \cup M) - \{\{A\} \mid A \text{ is in } S\}$ by a known method, where $F' = F_1^{(1)} \cup \ldots \cup F_n^{(n)} \cup \{\emptyset \rightarrow Z\}$.

(4) Let $DB = \mathcal{M}(S, F' \cup M) - \{\{A\} \mid A \text{ is in } S\}$. For each DEQ $A = B$ in $P = Q$, merge two blocks L_A and L_B in DB such that L_A and L_B contain A and B, respectively. Repeat this step until no blocks can be merged anymore.

(5) Let DB_{final} be the final partition obtained in step (4). Then $\{L \cap V \mid L \text{ is in } DB_{final}\}$ coincides with $\mathcal{M}(X, E) - \{\{A\} \mid A \text{ is in } \mathcal{F}(X, E)\}$. []

Thus we have the folloing theorem.

[Theorem4.8] Let $\underline{R} = \{\langle R_1, F_1 \cup M_1\rangle, \ldots, \langle R_n, F_n \cup M_n\rangle\}$ be a database scheme, let E be a relational expression such that (1) E contains no union and (2) each R_i occurs at most once in E, and let d be an FD or MVD. It can be determined in polynomial time whether d is valid in E over $\underline{R}$. []

Since join $R_1 \bowtie R_2 \bowtie \ldots \bowtie R_n$ can be transformed into an expression of the form $(R_1 \times \ldots \times R_n)[P = Q][V]$ (in polynomial time) [Ullman 80], we have the following corollary of Theorem4.8.

[Corollary4.1] It can be determined in polynomial time whether a given FD or MVD is valid in $R_1 \bowtie R_2 \bowtie \ldots \bowtie R_n$ over $\underline{R}$. []

4.3.6 An NP-completeness result under finite domains

Let $\underline{R} = \{\langle R_1, F_1\rangle, \ldots, \langle R_n, F_n\rangle\}$ be a database scheme, let E be a relational expression, and let d be an FD. Let $S_k = \{c_1, \ldots, c_k\}$ be a set of k constants. Then d is said to be <u>k-valid</u> in E over $\underline{R}$ if for every database instance I of $\underline{R}$ consisting of S_k, E(I) satisfies d. In a special case where $E = R_i$, d is 2-valid in over $\underline{R}$ if and only if d is valid in E over $\underline{R}$ (that is, F_i implies d) [Sagiv 80]. By Theorem4.7, if E contains no union, then it can be determined in polynomial time whether d is valid in E over $\underline{R}$. However, it is NP-complete to determine whether d is not 2-valid in E over $\underline{R}$, as shown below. Note that if only one constant occurs in a database instance I of $\underline{R}$, then E(I) is empty or has only one tuple, and thus E(I) trivially satisfies any dependency d. That is, it is meaningless to consider whether d is 1-valid in E over $\underline{R}$. Thus, 2-validity is theoretically the simplest case in finite domains.

[Theorem4.9] Let $\underline{R} = \{\langle R_1,F_1\rangle, \ldots, \langle R_n,F_n\rangle\}$ be a database scheme, let E be a relational expression consisting only of selections, restrictions, and cross products, and let d be an FD. Let $S_2 = \{c_1, c_2\}$ be a set of 2 constants. It is NP-complete to determine whether d is not 2-valid in E over $\underline{R}$.

(Proof) By Lemma4.5, it can be determined in nondeterministic polynomial time whether d is not 2-valid in E over $\underline{R}$. We transform the 3-satisfiability problem into the problem of determining whether there is a database instance I of $\underline{R}$ consisting of S_2 such that E(I) is not empty. This problem can be transformed into the problem of determining whether d is not 2-valid in E over $\underline{R}$, as presented in the proof of Lemma4.6. Thus Theorem4.9 will follow.

Let $P = Q_1 \wedge \ldots \wedge Q_m$, where each clause Q_i contains exactly three literals. Let $x_1, \ldots, x_n$ be all variables occurring in P. We construct a database scheme $\underline{R}$ and a relational expression E consisting only of selections, restrictions, and cross products such that there is a database instance I of $\underline{R}$ consisting of S_2 such that E(I) is not empty if and only if P is satisfiable.

Let $\underline{R} = \{\langle R_1,\{123 \rightarrow 4\}\rangle, \ldots, \langle R_m,\{123 \rightarrow 4\}\rangle\}$, where $\deg(R_i) = 4$ for all i. For $S_2 = \{c_1, c_2\}$, we assume that c_1 corresponds to "true" and c_2

corresponds to "false". Let x_{i1}, x_{i2}, x_{i3} be the three variables occurring in Q_i. Let $\{\delta_{i1}^{(1)}, \delta_{i2}^{(1)}, \delta_{i3}^{(1)}\}, \ldots, \{\delta_{i1}^{(7)}, \delta_{i2}^{(7)}, \delta_{i3}^{(7)}\}$ be the seven truth assignments to $\{x_{i1}, x_{i2}, x_{i3}\}$ that make Q_i true, and let $\{\delta_{i1}^{(8)}, \delta_{i2}^{(8)}, \delta_{i3}^{(8)}\}$ be the truth assignment to $\{x_{i1}, x_{i2}, x_{i3}\}$ that makes Q_i false. Then we define

$r_i = \{\delta_{i1}^{(1)}\delta_{i2}^{(1)}\delta_{i3}^{(1)}c_1, \ldots, \delta_{i7}^{(7)}\delta_{i2}^{(7)}\delta_{i3}^{(7)}c_1, \delta_{i1}^{(8)}\delta_{i2}^{(8)}\delta_{i3}^{(8)}c_2\}$, and

$E_i = R_i[1234 \equiv \delta_{i1}^{(1)}\delta_{i2}^{(1)}\delta_{i3}^{(1)}c_1] \text{ X } \ldots \text{ X } R_i[1234 \equiv \delta_{i1}^{(7)}\delta_{i2}^{(7)}\delta_{i3}^{(7)}c_1] \text{ X } R_i[1234 \equiv \delta_{i1}^{(8)}\delta_{i2}^{(8)}\delta_{i3}^{(8)}c_2]$. Let $I_0 = \{r_1, \ldots, r_m\}$. Then we have the following fact, whose proof is given in Appendix 3.

[Fact4.7] Let $E_s = E_1 \text{ X } \ldots \text{ X } E_m$. Then I_0 is the only database instance of $\underline{R}$ consisting of S_2 such that $E_s(I_0)$ is not empty. []

Let $E_t = (R_1[4 \equiv c_1] \text{ X } \ldots \text{ X } R_m[4 \equiv c_1])[\psi]$, where $\psi = \psi_1 \cup \ldots \cup \psi_n$ is a set of DEQs that chooses exactly all tuples that are assignable to $\{x_1, \ldots, x_n\}$ from $r_1 \text{ X } \ldots \text{ X } r_m$, as presented in the proof of Lemma4.1. That is, ψ is defined as follows. Let the first, second, and third positions of variables of Q_i correspond to the first, second, and third columns of the occurrense of R_i in E_t, respectively. That is, let the position $3(i-1) + \ell$ in P correspond to the column number $4(i-1) + \ell$ in E_t, where $1 \leq i \leq m$ and $1 \leq \ell \leq 3$. For each x_j, if $p_1, \ldots, p_k$ are all the positions of x_j occurring in P, then $\psi_j = \{q_1 = q_2, \ldots, q_1 = q_k\}$, where $p_1, \ldots, p_k$ corresponds to $q_1, \ldots, q_k$, respectively. Then for each r_i in I_0, since $\delta_{i1}^{(1)}\delta_{i2}^{(1)}\delta_{i3}^{(1)}c_1, \ldots, \delta_{i1}^{(7)}\delta_{i2}^{(7)}\delta_{i3}^{(7)}c_1$ are all the tuples that have the constant c_1 in 4 column, it holds that $R_i[4 \equiv c_1](I_0) = \{\delta_{i1}^{(1)}\delta_{i2}^{(1)}\delta_{i3}^{(1)}c_1, \ldots, \delta_{i1}^{(7)}\delta_{i2}^{(7)}\delta_{i3}^{(7)}c_1\}$. The first three columns of thses tuples represent the seven truth assignments to $\{x_{i1}, x_{i2}, x_{i3}\}$ that make Q_i true. Thus we can show in the same way as the proof of Lemma4.1 that P is satisfiable if and only if $E_t(I_0)$ is not empty.

Now we consider relational expression $E = E_s \times E_t$, which consists only of selections, restrictions, and cross products. By Fact4.7 and the discussions above, P is satisfiable if and only if there is a database instance I of $\underline{R}$ consisting of S_2 such that E(I) is not empty. Since $\underline{R}$ and E can be constructed from P in polynomial time, Theorem4.9 has been proved. []

CHAPTER 5

CONCLUSION

In Chapter 2, we have shown that (a) it can be determined in polynomial time whether a given database scheme $\underline{R}$ over U is consistent and that (b) given a sebset V of U, we can construct a relational expression whose value is rep(I)[V-total] for every database instance of $\underline{R}$, provided that $\underline{R}$ is consistent. There are some remaining problems.

(1) Is it decidable whether given a universal relation scheme <U,F> and a decomposition $\{R_1, \ldots, R_n\}$ of U, a database scheme $\underline{R} = \{\langle R_1,F_1\rangle, \ldots, \langle R_n,F_n\rangle\}$ over U is consistent? Here, each F_i is a cover of $\{X \rightarrow Y \mid F$ implies $X \rightarrow Y$ and $XY \subseteq R_i\}$. In this thesis, we have assumed that a cover of F is equivalent to that of $F_1 \cup \ldots \cup F_n$. Then the decomposition is said to preserve F. However, it is possible that the decomposition does not preserve F, but $\underline{R}$ is consistent. We note that given a universal relation scheme <U,F> and a decomposition $\{R_1, \ldots, R_n\}$ of U, if the decomposition preserves F, then the database scheme $\underline{R} = \{\langle R_1,F_1\rangle, \ldots, \langle R_n,F_n\rangle\}$ can be computed in polynomial time, but if the decomposition does not preserve F, then finding a cover for $F_1 \cup \ldots \cup F_n$ is NP-complete [Beeri and Honeyman 81].

(2) For the representative instance, the notion of boundedness has been recently proposed [Ullman et al 82]. Intuitively, a database scheme $\underline{R} = \{\langle R_1,F_1\rangle, \ldots, \langle R_n,F_n\rangle\}$ over U is bounded if for every database instance I of $\underline{R}$ such that rep(I) satisfies $F_1 \cup \ldots \cup F_n$, any tuple of rep(I) can be obtained from $\text{aug}_U(I)$ by a fixed number of applications of FD-rules for $F_1 \cup \ldots \cup F_n$. If $\underline{R}$ is bounded, then rep(I)[V-total] for any subset V of U can be computed efficiently [Ullman et al 82]. Theorem2.2 implies that if $\underline{R}$ is consistent, then $\underline{R}$ is bounded. However, it is possible that even if $\underline{R}$ is

not consistent, $\underline{R}$ is bounded. Is it decidable whether a given inconsistent database scheme is bounded?

(3) In Chapter 2, we consider only FDs as constraints. We do not know how to define the consistency of a database scheme with FDs and full MVDs.

In Chapter 3, we have shown some restricted solutions on implication problem for FDs and embedded MVDs. The most attractive but difficult problem is to determine whether implication problem for embedded MVDs is solvable.

In Chapter 4, we have shown that both the view nonemptiness and the tuple membership problem are NP-complete, but if the given relational expression contains no projection, then the tuple membership problem can be solved in polynomial time. And then we have shown some results on implication problem for view dependencies. As for implication problem for view dependencies, we consider only FDs and full MVDs as constraints. The decidability result for this problem (Theorem4.4) can be extended to a class of FDs and TDs, such that any chase process of any tableau under a given set of FDs and TDs always terminates. For example, consider the relation scheme $<R,C_n[U_0]>$ in Section 3.3.2. Then any chase process of any tableau under $C_n[U_0]$ always terminates by Lemma3.12. Thus for a given database scheme $\underline{R}$ = $\{<R_1,D_1>, \ldots, <R_n,D_n>\}$, if each D_i is the same form as $C_n[U_0]$, then it is decidable whether a given FD or TD is valid in a given relational expression over $\underline{R}$. There are some remaining problems for the implication problem.

(1) In Section 4.3.3, we have shown an NP-hardness result for the problem (Theorem4.6). But we do not know whether it is NP-complete.

(2) In Section 4.3.5, we have shown a polynomial time algorithm for the problem in the case where the given relational expression E contains no union and each R_i occurs once in E. But we do not know whether the result still holds, if each R_i may occur twice in E. Note that if the number of occurrences of R_i is not bounded, then the problem is NP-hard by Theorem4.6.

APPENDIX 1

Proofs of Lemmas in Chapter 1

Proof of Lemma2.1: First we prove the following fact.

[FactA.1] If $\underline{R}$ is not consistent, then there is a database instance I of $\underline{R}$ such that a restricted confliction occurs by only restricted applications of FD-rules for F to $aug_U(I)$.

(Proof) There must be a database instance $I = \{r_1, \ldots, r_n\}$ of $\underline{R}$ such that a confliction occurs by a chase process of $aug_U(I)$ under F. Consider the chase process until the first confliction occurs. Suppose that the chase process consists of k non-restricted and a number of restricted applications of FD-rules for F. We assume that if the confliction is restricted, then it is considered as a restricted application of FD-rule for F, and otherwise it is considered as a non-restricted one. We prove FactA.1 by induction on the number k.

Basis: If k = 0, then FactA.1 follows trivially.

Induction: Suppose that a confliction occurs by a chase process that consists of k non-restricted and a number of restricted applications of FD-rules for F to $aug_U(I)$. Consider the first non-restricted application of FD-rule in the chase process, that is, suppose that $aug_U(I)$ is transformed into a relation r by only restricted applications of FD-rules for F and then FD-rule for an FD $X \rightarrow Y$ in F_j is applied to two tuples μ and ν of r, where neither μ nor ν is any extension of any tuple of $aug_U(r_j)$. Consider the chase of $r_j \cup \{\mu[R_j]\}$ under F_j. If it does not satisfy F_j, then a restricted confliction occurs by only restricted applications of FD-rules for F_j to μ and extensions of tuples of $aug_U(r_j)$, and thus FactA.1 follows.

Suppose that the chase of $r_j \cup \{\mu[R_j]\}$ under F_j satisfies F_j and let μ' be the tuple obtained by replacing all the variables of the final extension of $\mu[R_j]$ in the chase of $r_j \cup \{\mu[R_j]\}$ under F_j with distinct constants. Let $I' = \{r_1, \dots, r_j \cup \{\mu'\}, \dots, r_n\}$. We can obtain the relation r, especially the tuples μ and ν, by only restricted applications of FD-rules for F to $aug_U(I')$. Since FD-rules for F are only restrictedly applied to $aug_U(I)$, each variable occurs once in only one tuple of r, and thus $\mu[X] = \nu[X]$ implies that μ and ν have the same constants in X columns. Thus FD-rule for $X \to Y$ can be restrictedly applied to μ and $aug_U(\mu')$ (and also to ν and $aug_U(\mu')$) instead of the original non-restricted application to μ and ν. That is, the first non-restricted application of FD-rule is transformed into two restricted applications by adding a tuple to I. If a confliction occurs in this time, then the confliction is restricted, since $X \to Y$ is in F_j and μ' is in $r_j \cup \{\mu'\}$. Thus FactA.1 follows. Suppose that no confliction occurs. Then we can show that a confliction occurs by following the rest of the original chase process, as explained below.

Suppose that by the non-restricted application of FD-rule for $X \to Y$ to μ and ν, the resulting tuples have the same variable v in A column for an attribute A in Y and suppose that by the restricted applications of FD-rule for $X \to Y$ to μ and $aug_U(\mu')$ (and to ν and $aug_U(\mu')$), the resulting tuples have the same constant c in A column. If v is replaced with another variable v' (or v' is replaced with v) in the original chase process, then v' is replaced with the constant c in the new chase process. If v is replaced with a constant c' ($\neq$ c) in the original chase process, then a confliction occurs in the new chase process. In this case, a confliction occurs by the new chase process earlier than by the original one. Since the rest of the original chase process consists of k-1 non-restricted and a number of restricted applications of FD-rules for F, FactA.1 follows from the induction hypothesis. []

Suppose that $\underline{R}$ is not consistent. By FactA.1, there is a database instance $I = \{r_1, \ldots, r_n\}$ of $\underline{R}$ such that a restricted confliction occurs by only restricted applications of FD-rules for F to $aug_U(I)$. Suppose that an extension ν of a tuple ν_0 of $aug_U(r_i)$ restrictedly conflicts with an extension μ of a tuple μ_0 of $aug_U(r_j)$ for an FD $X \rightarrow Y$ in F_j. Since ν restrictedly conflicts with μ_0 as well as μ for $X \rightarrow Y$ and ν can be obtained by extending ν_0 by a number of restricted applications of FD-rules for F without changing any other tuple of $aug_U(I)$, Lemma2.1 follows.

Proof of Lemma2.4: First we prove the following fact.

[FactA.2] If $X \rightarrow Y$ is in F_j and $X \rightarrow V$ is implied by $F_j - \{X \rightarrow Y\}$, then there is a subset $\{Z_1 \rightarrow W_1, \ldots, Z_s \rightarrow W_s\}$ of $F_j - \{X \rightarrow Y\}$ such that (1) $V \subseteq XW_1 \ldots W_s$ and (2) $Z_tW_t \subsetneq XY$ and $Z_t \subseteq XW_1 \ldots W_{t-1}$ for $1 \leq t \leq s$.

(Proof) Suppose that $X \rightarrow V$ is implied by $F_j - \{X \rightarrow Y\}$. Then there is a subset $\{Z_1 \rightarrow W_1, \ldots, Z_s \rightarrow W_s\}$ of $F_j - \{X \rightarrow Y\}$ such that $V \subseteq XW_1 \ldots W_s$ and $Z_t \subseteq XW_1 \ldots W_{t-1}$ for $1 \leq t \leq s$ [Beeri and Bernstein 79]. Since $X \rightarrow W_1 \ldots W_s$ is implied by $\{Z_1 \rightarrow W_1, \ldots, Z_s \rightarrow W_s\}$, which is a proper subset of F_j, it follows from Assumptions 2.2(a) and 2.2(b) that $W_1 \ldots W_s \subsetneq XY$. Thus FactA.2 follows. []

In order to prove Lemma2.4, it suffices to show that $X_m \rightarrow A$ is not implied by $F_j - \{X_m \rightarrow Y_m\}$. Suppose that $X_m \rightarrow A$ is implied by $F_j - \{X_m \rightarrow Y_m\}$. By FactA.2, there is a subset $H = \{Z_1 \rightarrow W_1, \ldots, Z_s \rightarrow W_s\}$ of $F_j - \{X_m \rightarrow Y_m\}$ such that (1) W_s contains A and (2) $Z_tW_t \subsetneq X_mY_m$ and $Z_t \subseteq X_mW_1 \ldots W_{t-1}$ for $1 \leq t \leq s$. Let $H_j^{(m)}$ be the intersection of F_j and $\{X_1 \rightarrow Y_1, \ldots, X_{m-1} \rightarrow Y_{m-1}\}$. Since the derivation $X_1 \rightarrow Y_1, \ldots, X_m \rightarrow Y_m$ is close, cover($H_j^{(m)}$) must contain H. Since H implies $X_m \to A$, cover($H_j^{(m)}$) also implies $X_m \to A$. This, however, contradicts that $X_m \to Y_m$ is irreducible.

<u>Proof of Lemma2.5</u>: Suppose that EXAM(R_i) returns "yes". We denote the final values of S, G_1, ..., G_n by S', G_1', ..., G_n', respectively. Since $R_iY_1...Y_m \subseteq \mathcal{F}(R_i,F)$ and $S' = \mathcal{F}(R_i,F)$, it holds that $X_k \subseteq S'$ for $1 \leq k \leq m$, and thus $X_k \to Y_k$ is in $G_1' \cup ... \cup G_n'$. Suppose that $X_m \to Y_m$ is not selected in step (2-i). Then there is an FD $X^{(p)} \to Y^{(p)}$ in F_j such that $X_mY_m \subseteq X^{(p)}Y^{(p)}$ and that $X_m \to Y_m$ is added to G_j at the p-th exection of step (2-iii). Then it holds that (1) $X^{(p)} \subseteq S^{(p)}$, (2) $X^{(p)} \to Y^{(p)}$ is minimal in $F_j - G_j^{(p)}$, and (3) $X_m \to Y_m$ is in $F_j - G_j^{(p)}$. In the following we prove Lemma2.5 by induction on the number m.

<u>Basis</u>: Consider the case where m = 1. Then $X_m \subseteq R_i$ implies $X_m \subseteq S^{(p)}$, and thus $X^{(p)}Y^{(p)} \subseteq X_mY_m$ by the minimality of $X^{(p)} \to Y^{(p)}$. Since $X_mY_m \subseteq X^{(p)}Y^{(p)}$, it holds that $X^{(p)}Y^{(p)} = X_mY_m$. There is an attribute A in X_m such that $X^{(p)} \to A$ is not implied by $F_j - \{X^{(p)} \to Y^{(p)}\}$. Because if there is no such attribute A, then $X^{(p)} \to X_m$ is implied by $F_j - \{X^{(p)} \to Y^{(p)}\}$, and thus $X^{(p)} \to X_mY_m$ (= $X^{(p)}Y^{(p)}$) is implied by $X^{(p)} \to X_m$ and $X_m \to Y_m$. This, however, contradicts Assumpion2.2(b). Note that A is in $Y^{(p)}$. Since $X_m \subseteq S^{(p)}$ and X_m contains A, $S^{(p)}$ contains A. Since $G_j^{(p)}$ does not contain $X^{(p)} \to Y^{(p)}$, $\mathcal{F}(X^{(p)},G_j^{(p)})$ does not contain A. Thus it holds that $S^{(p)} \cap Y^{(p)} - \mathcal{F}(X^{(p)},G_j^{(p)}) \neq \emptyset$. Thus EXAM($R_i$) returns "no" by Condition1 at the p-th execution of step (2-ii).

<u>Induction</u>: If $X_m \subseteq S^{(p)}$, then EXAM(R_i) returns "no" by the same reason above. Suppose that $X_m - S^{(p)} \neq \emptyset$. Since $X_m \subseteq R_iY_1...Y_{m-1}$, there is an FD $X_k \to Y_k$ with $1 \leq k \leq m-1$ such that Y_k contains an attribute B in $X_m - S^{(p)}$. Note that $X_k \to Y_k$ is different from $X^{(p)} \to Y^{(p)}$ by the irreducibility of $X_m \to Y_m$. Let $X_k \to Y_k$ be the first FD in the derivation such that Y_k

contains B. Then subsequence $X_1 \rightarrow Y_1$, ..., $X_k \rightarrow Y_k$ is a close derivation of B from R_i such that the last FD $X_k \rightarrow Y_k$ is irreducible by the fact that none of Y_1, ..., Y_{k-1} contains B. Thus $X_k \rightarrow Y_k$ is selected in step (2-i) by the induction hypothesis. Suppose that $X_k \rightarrow Y_k$ is in F_ℓ and that $X_k \rightarrow Y_k$ is selected at the q-th execution of step (2-i). Since (1) $S^{(p)}$ does not contain B but $S^{(q+1)}$ contains B and (2) $X^{(p)} \rightarrow Y^{(p)}$ and $X_k \rightarrow Y_k$ are different, it holds that $p < q$. Thus $S^{(p+1)} \subseteq S^{(q)}$, and thus $S^{(q)}$ contains B. Since $G_\ell^{(q)}$ does not contain $X_k \rightarrow Y_k$ and $X_k \rightarrow Y_k$ is irreducible, $\mathcal{F}(X_k, G_\ell^{(q)})$ does not contain B by Lemma3.4. Thus it holds that $S^{(q)} \cap Y_k - \mathcal{F}(X_k, G_\ell^{(q)}) \neq \emptyset$. That is, EXAM($R_i$) returns "no" by Condition1 at the q-th execution of step (2-ii).

Proof of Lemmama2.6: It suffices to show that no variable in $aug_U(I)^*$ is replaced with any constant by applying any FD-rules for F to $aug_U(I)^*$. There are two cases to be considered.

Case1: Consider the case where for $aug_U(I)^*$, a variable is replaced with a constant by an application of FD-rule for an FD in F. That is, suppose that there are two tuples μ and ν of $aug_U(I)^*$ and an FD $X \rightarrow Y$ in F_j such that (1) μ and ν have the same constants in X columns, (2) μ has a variable in A column, and (3) ν has a constant c in A column, where A is an attribute in Y. Let μ' be a tuple over R_i that is obtained by replacing all the variables of $\mu[R_i]$ with distinct constants (that do not appear in any other tuple). Then relation $r_i \cup \{\mu'\}$ satisfies F_i. Because if it does not satisfy an FD $Z \rightarrow W$ in F_i, then there is a tuple τ of r_i that agrees with μ' in Z columns but does not agree with μ' in W columns. Then in $aug_U(I)^*$, FD-rule for $Z \rightarrow W$ must be restrictedly applied to μ and an extension of $aug_U(\tau)$, but this contradicts that no FD-rule can be restrictedly applied to $aug_U(I)^*$. Let $I' = \{r_1, ..., r_i \cup \{\mu'\}, ..., r_n\}$. Since $aug_U(I') =$

$\mathrm{aug}_U(I) \cup \mathrm{aug}_U(\mu')$, we can obtain $\mathrm{aug}_U(I)^* \cup \mathrm{aug}_U(\mu')$ by a chase process of $\mathrm{aug}_U(I')$ under F. Note that the relation contains the tuple ν. Since $\mu'[A] \neq c$, tuple ν conflicts with $\mathrm{aug}_U(\mu')$ for $X \to Y$. This, however, contradicts that $\underline{R}$ is consistent.

Case2: Consider the case where after a number of variables have been replaced with other variables by a number of applications of FD-rules in F to $\mathrm{aug}_U(I)^*$, a variable is replaced with a constant by an application of FD-rule for an FD in F. That is, suppose that we obtain a relation r by a number of non-restricted applications of FD-rules for F to $\mathrm{aug}_U(I)^*$, where each application replaces variables with other variables. Then suppose that there are two tuples τ and ν of r and an FD $X \to Y$ in F_j such that (1) τ and ν agree in X columns, (2) τ has a variable in A column, and (3) ν has a constant c in A column, where A is an attribute in Y. If τ and ν have the same constants in X columns, then this case is reduced to Case1 above. Suppose that τ and ν have variables in X columns. Since by the technique of the proof of Lemma2.1, each non-restricted application of FD-rule can be transformed into two restricted applications of the FD-rule by adding a tuple to I, we can obtain a database instance $\overline{I}$ of $\underline{R}$ (by adding some tuples to I) such that each non-restricted application of FD-rule in the chase process from $\mathrm{aug}_U(I)^*$ to r is replaced with two restricted applications of the FD-rule. Then $\mathrm{aug}_U(\overline{I})^*$ has the extensions of τ and ν that have the same constants in X columns. Thus this case is reduced to Case1 above.

Proof of Lemma2.7: Suppose that there is a chase process $\mu_0 \overset{X_1 \to Y_1}{\underset{\nu_1}{\Longrightarrow}} \cdots \overset{X_m \to Y_m}{\underset{\nu_m}{\Longrightarrow}} \mu_m$. Lemma2.7 follows from the following three facts (1), (2), and (3).

(1) Addition: Suppose that $X_k \to Y_k$ is in F_j and let $X \to Y$ be an FD in $\mathrm{cover}(X_k \to Y_k)$. Consider a derivation $X_1 \to Y_1, \ldots, X_m \to Y_m, X \to Y$ of V

from R_i. Since $\mu_{k-1} \xRightarrow[\nu_k]{X_k \to Y_k} \mu_k$ and $\underline{R}$ is consistent, tuple ν_k agrees with μ_m in X_kY_k columns, so in XY columns. Thus FD-rule for $X \to Y$ can be restrictedly applied to μ_m and ν_k. Clearly μ_m remains unchanged by the application. That is, there is a chase process $\mu_0 \xRightarrow[\nu_1]{X_1 \to Y_1} \cdots \xRightarrow[\nu_m]{X_m \to Y_m} \mu_m \xRightarrow[\nu_k]{X \to Y} \mu_m$.

(2) Deletion: Suppose that sequence $X_1 \to Y_1$, ..., $X_{k-1} \to Y_{k-1}$, $X_{k+1} \to Y_{k+1}$, ..., $X_m \to Y_m$ is a derivation of V from R_i. Then clearly there is a chase process $\mu_0 \xRightarrow[\nu_1]{X_1 \to Y_1} \cdots \xRightarrow[\nu_{k-1}]{X_{k-1} \to Y_{k-1}} \mu_{k-1} \xRightarrow[\nu_{k+1}]{X_{k+1} \to Y_{k+1}} \mu_k' \cdots \xRightarrow[\nu_m]{X_m \to Y_m} \mu_{m-1}'$ such that μ_{m-1}' agrees with μ_m in V columns.

(3) Exchange: Suppose that sequence $X_1 \to Y_1$, ..., $X_{k+1} \to Y_{k+1}$, $X_k \to Y_k$, ..., $X_m \to Y_m$ is a derivation of V from R_i and that $X_k \to Y_k$ and $X_{k+1} \to Y_{k+1}$ are in F_j and F_ℓ, respectively. Since (i) $X_{k+1} \subseteq R_iY_1 \ldots Y_{k-1}$, (ii) μ_k' agrees with μ_{k-1} in $R_iY_1 \ldots Y_{k-1}$ columns, and (iii) $\mu_k \xRightarrow[\nu_{k+1}]{X_{k+1} \to Y_{k+1}} \mu_{k+1}$, tuple ν_{k+1} agrees with μ_{k-1} in X_{k+1} columns, so $\mu_{k-1} \xRightarrow[\nu_{k+1}]{X_{k+1} \to Y_{k+1}} \mu'$. Since $X_k \subseteq R_iY_1 \ldots Y_{k-1}$ and $\mu_{k-1} \xRightarrow[\nu_k]{X_k \to Y_k} \mu_k$, tuple μ' agrees with ν_k in X_k columns, so $\mu' \xRightarrow[\nu_k]{X_k \to Y_k} \mu''$. Clearly μ'' coincides with μ_{k+1}. Thus there is a chase process $\mu_0 \xRightarrow[\nu_1]{X_1 \to Y_1} \cdots \mu_{k-1} \xRightarrow[\nu_{k+1}]{X_{k+1} \to Y_{k+1}} \mu' \xRightarrow[\nu_k]{X_k \to Y_k} \mu_{k+1} \xRightarrow[\nu_{k+2}]{X_{k+2} \to Y_{k+2}} \cdots \xRightarrow[\nu_m]{X_m \to Y_m} \mu_m$.

Proof of Lemma2.8: It suffices to show that for all $Z_t \to W_t$ with $1 \leqq t \leqq s$, there is an FD $X_k \to Y_k$ with $1 \leqq k \leqq m$ such that $Z_tW_t \subseteq X_kY_k$. Because if so, then the derivation $X_1 \to Y_1$, ..., $X_m \to Y_m$ is transformed into a derivation $X_1 \to Y_1$, ..., $X_m \to Y_m$, $Z_1 \to W_1$, ..., $Z_s \to W_s$ by s addition operations, and then it is transformed into a derivation $Z_1 \to W_1$, ..., $Z_s \to W_s$, $X_1 \to Y_1$, ..., $X_m \to Y_m$ by a number of exchange operations, and finally it is transformed into the minimal derivation $Z_1 \to W_1$, ..., $Z_s \to W_s$ by m deletion operations. First we prove the following fact.

[FactA.3] For the last FD $Z_s \rightarrow W_s$, there is an FD $X_k \rightarrow Y_k$ with $1 \leq k \leq m$ such that $Z_sW_s \subseteq X_kY_k$.

(Proof) Suppose that there is no FD $X_k \rightarrow Y_k$ such that $Z_sW_s \subseteq X_kY_k$. Then we will show that $\underline{R}$ is not consistent. Suppose that $Z_s \rightarrow W_s$ is in F_j. There is an attribute A in $V \cap W_s$ such that $Z_s \rightarrow A$ is not implied by $F_j - \{Z_s \rightarrow W_s\}$. Because if there is no such attribute A, then $Z_s \rightarrow V \cap W_s$ is implied by $F_j - \{Z_s \rightarrow W_s\}$, and thus by FactA.2 in the proof of Lemma2.4, the derivation $Z_1 \rightarrow W_1, \ldots, Z_s \rightarrow W_s$ can not be minimal. Sequence $X_1 \rightarrow Y_1, \ldots, X_m \rightarrow Y_m, Z_1 \rightarrow W_1, \ldots, Z_s \rightarrow W_s$ is a derivation of A from R_i such that $R_iY_1\ldots Y_m$ contains A, since $V \subseteq R_iY_1\ldots Y_m$. Let H_j be the intersection of F_j and $\{X_1 \rightarrow Y_1, \ldots, X_m \rightarrow Y_m, Z_1 \rightarrow W_1, \ldots, Z_{s-1} \rightarrow W_{s-1}\}$. Since the derivation $Z_1 \rightarrow W_1, \ldots, Z_s \rightarrow W_s$ is minimal, there is no FD $Z_t \rightarrow W_t$ with $1 \leq t \leq s-1$ such that $Z_sW_s \subseteq Z_tW_t$. And there is no FD $X_k \rightarrow Y_k$ with $1 \leq k \leq m$ such that $Z_sW_s \subseteq X_kY_k$ by the assumption. Thus cover(H_j) does not contain $Z_s \rightarrow W_s$. Since $Z_s \rightarrow A$ is not implied by $F_j - \{Z_s \rightarrow W_s\}$, $Z_s \rightarrow W_s$ is irreducible. Thus $\underline{R}$ is not consistent by Lemma2.3. [ ]

The minimality of the derivation $Z_1 \rightarrow W_1, \ldots, Z_s \rightarrow W_s$ of V from R_i implies that subsequence $Z_1 \rightarrow W_1, \ldots, Z_{s-1} \rightarrow W_{s-1}$ is a minimal derivation of $Z_s(V - W_s)$ from R_i. Since $V \subseteq R_iY_1\ldots Y_m$ and $Z_sW_s \subseteq X_kY_k$ for an FD $X_k \rightarrow Y_k$ by FactA.3, sequence $X_1 \rightarrow Y_1, \ldots, X_m \rightarrow Y_m$ is a derivation of $Z_s(V - W_s)$ from R_i. By FactA.3 there is an FD $X_\ell \rightarrow Y_\ell$ such that $Z_{s-1}W_{s-1} \subseteq X_\ell Y_\ell$. In general, for each FD $Z_t \rightarrow W_t$, there is an FD $X_k \rightarrow Y_k$ such that $Z_tW_t \subseteq X_kY_k$. Thus Lemma2.8 has been proved.

<u>Proof of Lemma2.10</u>: Suppose that $Z_tW_t \rightarrow V$ is implied by cover(H) $\cup$ F_i. Let H' be a subset of cover(H) $\cup$ F_i that implies $Z_tW_t \rightarrow V$ and assume that no

proper subset of H' implies $Z_tW_t \to V$. We denote $H' = \{X_1 \to Y_1, \ldots, X_m \to Y_m\}$, where $X_k \to Y_k$ is in F_{p_k} for $1 \leqq k \leqq m$. Then H' implies $Z_tW_t \to Y_1\ldots Y_m$ and $V \subseteq Z_tW_tY_1\ldots Y_m$. H' is disjoint from F_{j_t}. Because if an FD $X \to Y$ in F_{j_t} is in H', then H' implies $Z_tW_t \to XY$. Then $Z_tW_t \to XY$ and $Z_t \to W_t$ implies $Z_t \to XY$, and thus $XY \subseteq Z_tW_t$ by Assumption2.2(a). Thus $H' - \{X \to Y\}$ would imply $Z_tW_t \to V$, a contradiction of the minimality of H'. Since H' implies $Z_tW_t \to V$ and $Z_tW_t \subseteq R_{j_t}$, there is a derivation of V from R_{j_t} consisting of the FDs in H'. Thus E contains a term $E_{j_t}[V]$. We assume without loss of generality that $X_1 \to Y_1, \ldots, X_m \to Y_m$ is a derivation of V from R_{j_t}. By the following four facts, $E_{j_t}[V]$ includes $E_i[V]$, that is, E is equivalent to the expression obtained by removing the term $E_i[V]$ from E.

(1) By Lemmas 2.7 and 2.8, $E_{j_t}[V]$ includes $(R_{j_t} \bowtie R_{p_1}[X_1Y_1] \bowtie \ldots \bowtie R_{p_m}[X_mY_m])[V]$.

(2) Since H' implies not only $Z_tW_t \to V$ but also $R_{j_t} \to V$, $(R_{j_t} \bowtie R_{p_1}[X_1Y_1] \bowtie \ldots \bowtie R_{p_m}[X_mY_m])[V]$ is equivalent to $(R_{j_t}[Z_tW_t] \bowtie R_{p_1}[X_1Y_1] \bowtie \ldots \bowtie R_{p_m}[X_mY_m])[V]$.

(3) Let $H_1 = H' \cap \mathrm{cover}(H)$ and $H_2 = H' \cap F_i$. Note that H_1 and H_2 are disjoint and $H' = H_1 \cup H_2$. We denote $H_1 = \{P_1 \to Q_1, \ldots, P_u \to Q_u\}$ and $H_2 = \{S_1 \to T_1, \ldots, S_v \to T_v\}$, where $P_\ell \to Q_\ell$ for $1 \leqq \ell \leqq u$ is in F_{q_ℓ}. By the fact that $H_1 \subseteq \mathrm{cover}(H)$, the derivation $Z_1 \to W_1, \ldots, Z_s \to W_s$ is transformed into a derivation $Z_1 \to W_1, \ldots, Z_s \to W_s, P_1 \to Q_1, \ldots, P_u \to Q_u$ by u addition operations. By Lemma2.7, $(R_i \bowtie R_{j_1}[Z_1W_1] \bowtie \ldots \bowtie R_{j_s}[Z_sW_s] \bowtie R_{q_1}[P_1Q_1] \bowtie \ldots \bowtie R_{q_u}[P_uQ_u])[V]$ includes $E_i[V]$. Since expression R_i is equivalent to $R_i \bowtie R_i[S_1T_1] \bowtie \ldots \bowtie R_i[S_vT_v]$, expression $R_i \bowtie R_{j_1}[Z_1W_1] \bowtie \ldots \bowtie R_{j_s}[Z_sW_s] \bowtie R_{q_1}[P_1Q_1] \bowtie \ldots \bowtie R_{q_u}[P_uQ_u]$ is equivalent to $R_i \bowtie R_i[S_1T_1] \bowtie \ldots \bowtie R_i[S_vT_v] \bowtie R_{j_1}[Z_1W_1] \bowtie \ldots \bowtie R_{j_s}[Z_sW_s] \bowtie R_{q_1}[P_1Q_1] \bowtie \ldots \bowtie R_{q_u}[P_uQ_u]$, which is transformed into expression $R_i \bowtie R_{j_1}[Z_1W_1] \bowtie \ldots \bowtie R_{j_{t-1}}[Z_{t-1}W_{t-1}] \bowtie R_{j_{t+1}}[Z_{t+1}W_{t+1}] \bowtie \ldots \bowtie R_{j_s}[Z_sW_s] \bowtie R_{j_t}[Z_tW_t] \bowtie R_{p_1}[X_1Y_1] \bowtie \ldots \bowtie R_{p_m}[X_mY_m]$, denoted $\overline{E}$, by a permutation of

the join sequence. Note that join operation is commutative and associative. Thus $\bar{E}[V]$ includes $E_i[V]$.

(4) Since $R_{j_t}[Z_tW_t] \bowtie R_{p_1}[X_1Y_1] \bowtie \dots \bowtie R_{p_m}[X_mY_m]$ is a subexpression of $\bar{E}$, expression $(R_{j_t}[Z_tW_t] \bowtie R_{p_1}[X_1Y_1] \bowtie \dots \bowtie R_{p_m}[X_mY_m])[V]$ includes $\bar{E}[V]$.

Proof of Lemma2.11: Suppose that E'_{final} contains a term $E_i[V]$ and that E_i is of the form $R_i \bowtie R_{j_1}[Z_1W_1] \bowtie \dots \bowtie R_{j_s}[Z_sW_s]$. Let $H = \{Z_1 \to W_1, \dots, Z_s \to W_s\}$. We show that there is a database instance I of $\underline{R}$ such that $E_i[V](I)$ contains a tuple μ and no other $E_j[V](I)$ in $E'_{final}(I)$ contains the tuple μ. We define $I = \{r_1, \dots, r_n\}$ as follows. (We can show that I is a database instance of $\underline{R}$ in the same way as the proof of Lemma2.3.)

(1) r_i consists of only one tuple that has a constant c in all the columns.

(2) For $1 \leqq j \leqq s$ with $j \neq i$, let $\{P_1 \to Q_1, \dots, P_p \to Q_p\}$ be the intersection of F_j and $\{Z_1 \to W_1, \dots, Z_s \to W_s\}$. Then let $r_j = \{\mu_1, \dots, \mu_p\}$, where each tuple μ_q for $1 \leqq q \leqq p$ has the constant c in P_qQ_q columns and distinct constants in all other columns.

Let ν_0 be a tuple of $aug_U(r_i)$. Then there is a chase process $\nu_0 \xRightarrow{Z_1 \to W_1} \nu_1 \xRightarrow{Z_2 \to W_2} \dots \xRightarrow{Z_s \to W_s} \nu_s$ such that ν_s has the constant c exactly in $R_iW_1 \dots W_s$ columns. Thus $E_i[V](I)$ contains a tuple μ that has the constant c in all the columns. Suppose that $E_j[V](I)$ with $j \neq i$ contains the tuple μ, that is, there is a chase process $\tau_0 \xRightarrow[\delta_1]{X_1 \to Y_1} \tau_1 \xRightarrow[\delta_2]{X_2 \to Y_2} \dots \xRightarrow[\delta_m]{X_m \to Y_m} \tau_m$ such that τ_m agrees with μ in V columns, where τ_0 is in $aug_U(r_j)$. Then τ_0 has the constant c exactly in Z_tW_t columns for some t such that $Z_t \to W_t$ is in F_j. Suppose that $X_k \to Y_k$ is in F_{p_k} for $1 \leqq k \leqq m$. If $p_k = i$, then $X_k \to Y_k$ is in F_i and δ_k has the constant c exactly in R_i columns, and otherwise δ_k has the constant c exactly in $Z_{q_k}W_{q_k}$ columns for some q_k.

By the fact that each constant except c occurs at most once in the database I, $\tau_0[X_1] = \delta_1[X_1]$ implies that δ_1 has the constant c in X_1 columns, and thus $X_1 \subseteq Z_tW_t$. If $p_1 = i$, then $X_1 \rightarrow Y_1$ is in F_i. If $p_1 \neq i$, then it holds that $X_1 \subseteq Z_{q_1}W_{q_1}$, and thus $Z_{q_1} \rightarrow W_{q_1}$ and $X_1 \rightarrow Y_1$ implies $Z_{q_1} \rightarrow W_{q_1}X_1Y_1$. It follows from Assumption2.2(a) that $X_1Y_1 \subseteq Z_{q_1}W_{q_1}$. Thus $X_1 \rightarrow Y_1$ is in cover(H). That is, $X_1 \rightarrow Y_1$ is in cover(H) $\cup$ F_i. In general, it holds that $X_k \subseteq Z_tW_tY_1 \ldots Y_{k-1}$ and $X_k \rightarrow Y_k$ is in cover(H) $\cup$ F_i. That is, cover(H) $\cup$ F_i implies $Z_tW_t \rightarrow Y_1 \ldots Y_m$, so cover(H) $\cup$ F_i implies $Z_tW_t \rightarrow V$.

The fact that E'_{final} contains the term $E_i[V]$ implies that E'_{final} does not contain the term $E_j[V]$ by step (3) of Algorithm2.4. Thus E'_{final} contains no redundant union. Since the derivation $Z_1 \rightarrow W_1$, ..., $Z_s \rightarrow W_s$ is minimal, all the joins in E_i are necessary in order to extend ν_0 to ν_s. Thus E' contains no redundant join.

APPENDIX 2

Proof of Lemma3.6 in Chapter 3

First we present a new inference rule for MVDs, called Rule1.

Rule1: $\{X \to\to Y|Z,\ XY \to\to W(V),\ XZ \to\to W(V)\}$ implies $X \to\to W(V)$.

Let $Y_1 = Y \cap W$, $Y_2 = Y - Y_1$, $Z_1 = Z \cap W$, $Z_2 = Z - Z_1$, $U_1 = W - YZ$, and $U_2 = V - XYZW$. Then Rule1 is restated as "$\{X \to\to Y_1Y_2|Z_1Z_2,\ XY_1Y_2 \to\to Z_1U_1|Z_2U_2,\ XZ_1Z_2 \to\to Y_1U_1|Y_2U_2\}$ implies $X \to\to Y_1Z_1U_1|Y_2Z_2U_2$." The validity of Rule1 follows from Figure A.1 below. In Figure A.1, we use another inference rules for MVDs: Rule2 [Delobel 78] [Tanaka et al 79], Projection, and Augmentation [Fagin 77] [Zaniolo 76].

Rule2: $\{X \to\to Y|Z,\ XY \to\to Z|W\}$ implies $X \to\to Z|YW$.

Projection: $X \to\to Y|ZW$ implies $X \to\to Y|Z$.

Augmentation: $X \to\to Y|ZW$ implies $XZ \to\to Y|W$.

In order to prove Lemma3.6, it suffices to show that (1) D implies $X \to\to Q$ and (2) there is no nonempty proper subset Q' of Q such that D implies $X \to\to Q'$. Since $X \to\to P_1|\ldots|P_s$ implies $U - P_1P_2 \to\to P_1|P_2$ by Augmentation,

$$\left\{\begin{array}{l} U - P_1P_2 \to\to P_1|P_2 \\ U - P_2 \to\to Q(V) \\ U - P_1 \to\to Q(V) \end{array}\right\} \text{ implies } U - P_1P_2 \to\to Q(V) \text{ by Rule1.}$$

Since $X \to\to P_1|\ldots|P_s$ implies $U - P_1P_2P_3 \to\to P_1P_2|P_3$ by Augmentation and another inference rule for MVDs (Union) [Fagin 77] [Zaniolo 76],

$$\left\{\begin{array}{l} U - P_1P_2P_3 \to\to P_1P_2|P_3 \\ U - P_3 \to\to Q(V) \\ U - P_1P_2 \to\to Q(V) \end{array}\right\} \text{ implies } U - P_1P_2P_3 \to\to Q(V) \text{ by Rule1.}$$

By repeating this process, we have finally $U - P_1P_2\ldots P_s \to\to Q(V)$. Thus D implies $X \twoheadrightarrow Q(V)$. If there is a nonempty proper subset Q' such that D implies $X \twoheadrightarrow Q'(V)$, then D implies $\{U - P_1 \twoheadrightarrow Q'(V), \ldots, U - P_s \twoheadrightarrow Q'(V)\}$ by Augmentation. Thus for all i with $1 \leq i \leq s$, Q' is a union of some of the blocks in $\mathcal{M}(U - P_i,V,D)$. This, however, contradicts the minimality of Q.

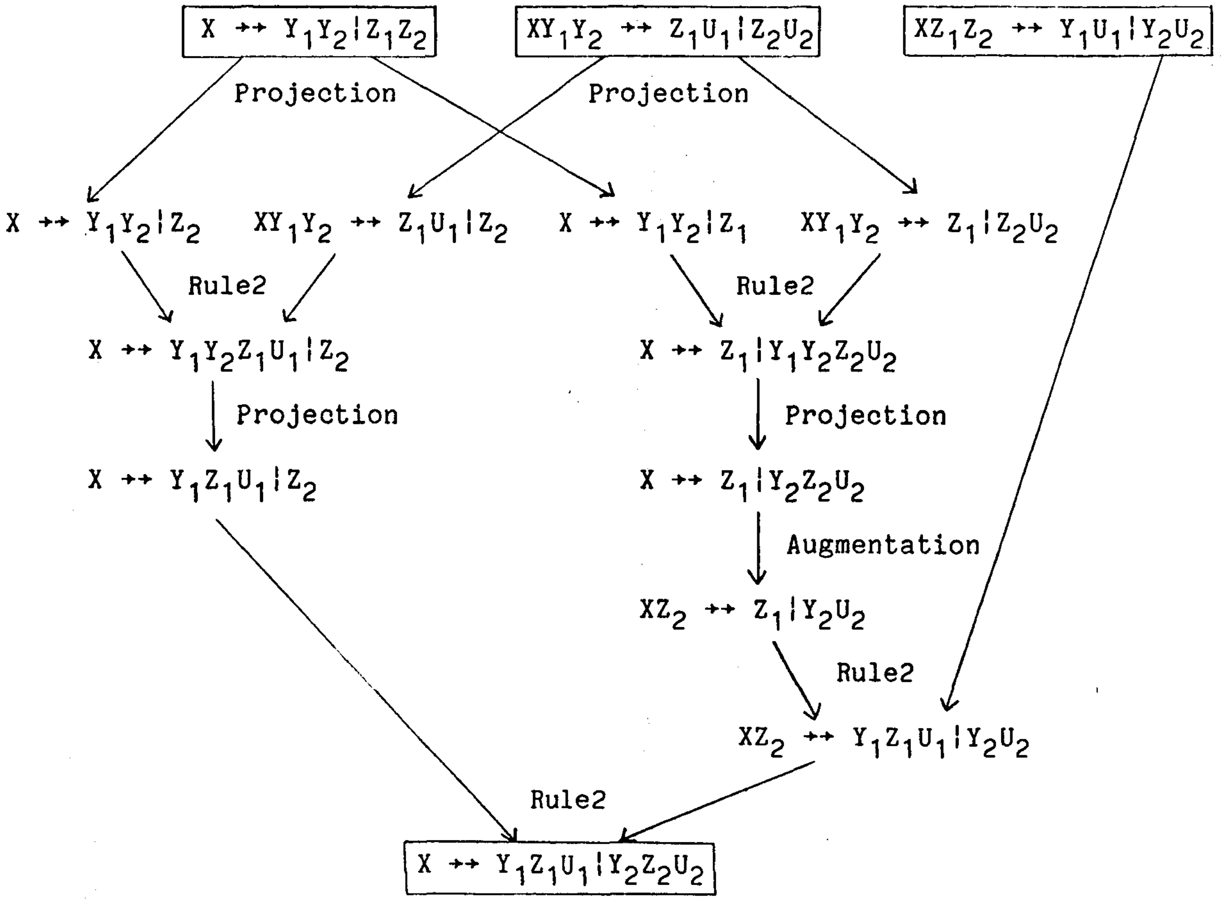
$X \twoheadrightarrow Y_1Y_2|Z_1Z_2$
$XY_1Y_2 \twoheadrightarrow Z_1U_1|Z_2U_2$
$XZ_1Z_2 \twoheadrightarrow Y_1U_1|Y_2U_2$
Projection
Projection
$X \twoheadrightarrow Y_1Y_2|Z_2$
$XY_1Y_2 \twoheadrightarrow Z_1U_1|Z_2$
$X \twoheadrightarrow Y_1Y_2|Z_1$
$XY_1Y_2 \twoheadrightarrow Z_1|Z_2U_2$
Rule2
Rule2
$X \twoheadrightarrow Y_1Y_2Z_1U_1|Z_2$
$X \twoheadrightarrow Z_1|Y_1Y_2Z_2U_2$
Projection
Projection
$X \twoheadrightarrow Y_1Z_1U_1|Z_2$
$X \twoheadrightarrow Z_1|Y_2Z_2U_2$
Augmentation
$XZ_2 \twoheadrightarrow Z_1|Y_2U_2$
Rule2
$XZ_2 \twoheadrightarrow Y_1Z_1U_1|Y_2U_2$
Rule2
$X \twoheadrightarrow Y_1Z_1U_1|Y_2Z_2U_2$

Figure A.1 Derivation of Rule1

APPENDIX 3

Proofs of Facts in Chapter 4

<u>Proof of Fact4.1</u>: It suffices to show the following two facts.

(1) For each ℓ with $1 \leq \ell \leq n$, chase($r'_\ell \cup r''_\ell, M_\ell$) satisfies F_ℓ.

(2) E(I') contains μ and ν but does not contain τ.

First we prove fact (1). Since $I = \{r_1, \ldots, r_n\}$ is a database instance of $\underline{R}$, r_ℓ satisfies $F_\ell \cup M_\ell$. Furthermore, since r_ℓ contains $r'_\ell \cup r''_\ell$, it follows from the definition of the chase that r_ℓ contains chase($r'_\ell \cup r''_\ell, M_\ell$). Since r_ℓ satisfies F_ℓ, chase($r'_\ell \cup r''_\ell, M_\ell$) satisfies F_ℓ. Next we prove fact (2). By the definition of $I_{\mu'}$ and $I_{\nu'}$, E($I_{\mu'}$) contains μ and E($I_{\nu'}$) contains ν. Thus E(I') contains μ and ν. Since r_ℓ contains chase($r'_\ell \cup r''_\ell, M_\ell$), E(I) contains E($I'$). Since E(I) does not contain τ by the assumption, E(I') does not contain τ.

<u>Proof of Fact4.2</u>: Since (1) for each of column numbers $X_1, \ldots, X_n, Z_1, \ldots, Z_m$, tuple μ_0 has a different constant c from all other tuples of r and (2) the left-hand side of each MVD in M contains at least one column number of $X_1, \ldots, X_n, Z_1, \ldots, Z_m$, it holds that chase(r,M) - $\{\mu_0\}$ = chase($\overline{r}$,M). Furthermore since all tuples of $\overline{r}$ have the same constant u in W column, all tuples of chase($\overline{r}$,M) also have the same constant u in W column. That is, chase($\overline{r}$,M) satisfies FD d: $Y_1 \ldots Y_m \rightarrow W$. By the fact that $\mu_0[Y_1 \ldots Y_m]$ = 1...1 and $\mu_0[W]$ = v, chase(r,M) (= chase($\overline{r}$,M) $\cup$ $\{\mu_0\}$) does not satisfy FD d if and only if chase($\overline{r}$,M) contains a tuple τ such that $\tau[Y_1 \ldots Y_m]$ = 1...1. Thus Fact4.2 follows.

<u>Proof of Fact4.3</u>: We show that if $\tau'[Y_i] = 1$, then for some k, $\tau'[Z_i] =$

a_{ik} and $\tau'[X_{i1}X_{i2}X_{i3}] = \nu_{ik}[X_{i1}X_{i2}X_{i3}]$ by induction on the number of applications of MVD-rules for M when computing chase($\overline{r}$,M) from $\overline{r}$. Note that $\{x_{i1}, x_{i2}, x_{i3}\} = \{\nu_{ik}[X_{i1}], \nu_{ik}[X_{i2}], \nu_{ik}[X_{i3}]\}$ makes Q_i true.

Basis: Obvious.

Induction: Let r' be a relation obtained by a number of applications of MVD-rules for M to r. Suppose that for every tuple τ' of r', if $\tau'[Y_i] = 1$, then $\tau'[Z_i] = a_{ik}$ and $\tau'[X_{i1}X_{i2}X_{i3}] = \nu_{ik}[X_{i1}X_{i2}X_{i3}]$ for some k. Consider an application of MVD-rule for an MVD in M to r'. There are two cases to be considered.

Case1: (An application of MVD-rule for MVD_j with $1 \leq j \leq n$) Suppose that we have a new tuple τ' by the application of MVD-rule for MVD_j: $Y_1 \ldots Y_m Z_1 \ldots Z_m \twoheadrightarrow X_j$ to μ and ν of r'.

Suppose that $\tau'[Y_i] = 1$. Since $\mu[Y_1 \ldots Y_m Z_1 \ldots Z_m] = \nu[Y_1 \ldots Y_m Z_1 \ldots Z_m] = \tau'[Y_1 \ldots Y_m Z_1 \ldots Z_m]$ by the definition of MVD-rules, it holds that $\mu[Y_i] = \nu[Y_i] = 1$ and $\mu[Z_i] = \nu[Z_i]$. Thus it follows from the induction hypothesis that $\mu[X_{i1}X_{i2}X_{i3}] = \nu[X_{i1}X_{i2}X_{i3}] = \nu_{ik}[X_{i1}X_{i2}X_{i3}]$ and $\mu[Z_i] = \nu[Z_i] = a_{ik}$, and thus $\tau'[X_{i1}X_{i2}X_{i3}] = \nu_{ik}[X_{i1}X_{i2}X_{i3}]$ and $\tau'[Z_i] = a_{ik}$ by the definition of MVD-rules. Thus Fact4.3 follows in this case.

Case2: (An application of MVD-rule for MVD_i with $n+1 \leq i \leq n+m$) Suppose that we have a new tuple τ' by the application of MVD-rule for MVD_i: $X_{i1}X_{i2}X_{i3} \twoheadrightarrow Y_i Z_i$ to μ and ν of r'.

Suppose that $\tau'[Y_j] = 1$. It follows from the definition of MVD-rules that if $i = j$, then $\tau'[X_{j1}X_{j2}X_{j3}Y_jZ_j] = \mu[X_{j1}X_{j2}X_{j3}Y_jZ_j]$, and otherwise $\tau'[X_{j1}X_{j2}X_{j3}Y_jZ_j] = \nu[X_{j1}X_{j2}X_{j3}Y_jZ_j]$. In both cases, Fact4.3 follows from the induction hypothesis.

Proof of Fact4.4: Suppose that F_ℓ implies an FD $Y \rightarrow A$ and that r_ℓ contains two tuples ν and τ such that $\nu[Y] = \tau[Y]$. It suffices to show that

$\nu[A] = \tau[A]$. Since r_ℓ contains two tuples ν and τ, there are two occurrences R_{k_i} and R_{k_j} of R_ℓ in E such that $\nu = \mu[U_{k_i}^{(i)}]$ and $\tau = \mu[U_{k_j}^{(j)}]$. Since $\nu[Y] = \tau[Y]$ implies $\mu[Y^{(i)}] = \mu[Y^{(j)}]$, $B^{(i)}$ and $B^{(j)}$ for each B in Y are in the same block in PL_{final} by the definition of μ. Thus it holds that $Y \subseteq W$, where W is the set defined in step (4-i) of Algorithm4.1. Since $Y \rightarrow A$ and $Y \subseteq W$ imply $W \rightarrow A$, A is in $\mathcal{F}(W,F_\ell)$. By step (4-iii) of Algorithm4.1, $A^{(i)}$ and $A^{(j)}$ must be in the same block in PL_{final}. Thus $\mu[A^{(i)}] = \mu[A^{(j)}]$ by the definition of μ, that is, $\nu[A] = \tau[A]$.

Proof of Fact4.5: Suppose that F_ℓ implies an FD $Y \rightarrow A$ and that $r_\ell \cup r_\ell'$ contains two tuples ν and τ such that $\nu[Y] = \tau[Y]$. It suffices to show that $\nu[A] = \tau[A]$. There are three cases to be considered.

Case1: Suppose that $\nu = \mu_1[U_{k_i}^{(i)}]$ and $\tau = \mu_1[U_{k_j}^{(j)}]$ $(i \neq j)$. Then $\nu[Y] = \tau[Y]$ implies $\mu_1[Y^{(i)}] = \mu_1[Y^{(j)}]$. Thus $\mu_1[A^{(i)}] = \mu_1[A^{(j)}]$ by Fact4.4, that is, $\nu[A] = \tau[A]$. Similarly, if $\nu = \mu_2[U_{k_i}^{(i)}]$ and $\tau = \mu_2[U_{k_j}^{(j)}]$, then $\nu[A] = \tau[A]$.

Case2: Suppose that $\nu = \mu_1[U_{k_i}^{(i)}]$ and $\tau = \mu_2[U_{k_i}^{(i)}]$. Then $\nu[Y] = \tau[Y]$ implies $\mu_1[Y^{(i)}] = \mu_2[Y^{(i)}]$. Thus $Y^{(i)} \subseteq S$ by the definition of μ_1 and μ_2. Since F_ℓ implies $Y \rightarrow A$, $F_\ell^{(i)}$ (and also F) implies $Y^{(i)} \rightarrow A^{(i)}$. Thus $Y^{(i)} \rightarrow A^{(i)}$ and $Y^{(i)} \subseteq S$ imply $S \rightarrow A^{(i)}$, and thus $\mathcal{F}(S,F)$ contains $A^{(i)}$. Since $S = \mathcal{F}(X,F) = \mathcal{F}(S,F)$ by the definition of closures, S contains $A^{(i)}$. Thus $\mu_1[A^{(i)}] = \mu_2[A^{(i)}]$ by the definition of μ_1 and μ_2, that is, $\nu[A] = \tau[A]$.

Case3: Suppose that $\nu = \mu_1[U_{k_i}^{(i)}]$ and $\tau = \mu_2[U_{k_j}^{(j)}]$ $(i \neq j)$. Then $\nu[Y] = \tau[Y]$ implies $\mu_1[Y^{(i)}] = \mu_2[Y^{(j)}]$. Thus $\mu_1[Y^{(i)}] = \mu_1[Y^{(j)}] = \mu_2[Y^{(i)}] = \mu_2[Y^{(j)}]$ by the definition of μ_1 and μ_2. Thus $\mu_1[A^{(i)}] = \mu_1[A^{(j)}] = \mu_2[A^{(i)}] = \mu_2[A^{(j)}]$ by Cases 1 and 2 above, that is, $\nu[A] = \tau[A]$.

Proof of Fact4.6: Let I be a database instance of $\underline{R}$. Let U = {1, ..., deg(E_0)} and let Z = U - YW. Suppose that E_0[A = B](I) contains two tuples μ and ν such that $\mu[X] = \nu[X]$, $\mu[YW] \neq \nu[YW]$ and $\mu[Z] \neq \nu[Z]$. It suffices to show that E_0[A = B](I) contains a tuple τ such that $\tau[X] = \mu[X] = \nu[X]$, $\tau[Z] = \mu[Z]$ and $\tau[YW] = \nu[YW]$, that is, E_0[A = B](I) satisfies X $\twoheadrightarrow$ YW.

Since $E_0[A = B](I) \subseteq E_0(I)$, relation E_0(I) contains μ and ν. Since X $\twoheadrightarrow$ Y and X $\twoheadrightarrow$ W imply X $\twoheadrightarrow$ Z by inference rules for MVDs [Beeri et al 77], E_0(I) satisfies X $\twoheadrightarrow$ Z, and thus E_0(I) contains a tuple τ such that $\tau[X] = \mu[X] = \nu[X]$, $\tau[Z] = \mu[Z]$ and $\tau[YW] = \nu[YW]$. Since tuple ν of E_0[A = B](I) satisfies DEQ A = B, tuple τ also satisfies DEQ A = B. Thus τ is in E_0[A = B](I).

Proof of Fact4.7: Since for no tuples μ and ν of r_i, μ and ν agree in 123 columns, r_i satisfies FD 123 → 4, and thus I_0 is a database instance of $\underline{R}$. Since r_i contains exactly all the tuples that are defined by VEQs appearing in E_i, $E_i(r_i) = \{\delta_{i1}^{(1)}\delta_{i2}^{(1)}\delta_{i3}^{(1)}c_1 \ldots \delta_{i1}^{(7)}\delta_{i2}^{(7)}\delta_{i3}^{(7)}c_1\delta_{i1}^{(8)}\delta_{i2}^{(8)}\delta_{i3}^{(8)}c_2\}$, and thus $E_s(I_0)$ (= $E_1(r_1)$ X ... X $E_m(r_m)$) is not empty.

Suppose that $I' = \{r_1', \ldots, r_m'\}$ is a database instance of $\underline{R}$ consisting of S_2 such that $E_s(I')$ is not empty. Since r_i' must contain all the tuples that are defined by VEQs appearing in E_i, r_i' contains r_i. If r_i' is different from r_i, then there is a tuple τ of r_i' that is not in r_i. Since r_i[123] contains all the possible eight tuples consisting of S_2 by the definition of r_i, there is a tuple τ' of r_i that agrees with τ in 123 columns. Since τ' is different from τ, τ' does not agree with τ in 4 column, and thus τ and τ' does not satisfy FD 123 → 4. This, however, contradicts that r_i' satisfies FD 123 → 4. Thus r_i' coincides with r_i. By the discussions above, I_0 is the only database instance of $\underline{R}$ such that $E_s(I_0)$ is not empty.

REFERENCES

[Aho et al 79]

Aho,A.V., Beeri,C. and Ullman,J.D. The theory of joins in relational databases. ACM Trans. Database Syst. 4, 3, Sept. 1979, pp.297-314.

[Armstrong 74]

Armstrong,W.W. Dependency structures of database relationships. Proc. IFIP 74, North Holland, 1974, pp.580-583.

[Beeri 79]

Beeri,C. On the role of data dependencies in the construction of relational database schemas. Technical Report TR-43, Dept. of Computer Science, Hebrew Univ., Jerusalem, Israel, Jan. 1979.

[Beeri 80]

Beeri,C. On the membership problem for multivalued dependencies in relational databases. ACM Trans. Database Syst. 5, 3, Sept. 1980, pp.241-259.

[Beeri and Bernstein 79]

Beeri,C. and Bernstein,P.A. Computational problems related to the design of normal form relational schemas. ACM Trans. Database Syst. 4, 1, Mar. 1979, pp.30-59.

[Beeri and Honeyman 81]

Beeri,C and Honeyman,P. Preserving functional dependencies. SIAM J. Comput. 10, 3, Aug. 1981, pp.647-656.

[Beeri et al 77]

Beeri,C, Fagin,R. and Howard,J.H. A complete axiomatization for functional and multivalued dependencies. Proc. ACM-SIGMOD Intern. Conf. on the Management of Data, Toronto, Canada, Aug. 1977, pp.47-61.

[Beeri et al 78]

Beeri,C., Bernstein,P.A. and Goodman,N. A sophisticate´s introduction to database normalization theory. Proc. 4th VLDB Conf., West Berlin, Germany, Sept. 1978, pp.113-124.

[Bernstein 76]

Bernstein,P.A. Synthesizing third normal form relations from functional dependencies. ACM Trans. Database Syst. 1, 4, Dec. 1976, pp.277-298.

[Codd 70]

Codd,E.F. A relational model of data for large shared data banks. Commun. ACM 13, 6, June 1970, pp.377-387.

[Codd 72]

Codd,E.F. Relational completeness of data base sublanguages. Data Base Systems, R.Rustin, ed., Prestice-Hall, 1972, pp.65-98.

[Delobel 78]

Delobel,C. Normalization and hierarchical dependencies in the relational data model. ACM Trans. Database Syst. 2, 3, Sept. 1978, pp.201-222.

[Fagin 77]

Fagin,R. Multivalued dependencies and a new normal forms for relational databases. ACM Trans. Database Syst. 2, 3, Sept. 1977, pp.262-278.

[Galil 82]

Galil,Z. An almost linear-time algorithm for computing a dependency basis in a relational database. J. ACM 29, 1, Jan. 1982, pp.96-102.

[Garey and Johnson 79]

Garey,M.R. and Johnson,D.S. Computers and Intractability -- A Guide to the Theory of NP-Completeness. Freeman, San Francisco, 1979.

[Graham and Yannakakis 82]

Graham,M.H. and Yannakakis,M. Independent database schemas (Extended abstruct). Proc. ACM Symposium on Principles of Database Systems, Mar. 1982, pp.199-204.

[Gurevich and Lewis 82]

Gurevich,Y. and Lewis,H.R. The inference problem for template dependencies. Proc. ACM Symposium on Principles of Database Systems, Mar. 1982, pp.221-229.

[Hagihara et al 79]

Hagihara,K., Ito,M., Taniguchi,K. and Kasami,T. Decision problems for multivalued dependencies in relational databases. SIAM J. Comput. 8, 2, May 1979, pp.247-264.

[Honeyman 80]

Honeyman,P. Extension joins. Proc. 6th VLDB Conf., 1980, pp.239-244.

[Honeyman 82]

Honeyman,P. Testing satisfaction of functional dependencies. J. ACM 29, 3, July 1982, pp.668-677.

[Ito et al 80]

Ito,M., Taniguchi,K. and Kasami,T. Inference of embedded multivalued dependencies in relational databases. IECE Japan J63-D, 9, Sept. 1980, pp.683-690 (in Japanese).

[Ito et al 81a]

Ito,M., Iwasaki,M., Taniguchi,K. and Kasami,T. Inference of dependencies for relational expressions. Tech. Rep. of Languages and Automata

Symposium, Kyoto, Feb. 1981 (in Japanese).

[Ito et al 81b]

Ito,M., Taniguchi,K. and Kasami,T. A result on implication problem for functional and template dependencies in relational databases. IECE Japan J64-D, 10, Oct. 1983, pp.919-926 (in Japanese).

[Ito et al 82]

Ito,M., Iwasaki,M., Taniguchi,K. and Kasami,T. Some decision problems on views in relational databases. IECE Japan J65-D, 6, June 1982, pp.790-796 (in Japanese).

[Ito et al 83a]

Ito,M., Taniguchi,K. and Kasami,T. Membership problem for embedded multivalued dependencies under some restricted conditions. Theoret. Comput. Sci. 22, 1983, pp.175-194.

[Ito et al 83b]

Ito,M., Iwasaki,M. and Kasami,T. An algorithm for testing consistency of a database scheme in relational databases. IECE Japan J66-D, 7, July 1983, pp.781-788 (in Japanese).

[Ito et al 83c]

Ito,M., Iwasaki,M., Taniguchi,K. and Kasami,T. Membership problems for dependencies on views in relational databases. to appear in IECE Japan J66-D (in Japanese).

[Iwasaki et al 82]

Iwasaki,M., Ito,M. Kasami,T. A result on the representative instance in relational databases. Tech. Rep. of Languages and Automata Symposium, Kyoto, Feb. 1982 (in Japanese).

[Kent 81]

Kent,W. Consequences of assuming a universal relation. ACM Trans. Database Syst. 6, 4, Dec. 1981, pp.539-556.

[Klug 80]

Klug,A. Calculating constraints on relational expressions. ACM Trans. Database Syst. 4, 4, Dec. 1980, pp.260-290.

[Klug and Price 82]

Klug,A., and Price,R. Determining view dependencies using tableaux. ACM Trans. Database Syst. 7, 3, Sept. 1982, pp.361-380.

[Maier et al 79]

Maier,D., Mendelzon,A.O. and Sagiv,Y. Testing implications of data dependencies. ACM Trans. Database Syst. 4, 4, Dec. 1979, pp.455-469.

[Maier et al 80]

Maier,D., Mendelzon,A.O., Sadri,F. and Ullman,J.D. Adequacy of decompositions of relational databases. J. Compt. System Sci. 21, 1980, pp.368-379.

[Parker and Parsaye-Ghomi 80]

Parker,D.S. and Parsaye-Ghomi,K. Inferences involving embedded multivalued dependencies and transitive dependencies. Proc. ACM-SIGMOD, Santa Monica, May 1980.

[Rissanen 77]

Rissanen,J. Independent components of relations. ACM Trans. Database Syst. 2, 4, Dec. 1977, pp.317-325.

[Sadri and Ullman 80]

Sadri,F. and Ullman,J.D. A complete axiomatization for a large class of dependencies in relational databases. Proc. ACM Symposium Theory of Computing, Los Angeles, Apr. 1980.

[Sadri and Ullman 82]

Sadri,F. and Ullman,J.D. The theory of functional and template depdencies. Theoret. Comput. Sci. 17, 1982, pp.317-331.

[Sagiv 80]

Sagiv,Y. An algorithm for inffering multivalued dependencies with an application to propositional logic. J. ACM 27, 2, Apr. 1980, pp.250-262.

[Sagiv 81]

Sagiv,Y. Can we use the universal instance assumption without using ulls? Proc. ACM SIGMOD Int. Conf. Management of Data, 1981, pp.108-120.

[Sagiv 83]

Sagiv,Y. A characterization of globally consistent databases and their correct access paths. ACM Trans. Database Syst. 8, 2, June 1983, pp.266-286.

[Sagiv and Walecka 82]

Sagiv,Y. and Walecka,S. Subset dependencies as an alternative to embedded multivalued dependencies. J. ACM 29, 1, 1982, pp.103-117.

[Smith and Chang 75]

Smith,J.M. and Chang,P.Y. Optimizing the performance of a relational algebra database interface. Commun. ACM 18, 10, Oct. 1975, pp.568-579.

[Tanaka et al 79]

Tanaka,K., Kambayashi,Y. and Yajima,S. Properties of embedded multivalued dependencies in relational databases. Trans. IECE Japan E-62, 8, Aug. 1979, pp.536-543.

[Ullman 80]

Ullman,J.D. Principles of Database Systems. Computer Science Press, 1980.

[Ullman 81]

Ullman,J.D. A view of directions in relational database theory. Lecture Notes in Computer Science (Automata, Languages and Programming), July 1981, pp.165-176.

[Ullman 82]

Ullman,J.D. The U.R. strikes back. Proc. ACM Symposium on Princeples of Database Systems, Mar. 1982, pp.10-22.

[Ullman et al 82]

Ullman,J.D., Vardi,M.Y. and Maier,D. The equivalence of universal relation definitions. Technical Report STAN-CS-82-940, Dept. of Computer Science, Stanford Univ., Oct. 1982.

[Vardi 82]

Vardi,M.Y. The implication and finite implication problems for typed template dependencies. Proc. ACM Symposium on Principles of Database Systems, Mar. 1982, pp.230-238.

[Vassiliou 80]

Vassiliou,Y. A formal treatment of imperfect information in database management. Technical Report CSRG-123, Univ. of Toronto, 1980.

[Zaniolo 76]

Zaniolo,C. Analysis and design of relational schemata for database systems. Doctoral dissertation, UCLA, 1976.