A Proof Procedure for Data Dependencies

CATRIEL BEERI AND MOSHE Y. VARDI

The Hebrew University of Jerusalem, Jerusalem, Israel

Abstract. A class of dependencies, tuple and equality generating dependencies, is defined, and the chase process is generalized to deal with these dependencies. For total dependencies the chase is an exponential time decision procedure for the implication problem, and in some restricted cases it can be modified to run in polynomial time. For nontotal dependencies the chase is only a proof procedure. However, several cases for which it is a decision procedure are shown. It is also shown that equality is redundant for deciding implication of tuple-generating dependencies, and is "almost redundant" for deciding implication of equality-generating dependencies.

Categories and Subject Descriptors: F.4.1. **[Mathematical Logic and Formal Languages]:** Mathematical Logic—*mechanical theorem proving;* H.2.1. **[Database Management]:** Logical Design—*schema and subschema*

General Terms: Design, Languages, Theory

Additional Key Words and Phrases: Relational database, data dependency, logical implication, proof procedure, decision procedure, solvability

1. *Introduction*

One of the important issues in the design of relational database schemas is the specification of the constraints that the data must satisfy to model correctly the part of the world under consideration. These constraints determine which databases are considered meaningful.

Of particular interest are the constraints called *data dependencies* or *dependencies* for short. The study of dependencies began with the *functional dependencies* in [21]. After the introduction of *multivalued dependencies* in [23] and [55], the field became chaotic for a few years in which researchers introduced many new classes of dependencies, for example, *mutual dependencies* [36], *join dependencies* [2, 42], *transitive dependencies* [38], *general dependencies* [41], and *subset dependencies* [46].

All these dependencies, however, have a similar semantics. Intuitively, the meaning of a dependency is that if some tuples, fulfilling certain equalities, exist in the database, then either some other tuples must also exist in the database, or some values in the given tuples must be equal. In this paper we introduce the class of *tuple-* and *equality-generating dependencies*, which seems to capture this intuitive semantics. We believe that this class includes most cases of interest. Our attempt

The research was partially supported by Grant 1849/79 of the U.S.A.–Israel Binational Science Foundation.

Authors' current addresses: C. Beeri, The Hebrew University of Jerusalem, Jerusalem, Israel; M. Y. Vardi, Center for Study of Language and Information, Stanford University, Ventura Hall, Stanford, CA 94305.

Permission to copy without fee all or part of this material is granted provided that the copies are not made or distributed for direct commercial advantage, the ACM copyright notice and the title of the publication and its date appear, and notice is given that copying is by permission of the Association for Computing Machinery. To copy otherwise, or to republish, requires a fee and/or specific permission.
© 1984 ACM 0004-5411/84/1000-0718 $00.75

here is at unification rather than generalization. It enables us to deal with two kinds of dependencies, rather than a large number of different classes of dependencies.

Most of the papers in dependency theory deal exclusively with various aspects of the *implication problem*, that is, the problem of deciding whether a given set of dependencies logically implies another dependency. The reason for the prominence of this problem is that an algorithm for deciding implication of dependencies enables us to decide whether two given sets of dependencies are equivalent or whether a given set of dependencies is redundant. A solution for the last two problems seems a significant step toward automated database schema design [10, 11, 16, 32, 51], which some researchers see as the ultimate goal for research in dependency theory [8].

A decision procedure[1] for the implication problem for functional and join dependencies called the *chase* was developed in [2] and [33]. This procedure is generalized in this paper to tuple-generating dependencies and equality-generating dependencies, and is shown to be a proof procedure for the implication problem for these dependencies.[2] In several cases, however, the chase is also a decision procedure, for example, if all dependencies are total. In some cases we also show how the chase may lead to an efficient decision procedure.

Tuple- and equality-generating dependencies can be expressed as first-order sentences [25, 37, 49]. Thus, it might be argued, there is no need to develop a proof procedure for dependencies, since any proof procedure for first-order logic will do. However, dependencies are just a fragment of first-order logic, a fragment that seems to be suitable for expressing integrity constraints of databases, and we would like to have a proof procedure that is specialized to this fragment. Our specialized proof procedure, the chase, turns out to be quite useful. For example, in this paper we use it to prove many decidability results, and in another paper [15] we use it to devise a formal system for dependencies.

The outline of the paper is as follows. In Section 2 we define the relational model, tableaux, total tuple-generating dependencies, and equality-generating dependencies. The implication problem and the chase as a decision procedure for implication of these dependencies are described in Section 3. We study the complexity of the chase and investigate the role of equality in the chase. It is shown that the result of the chase has a closure property that enables us to devise efficient tests for the implication of multivalued dependencies. In Section 4 we define general tuple-generating dependencies, for which the chase is a proof procedure. We point out several classes of dependencies for which the chase is a decision procedure. In Section 5, we conclude the paper by pointing out several possible generalizations.

2. *Basic Definitions*

2.1. ATTRIBUTES AND RELATIONS. *Attributes* are symbols taken from a given finite set $U = \{A_1, \ldots, A_n\}$ called the *universe*. All sets of attributes are subsets of U. We use the letters A, B, C, ... to denote single attributes, and X, Y, ... to denote sets of attributes. We do not distinguish between the attribute A and the set $\{A\}$. The union of X and Y is denoted by XY, and the complement of X in U is denoted by $\bar{X}$.

[1] We distinguish between a *decision procedure*, which always halts, and a *proof procedure*, which may run forever if the answer to the decision problem is negative.

[2] Similar generalizations were studied in [28], [40], [43], and [54].

With each attribute A is associated an infinite set, called its *domain*, denoted DOM(A), such that DOM(A) $\cap$ DOM(B) = $\emptyset$ for $A \neq B$. Let Dom = $\bigcup_{A \in U}$ DOM(A). For an attribute set X, an X-value is a mapping w: $X \to$ Dom, such that $w(A) \in$ DOM(A) for all $A \in X$. A *tuple* is a U-value. A *relation* is a set (not necessarily finite) of tuples. We use the letters t, u, w, ... to denote tuples, and I, J, ... to denote relations.

For a tuple w and a set $Y \subseteq U$, we denote the restriction of w to Y by $w[Y]$. For an attribute A, we do not distinguish between $w[A]$ (which is an A-value) and $w(A)$ (which is an element of DOM(A)). Let I be a relation. The set of X-values in I is $I[X] = \{w[X] \mid w \in I\}$. The set of values in I is VAL(I) = $\bigcup_{A \in U} I[A]$. For a tuple w, VAL(w) stands for VAL($\{w\}$).

2.2. TABLEAUX. A *valuation* is a mapping h: *Dom* $\to$ *Dom*, such that $a \in$ DOM(A) implies $h(a) \in$ DOM(A) for all $a \in$ Dom. The valuation h can be extended to tuples and relations as follows. Let w be a tuple. Then $h(w) = h \circ w$ ($\circ$ denotes functional composition). Let I be a relation. Then $h(I) = \{h(w) \mid w \in I\}$. Usually, we are interested only in a small subset of Dom, for example, VAL(I). We let h be undefined for other values, and say that h is a valuation on I.

Let I be a relation, and let h be a valuation on I. An *extension* of h to another relation I' is a valuation h' on I', which agrees with h on VAL(I). If h is the identity valuation on VAL(I), then we say that h is the identity on I.

A *tableau* [3] is a pair $T = \langle w, I \rangle$, where w is a tuple and I is a finite relation, such that VAL(w) $\subseteq$ VAL(I). With each tableau $T = \langle w, I \rangle$, we associate an operation on relations as follows: $T(J) = \{h(w) \mid h$ is a valuation such that $h(I) \subseteq J\}$. Namely, $T(J)$ is the set of images of w under all valuations that map every tuple of I to some tuple of J. Observe that $I \subseteq T(I)$, and that $I \subseteq J$ implies $T(I) \subseteq T(J)$ (monotonicity).

LEMMA 1 [15]. *Let $\langle u, I \rangle$ and $\langle v, J \rangle$ be tableaux. We can effectively construct a tableau $\langle w, K \rangle$ such that, for any relation L, $\langle u, I \rangle(\langle v, J \rangle(L)) = \langle w, K \rangle(L)$.*

COROLLARY. *Let $\langle u_1, I_1 \rangle, \ldots, \langle u_n, I_n \rangle$ be tableaux. We can effectively construct a tableau $\langle v, J \rangle$ such that, for any relation K, $\langle u_1, I_1 \rangle(\cdots(\langle u_n, I_n \rangle(K))\cdots) = \langle v, J \rangle(K)$.*

2.3. DEPENDENCIES. For any given application, only a subset of all possible relations is of interest. This subset is defined by constraints that are to be satisfied by the relations of interest. A class of constraints that has been extensively studied is the class of *dependencies*.

As an example, consider *functional dependencies* [21], and *multivalued dependencies* [23, 55]. A functional dependency (FD) is a statement $X \to Y$. It is satisfied by a relation I if for all tuples $t_1, t_2 \in I$, we have that if $t_1[X] = t_2[X]$, then $t_1[Y] = t_2[Y]$. A multivalued dependency (MVD) is a statement $X \twoheadrightarrow Y$. It is satisfied by a relation I if for all $t_1, t_2 \in I$, we have that if $t_1[X] = t_2[X]$, then there exists a tuple $t \in I$ such that $t[XY] = t_1[XY]$ and $t[XZ] = t_2[XZ]$, where $Z = \overline{XY}$.

Equality-generating dependencies generalize FDs. An equality-generating dependency (EGD) says that if some tuples, fulfilling certain equalities, exist in the database, then some values in these tuples must be equal. Formally, an EGD is a pair $\langle (a_1, a_2), I \rangle$, where a_1 and a_2 are A-values for some attribute A, and I is a finite relation, such that $a_1, a_2 \in I[A]$. We also call this EGD an A-EGD. A relation J satisfies $\langle (a_1, a_2), I \rangle$ if, for any valuation h such that $h(I) \subseteq J$, we have $h(a_1) = h(a_2)$. Note that, if $a_1 = a_2$, then $\langle (a_1, a_2), I \rangle$ is trivially satisfied by every relation.

Total tuple-generating dependencies[3] generalize MVDs. A total tuple generating dependency (TTGD) says that if some tuples, fulfilling certain equalities, exist in the database, then another tuple, whose values are taken from these tuples, must also exist in the database. Formally, a TTGD is represented by a tableau $T = \langle w, I \rangle$. A relation J satisfies T if, for any valuation h such that $h(I) \subseteq J$, we have that $h(w) \in J$. That is, J satisfies T if $T(J) = J$. Observe that we use T both as an operator on relations and as a dependency. The meaning should be clear from the context. Note that if $w \in I$, then $\langle w, I \rangle$ is trivially satisfied by every relation. In the sequel, we use D and E to denote finite sets of dependencies, and we use d and e to denote individual dependencies.

Example 1. Let $U = \{A, B, C, D\}$, DOM$(A) = \{a0, a1, \ldots\}$, DOM$(B) = \{b0, b1, \ldots\}$, etc. Let I and J be the relations:

I:

A	B	C	D
$a0$	$b0$	$c1$	$d0$
$a0$	$b1$	$c0$	$d1$

J:

A	B	C	D
$a0$	$b0$	$c0$	$d0$
$a1$	$b0$	$c0$	$d1$

Let u be the tuple:

A	B	C	D
$a0$	$b0$	$c0$	$d0$

Let d_1 be the TTGD $\langle u, I \rangle$. Then d_1 is equivalent to the MVD $A \twoheadrightarrow C$. Let d_2 be the EGD $\langle (a0, a1), J \rangle$. Then d_2 is equivalent to the FD $BC \rightarrow A$. □

The class of TTGDs and EGDs contains as special cases almost all classes of dependencies that were investigated in the literature, for example, functional, multivalued, join, and transitive dependencies. It is equivalent to the class of *generalized dependencies* of Grant and Jacobs [28]. (Their database model is, however, more general than ours. See Section 5.1.) It does not include, however, the classes of *embedded multivalued dependencies* [23] and *embedded join dependencies* [33]. We come back to this in Section 4.

3. *A Decision Procedure for Implication of Total Dependencies*

3.1. The Implication Problem. For a set of dependencies D we denote by SAT(D) the set of relations that satisfy all dependencies in D. D *implies* a dependency d, denoted $D \models d$, if SAT$(D) \subseteq$ SAT(d). That is, $D \models d$ if d is satisfied by every relation that satisfies all dependencies in D. If D_1 and D_2 are sets of dependencies such that $D_1 \models D_2$ and $D_2 \models D_1$, then D_1 and D_2 are *equivalent*, denoted $D_1 \models\!\dashv D_2$. The *implication problem* is to test, for a given set of dependencies D and a dependency d, whether $D \models d$. Algorithms that decide implication for some families of dependencies were investigated in [6], [7], [33], [34], and [52].

Lemma 2.[4] *Let D be a finite set of TTGDs. We can construct a TTGD d such that $D \models\!\dashv d$.*

Proof. Let $D = \{\langle u_1, I_1 \rangle, \ldots, \langle u_n, I_n \rangle\}$. For any relation K, let $T^j(K)$ denote $\langle u_j, I_j \rangle(\cdots(\langle u_n, I_n \rangle(K))\cdots)$. By Lemma 1 we can construct a tableau $\langle v, J \rangle$ such that, for all K, $T^1(K) = \langle v, J \rangle(K)$. We claim that $D \models \langle v, J \rangle$ and $\langle v, J \rangle \models D$. If $K \in$ SAT(D), then we can show that $T^j(K) = K$, for $n \geq j \geq 1$, by backward induction

[3] The reason for this terminology will be clarified in Section 4.

[4] This result was independently proved in [26].

on j. It follows that $\langle v, J\rangle(K) = K$ and $K \in \mathrm{SAT}(\langle v, J\rangle)$. If $K \in \mathrm{SAT}(\langle v, J\rangle)$, then $T^j(K), = K$, for $1 \le j \le n$, because if $K \subset T^i(K)$, for some $1 \le i \le n$, then $K \subset T^1(K) = \langle v, J\rangle$ (by the monotonicity property).[5] Therefore, $\langle u_j, I_j\rangle(K) = K$, for $1 \le j \le n$, and $K \in \mathrm{SAT}(D)$. □

3.2. THE CHASE. Intuitively, to test whether $D \models \langle w, I\rangle$ (or $\langle (a_1, a_2), I\rangle$), we "chase" I by D into $\mathrm{SAT}(D)$ and then check if w is in I (or if a_1 and a_2 are identical in I). A *chase of I by D* is a maximal sequence of distinct relations $I_0, I_1, \ldots$ such that $I = I_0$ and I_{n+1} is obtained from I_n by an application of a *chase rule*. To each dependency in D there corresponds a chase rule; TT-rules correspond to TTGDs and E-rules correspond to EGDs.

TT-rule (for a TTGD $\langle w, J\rangle$ in D): I_{n+1} is $\langle w, J\rangle(I_n)$.
E-rule (for an EGD $\langle (a_1, a_2), J\rangle$ in D): Initially I_{n+1} is I_n. Now for some valuation h on J such that $h(J) \subseteq I_n$, identify $h(a_1)$ and $h(a_2)$ in I_{n+1}.

We use "apply a dependency" instead of "apply a chase rule for a dependency."

To make the E-rule unambiguous, we assume that $\mathrm{DOM}(A)$ is totaly ordered for all attributes A in U, and, whenever two values are identified, the greater is identified with the smaller. Given $\langle w, I\rangle$, we take $w(A)$ as the smallest value in $\mathrm{DOM}(A)$. (We can always rename the values in w and I so that this is true.) Similarly, given $\langle (a_1, a_2), I\rangle$, we take a_1 and a_2 as the smallest values in $\mathrm{DOM}(A)$ and $a_1 < a_2$. Thus, the values in w or a_1 do not change in the chase and a_2 can be identified only with a_1.

To trace the tuples of I in the chase, we adjoin to each tuple an ordinal number; that is $I = \{w_1, \ldots, w_m\}$. The ordinal numbers do not change during the computation of the chase, though the values in the tuples may change by the E-rules. Thus, I_n consists of the tuples $w_1^n, \ldots, w_m^n$ (not necessarily distinct), which correspond to the tuples $w_1, \ldots, w_m$ of I, plus other tuples, which were added ty TT-rules.

LEMMA 3. *Let I be a finite relation, let J be a relation in $SAT(D)$, let h be a valuation on I such that $h(I) \subseteq J$, and let I_n be a relation in a chase of I by D. Then*

(1) $h(I_n) \subseteq J$;
(2) $h(w_j) = h(w_j^n)$, $1 \le j \le m$.

PROOF. The proof is by induction on n.

Basis ($n = 0$). Trivial.
Induction. Suppose that $h(I_{n-1}) \subseteq J$ and $h(w_j) = h(w_j^{n-1})$, $1 \le j \le m$.

If I_n is obtained by a TT-rule, then $w_j^{n-1} = w_j^n$. Hence $h(w_j^{n-1}) = h(w_j^n) = h(w_j)$, $1 \le j \le m$. Let now t be in I_n; that is, for some TTGD $\langle u, K\rangle \in D$ there is a valuation g on K such that $g(K) \subseteq I_{n-1}$ and $g(u) = t$. But then $h \circ g(K) \subseteq J$, and since $J \subseteq \mathrm{SAT}(D)$ it must be the case that $h(t) = h \circ g(u) \in J$.

If I_n is obtained by an E-rule, then either $w_j^{n-1} = w_j^n$ and $h(w_j^n) = h(w_j^{n-1}) = h(w_j)$, or some value in w_j^{n-1} is changed by the E-rule. That is, there is an EGD $\langle (a_1, a_2), K\rangle \in D$ and a valuation g on K such that $g(K) \subseteq I_{n-1}$ and $w_j^{n-1}[A] = g(a_1)$ is identified with $v[A] = g(a_2)$ for some $v \in I_{n-1}$. But then $h \circ g(K) \subseteq J$, and, since $J \in \mathrm{SAT}(D)$, it must be the case that

$$h(w_j^{n-1}[A]) = h \circ g(a_1) = h \circ g(a_2) = h(v[A]) = h(w_j^n[A]).$$

Hence $h(w_j^n) = h(w_j^{n-1}) = h(w_j)$. □

[5] We use $\subseteq$ to denote set containment and $\subset$ to denote proper containment.

Since we do not require any specific order for the application of the rules in the chase, many chases are possible.

LEMMA 4. *Let I be a finite relation. Then all chases of I by D are finite and have the same last relation, which is in SAT(D).*

PROOF. There is a finite number of relations composed of values from I. Since an E-rule application eliminates values, and a TT-rule application adds new tuples composed of values that already exist in the relation, all relations in the computation sequence are distinct and are contained in the relation consisting of all tuples with values from I. Hence, all chases are finite. Let J be the result of a chase; that is, J is the last relation in a chase. If $J \notin \text{SAT}(D)$, then either there is a TTGD $\langle u, M\rangle \in D$ and a valuation h on M such that $h(M) \subseteq J$ but $h(u) \notin J$, or there is an EGD $\langle (a_1, a_2), M\rangle \in D$ and a valuation h on M such that $h(M) \subseteq J$ but $h(a_1) \neq h(a_2)$. In the first case a TT-rule can be applied, and in the last case an E-rule can be applied—contradicting the assumption that J is the result of a chase.

Uniqueness of the last relation can be proved by the argument of [33, Lemma 5] with minor modifications using Lemma 3, or by the argument of [27]. We give here a different proof for the case that D is a set of TTGDs. Suppose that two chases yield two different results $J, K \in \text{SAT}(D)$. Then $L = J \cap K \in \text{SAT}(D)$, because for all TTGDs $\langle u, M\rangle \in D$ and valuations h on M, if $h(M) \subseteq L$, then $h(M) \subseteq J$ and $h(M) \subseteq K$, and since $J, K \in \text{SAT}(D)$, $h(u) \in J \cap K = L$. Also $I \subseteq L$, because $I \subseteq J$ and $I \subseteq K$. Suppose that $J \neq K$. Then either $L \subset J$ or $L \subset K$. Assume that $L \subset J$. Let $I_0, \ldots, I_n = J$ be a chase of I by D, and let I_k be the first relation in the sequence such that $L \subset I_k$. There is a tuple t such that $t \in I_k - L$ and $t \notin I_{k-1}$. That is, for some TTGD $\langle u, M\rangle \in D$ and a valuation h on M, we have $h(M) \subseteq I_{k-1} \subseteq L$ but $h(u) = t \notin L$-contradicting the claim that $L \in \text{SAT}(D)$. □

Let $\text{chase}_D(I)$ denote the unique last relation of all chases of I by D.
We can use the chase as a decision procedure.

THEOREM 1. *Let D be a set of TTGDs and EGDs. Then*

(1) $D \models \langle w, I\rangle$ *if and only if* $w \in \text{chase}_D(I)$;
(2) $D \models \langle (a_1, a_2), I\rangle$ *if and only if* $a_2 \notin \text{VAL}(\text{chase}_D(I))$ *or* $a_2 = a_1$.

PROOF. $\text{chase}_D(I)$ is the last relation I_n in some chase of I by D.

(*If*). Let $J \in \text{SAT}(D)$ be a relation, and let h be a valuation on I such that $h(I) \subseteq J$. By Lemma 3, $h(I_n) \subseteq J$.

(1) Since $w \in I_n$, $h(w) \in J$, so $J \in \text{SAT}(\langle w, I\rangle)$.
(2) Let $a_1 = w_i[A]$ and $a_2 = w_j[A]$, for some tuples $w_i, w_j \in I$. To prove that $J \in \text{SAT}(\langle a_1, a_2\rangle, I)$, we have to show that $u[A] = v[A]$, where $u, v \in J$, $u = h(w_i)$ and $v = h(w_j)$. If $a_1 = a_2$, then we are done, so assume that $a_1 \neq a_2$. Now, a_1 and a_2 are the smallest values in DOM(A), so a_2 can not be identified with any element except for a_1. Since $a_2 \notin \text{VAL}(I_n)$, it must be the case that a_2 was identified with a_1; that is, $w_j^n[A] = w_j^n[A] = a_1$. But, by Lemma 3, $h(w_j^n) = u$ and $h(w_j^n) = v$, so $u[A] = v[A]$.

(*Only if*). Clearly, $w_i[B] = w_j[B]$ implies $w_i^n[B] = w_j^n[B]$, for all tuples $w_i, w_j \in I$ and for all attributes $B \in U$. Thus, we can define a valuation h on I such that $h(w_i) = w_i^n$, $1 \leq i \leq m$.

(1) Suppose that $w \notin I_n$, h is the identity on w. Thus $h(w) \notin I_n$, and $I_n \in \text{SAT}(D) - \text{SAT}(\langle w, I\rangle)$—contradiction.

(2) Suppose that $a_1 \neq a_2$ and $a_2 \in \mathrm{VAL}(I_n)$. Thus, $h(a_1) = a_1 \neq a_2 = h(a_2)$ and $I_n \in \mathrm{SAT}(D) - \mathrm{SAT}(\langle (a_1, a_2), I\rangle)$—contradiction. □

Example 1 (continued). Let d be the TTGD $\langle u, K\rangle$:

	A	B	C	D
u:	$a0$	$b0$	$c0$	$d0$
	$a0$	$b0$	$c1$	$d1$
K:	$a0$	$b1$	$c0$	$d2$
	$a1$	$b0$	$c0$	$d0$

Let E be $\{d_1, d_2\}$. We use the chase to show that $E \models d$. We start by applying d_1 to K, getting K_1:

	A	B	C	D
	$a0$	$b0$	$c1$	$d1$
	$a0$	$b1$	$c0$	$d2$
	$a1$	$b0$	$c0$	$d0$
new tuple:	$a0$	$b0$	$c0$	$d1$
new tuple:	$a0$	$b1$	$c1$	$d2$

Now we apply d_2 to K_1 getting K_2:

	A	B	C	D
	$a0$	$b0$	$c1$	$d1$
	$a0$	$b1$	$c0$	$d2$
new tuple:	$a0$	$b0$	$c0$	$d0$
	$a0$	$b0$	$c0$	$d1$
	$a0$	$b1$	$c1$	$d2$

The new tuple is exactly u. Thus $u \in \mathrm{chase}_E(K)$ and $E \models \langle u, K\rangle$. □

A dependency d is *trivial* if $\emptyset \models d$; that is, d is satisfied by every relation. Obviously a trivial dependency is a meaningless constraint.

LEMMA 5. *A dependency d is trivial if and only if*

(1) *d is a TTGD $\langle w, I\rangle$ and $w \in I$, or*
(2) *d is an EGD $\langle (a_1, a_2), I\rangle$ and $a_1 = a_2$.*

PROOF. The claim follows by Theorem 1 from the fact that $\mathrm{chase}_\emptyset(I) = I$. □

3.3. COMPLEXITY ANALYSIS. Let d be $\langle *, I\rangle$ ($*$ is either a tuple or a pair of values), where $|I| = m$ and $|U| = n$. Let $I_0, I_1, \ldots, I_p$ be a chase of I by D. There are at most m^n tuples w such that $\mathrm{VAL}(w) \subseteq \mathrm{VAL}(I)$. Thus, $|I_j| \leq m^n$ for $1 \leq j \leq p$. To compute I_{j+1} from I_j, we have to find either a TTGD $\langle u, J\rangle \in D$ and a valuation h on J such that $h(J) \subseteq I_j$ but $h(u) \notin I_j$, or an EGD $\langle (a_1, a_2), J\rangle \in D$ and a valuation h on J such that $h(J) \subseteq I_j$ but $h(a_1) \neq h(a_2)$. Let s be the number of symbols in D. For each valuation h, checking the above takes time $O(sm^n)$. There are at most m^s possible valuations on D. Thus comuting I_{j+1} from I_j takes time $O(sm^{n+s})$. Since a TT-rule adds at least one tuple and an E-rule eliminates at least one value, $p \leq m^n + mn \simeq m^n$. It follows that the computation of $\mathrm{chase}_D(I)$ takes time $O(sm^{2n+s})$. Checking for the conditions of Theorem 1 may take additional time $O(m^n)$. Thus we have proved:

THEOREM 2. *Testing whether D implies d can be done in time $O(sm^{2n+s})$, where s is the number of symbols in D, m the number of tuples in d, and n the number of attributes.*

Example 2. Let $U = \{A, B, C, D_1, \ldots, D_n\}$. For domains we take $\mathrm{DOM}(\alpha) = \{\alpha\} \times N$,[6] for all attributes $\alpha \in U$. For example, $\langle A, 1\rangle \in \mathrm{DOM}(A)$ and $\langle D_1, 2\rangle \in \mathrm{DOM}(D_1)$. For clarity we usually omit the first element of the pair.

Let $E = \{A \twoheadrightarrow BC, A \twoheadrightarrow D_1, \ldots, A \twoheadrightarrow D_n\}$, and let d be $A \twoheadrightarrow B$. Observe that the MVD $X \twoheadrightarrow Y$ is equivalent to the TTGD $\langle u, \{v, w\}\rangle$, where $X = \{E \mid v[E] = w[E]\}$ and $XY = \{E \mid u[E] = v[E]\}$. In particular, $A \twoheadrightarrow B$ is equivalent to the TTGD $\langle t, I\rangle$, $I = \{r, s\}$, where $t[A] = r[A] = s[A] = 1$, $t[B] = r[B] = 1$, $s[B] = 2$, $t[C] = s[C] = 2$, $r[C] = 1$, $r[D_i] = 1$, $t[D_i] = s[D_i] = 2$, for $1 \le i \le n$. The reader can verify that $J = \mathrm{chase}_E(I)$ is exactly

$$\{u \mid u[A] = 1,\ u[B] = u[C] = i_0,\ u[D_j] = i_j,\ 1 \le j \le n,\ \text{where}\ \{i_0, \ldots, i_n\} \subseteq \{1, 2\}\}$$

Thus, $|J| = 2^{n+1}$. But $t \notin J$, so $E \not\models d$. That is, the chase adds an exponential number of tuples before giving the negative answer $E \not\models d$. □

Theorem 2 gives an exponential time upper bound for the implication problem for TTDs and EGDs. An exponential time lower bound is proven in [19], and NP-hardness results for some special cases are proven in [12] and [34]. In Section 3.5 we indicate how testing implication of MVDs can be done in polynomial time.

3.4. THE ROLE OF EGDs. Equality is not an essential part of first-order logic. That is, the equality predicate can be represented by any binary predicate symbol by adding the necessary equality axioms. In [13] we show how this can be done for dependencies expressed in the language of first-order logic. Since we assumed that distinct attributes have disjoint domains, we can not employ this technique in our formalism. Nevertheless, the role of EGDs can be "minimized." We show here that every EGD is equivalent to a conjunction of a TTGD and an FD, and that the implication problem for TTGDs and EGDs is reducible to the implication problem for TTGDs.

Let e be an A-EGD, $e = \langle (a_1, a_2), I\rangle$. Let w_1 be a tuple such that $w_1[A] = a_1$ and for all attributes $B \in \bar{A}$, $w_1[B] \notin I[B]$. Let w_2 be a tuple such that $w_2[\bar{A}] = w_1[\bar{A}]$ and $w_2[A] = a_2$. We associate with e two TTGDs. e_1 is $\langle w_1, I \cup \{w_2\}\rangle$, and e_2 is $\langle w_2, I \cup \{w_1\}\rangle$. Intuitively, e_1 states that, given I, wherever a_2 appears, a_1 also appears. More precisely, if a relation contains $h(I)$, then for each tuple in it that contains $h(a_2)$, there exists a tuple identical to it except that $h(a_2)$ is replaced by $h(a_1)$. Similarly, e_2 states that wherever a_1 appears, a_2 also appears.

Example 3. Let e be $\langle (a0, a1), J\rangle$, where J is the relation:

A	B	C	E
$a0$	$b0$	$c0$	$d0$
$a1$	$b0$	$c0$	$d1$

e is equivalent to the FD $BC \to A$. w_1 and w_2 are the tuples:

	A	B	C	D
w_1:	$a0$	$b1$	$c1$	$d2$
w_2:	$a1$	$b1$	$c1$	$d2$

[6] N is the set of natural numbers.

e_1 is the TTGD $\langle w_1, J \cup \{w_2\}\rangle$:

A	B	C	D
$a0$	$b1$	$c1$	$d2$
$a0$	$b0$	$c0$	$d0$
$a1$	$b0$	$c0$	$d1$
$a1$	$b1$	$c1$	$d2$

e_2 is the TTGD $\langle w_2, J \cup \{w_1\}\rangle$:

A	B	C	D
$a1$	$b1$	$c1$	$d2$
$a0$	$b0$	$c0$	$d0$
$a1$	$b0$	$c0$	$d1$
$a0$	$b1$	$c1$	$d2$

LEMMA 6. *Let e be a nontrivial A-EGD. Then $e \models \exists\{\bar{A} \to A, e_1\}$ and $e \models \exists\{\bar{A} \to A, e_2\}$.*

PROOF. We prove the claim for e_1. The proof for e_2 is analogous.

Let $J \in \mathrm{SAT}(e)$, and let u and v be tuples in J such that $u[\bar{A}] = v[\bar{A}]$. Define a valuation h on I as follows: Let $t \in I$. If $t[A] = a_1$, then $h(t) = u$; else $h(t) = v$. It is easy to see that h is well defined, $h(a_1) = u[A]$, and $h(a_2) = v[A]$. Since I satisfies e, we must have that $u[A] = v[A]$. Therefore, I satisfies $\bar{A} \to A$. We have shown that $e \models \bar{A} \to A$.

To show that $e \models e_1$ we prove that $w_1 \in \mathrm{chase}_{\{e\}}(I \cup \{w_2\})$. Let h be the identity valuation on I. Clearly, $h(I) \subseteq I \cup \{w_2\}$. Applying e to $I \cup \{w_2\}$, we identify $h(a_2) = a_2$ with $h(a_1) = a_1$ and get w_1.

To show that $\{\bar{A} \to A, e_1\} \models e$, we prove that $a_2 \notin \mathrm{VAL}(\mathrm{chase}_{\{\bar{A}\to A, e_1\}}(I))$. To compute $\mathrm{chase}_{\{\bar{A}\to A, e_1\}}(I)$, we apply first $e_1 = \langle w_1, I \cup \{w_2\}\rangle$ to get I_1. Let $u \in I$ be a tuple such that $u[A] = a_2$, then $u \in I_1$, and there is also a new tuple $v \in I_1$, where $v[A] = a_1$ and $v[\bar{A}] = u[\bar{A}]$. Applying $\bar{A} \to A$ to I_1 we identify a_2 with a_1, so $a_2 \notin \mathrm{VAL}(\mathrm{chase}_{\{\bar{A}\to A, e_1\}}(I))$. □

Since a trivial EGD is clearly equivalent to some trivial TTGD and also to some trivial FD, Lemma 7 entails that every EGD is equivalent to a conjunction of a TTGD and an FD.

Let us consider first implication of TTGDs. Let D^* be the result of replacing each EGD e in D by the associated TTGDs e_1 and e_2.

THEOREM 3. *Let d be a TTGD. Then $D \models d$ if and only if $D^* \models d$.*

PROOF

(*If*). By Lemma 6, $D \models D^*$. The claim follows.

(*Only if*). Let d be $\langle u, J\rangle$. $D \models d$ only if $u \in \mathrm{chase}_D(J)$. We show that $\mathrm{chase}_D(J) \subseteq \mathrm{chase}_{D^*}(J)$; hence $u \in \mathrm{chase}_{D^*}(J)$ and $D^* \models d$.

Let $J_0, \ldots, J_n$ be a chase of I by D. We describe a chase of I by D^*: $J_0', \ldots, J_n', \ldots, J_m'$, such that $J_i \subseteq J_i'$, for $1 \le i \le n$. J_0' is J_0, and clearly $J_0 \subseteq J_0'$. Assume that $J_i \subseteq J_i'$. If J_{i+1} is obtained by applying a TTGD to J_i, then J_{i+1}' is obtained by applying the same TTGD to J_i'. Obviously, in this case $J_{i+1}' \subseteq J_{i+1}$. If J_{i+1} is obtained by applying an EGD e to J_i, then J_{i+1}' is obtained by applying e_1 or e_2 to J_i'. Let e be $\langle (a_1, a_2), I\rangle$. Then e_1 is $\langle w_1, I \cup \{w_2\}\rangle$ and e_2 is $\langle w_2, I \cup \{w_1\}\rangle$. Let $t \in J_{i+1} - J_i$. Then t was obtained by identifying $v[A]$ with $w[A]$, for some

tuples v, $w \in J_i$, where $v[A] = h(a_1)$ and $w[A] = h(a_2)$ for some valuation h on I such that $h(I) \subseteq J_i$. (The other case, $v[a] = h(a_2)$ and $w[A] = h(a_1)$, is symmetrical.) Recall that $w_1[A] = a_1$ and all other values in w_1 are new distinct values. Thus, h can be extended to w_1 so that $h(w_1) = v$. But then we have $h(I \cup \{w_1\}) \subseteq J_i \subseteq J_i'$ and $h(w_2) = t$, so by applying e_1 to J_i' we get $t \in J_{i+1}'$. It follows that $J_{i+1} \subseteq J_{i+1}'$ and, in particular, $J_n \subseteq J_n'$. Now $J_{n+1}', \ldots, J_m'$ are obtained by application of TTGDs in D^*; thus $J_n \subseteq J_i'$ for $n < i \leq m$. Hence, $\text{chase}_D(J) \subseteq \text{chase}_{D^*}(J)$. □

Consider now implication of EGDs.

LEMMA 7. *Let D be a set of TTGDs, and let e be an EGD. Then $D \models e$ if and only if e is trivial.*

PROOF

(*If*). If e is trivial, then $\varnothing \models e$ and obviously $D \models e$.

(*Only if*). Suppose that e is a nontrivial EGD $\langle (a_1, a_2), I \rangle$; that is, $a_1 \neq a_2$. But, obviously, $a_2 \in \text{VAL}(\text{chase}_D(I))$, so $D \not\models e$. □

THEOREM 4. *Let e be a nontrivial A-EGD $\langle (a_1, a_2), I \rangle$. Then $D \models e$ if and only if for some nontrivial A-EGD $\langle (a_i, a_j), J \rangle \in D$ and a valuation h, we have that $h(J) \subseteq \text{chase}_{D^*}(I)$, $h(a_i) = a_1$ and $h(a_j) = a_2$.*

PROOF

(*If*). Since $D \models D^*$, $D \models e$ iff $D \cup D^* \models e$. Thus, it suffices to show that $a_2 \notin \text{VAL}(\text{chase}_{D \cup D^*}(I))$. To compute $\text{chase}_{D \cup D^*}(I)$, we apply first TTGDs in D^* until no more change is effected, that is, until we get $\text{chase}_{D^*}(I)$ in the chase. By assumption, there is an EGD $\langle (a_i, a_j), J \rangle \in D$ and a valuation h on J such that $h(J) \subseteq \text{chase}_{D^*}(I)$, $h(a_i) = a_1$, and $h(a_j) = a_2$. Applying $\langle (a_i, a_j), J \rangle$ to $\text{chase}_{D^*}(I)$, we identify a_2 with a_1, so $a_2 \notin \text{VAL}(\text{chase}_{D \cup D^*}(I))$.

(*Only if*). Let $I_0, \ldots, I_n$ be a chase of I by D. By assumption, for some $0 \leq k \leq n$, an EGD $\langle (a_i, a_j), J \rangle \in D$, and a valuation h on J, we have $h(J) \subseteq I_k$, $h(a_i) = a_1$, and $h(a_j) = a_2$. Construct a chase of I by D^*: $I_0', \ldots, I_n', \ldots, I_m'$, as described in the proof of Theorem 3. We know that $I_k \subseteq I_k' \subseteq I_m'$; thus, $h(J) \subseteq \text{chase}_{D^*}(I)$, $h(a_i) = a_1$, and $h(a_j) = a_2$. Since $a_1 \neq a_2$, also $a_i \neq a_j$; so $\langle (a_i, a_j), J \rangle$ is nontrivial. □

COROLLARY. *If e is a nontrivial A-EGD, then $D \models e$ if and only if $D^* \models e_1$ and there is a nontrivial A-EGD in D.*

PROOF

(*If*). Assume that $D^* \models e_1$ and there is a nontrivial A-EGD in D. By Lemma 6, this nontrivial A-EGD implies the FD $\bar{A} \to A$. Again by Lemma 6, $\{\bar{A} \to A, e_1\} \models e$.

(*Only if*). Assume that $D \models e$. By Lemma 6, $D \models e_1$, and by Theorem 3, $D^* \models e_1$. By the above theorem there is a nontrivial A-EGD in D. □

3.5. TUPLE CLOSURE. Lemma 4 asserts that $\text{chase}_D(I)$ is invariant under changes in the order of rules application. The following lemma says that $\text{chase}_D(I)$ is invariant also under certain changes in D.

LEMMA 8. *Let I be a finite relation. If $D \models d$, then $\text{chase}_D(I) = \text{chase}_{D \cup \{d\}}(I)$.*

PROOF. By Lemma 4, to compute $\text{chase}_{D \cup \{d\}}(I)$ we can apply the rules in any order. Apply the rules for dependencies in D until no more change is effected, that is, until we get the relation $\text{chase}_D(I)$ in the chase. But $\text{chase}_D(I) \in \text{SAT}(D) =$

SAT($D \cup \{d\}$); hence, no more dependencies in $D \cup \{d\}$ can be applied and $\text{chase}_D(I) = \text{chase}_{D \cup \{d\}}(I)$. □

Using the above invariance result, we derive the following closure property of $\text{chase}_D(I)$.

THEOREM 5. *Let I be a finite relation, and let w be a tuple such that $VAL(w) \subseteq VAL(I)$. The following statements are equivalent:*

(1) $D \models \langle w, I \rangle$,
(2) $w \in chase_D(I)$, *and*
(3) $\langle w, I \rangle(chase_D(I)) = chase_D(I)$.

PROOF

(1 $\Leftrightarrow$ 2) By Theorem 1.

(2 $\Rightarrow$ 3) Suppose that $w \in \text{chase}_D(I)$. By Theorem 1, $D \models \langle w, I \rangle$. Hence, by Lemma 8, $\text{chase}_{D \cup \{\langle w,I \rangle\}}(I) = \text{chase}_D(I)$. However, if $\text{chase}_D(I) \subset \langle w, I \rangle(\text{chase}_D(I))$, then also $\text{chase}_D(I) \subset \text{chase}_{D \cup \{\langle w,I \rangle\}}(I)$. It follows that $\langle w, I \rangle(\text{chase}_D(I)) = \text{chase}_D(I)$.

(2 $\Leftarrow$ 3) Suppose that $\langle w, I \rangle(\text{chase}_D(I)) = \text{chase}_D(I)$. We know (see the proof of Theorem 1) that we can define a valuation h on I such that $h(I) \subseteq \text{chase}_D(I)$ and $h(w) = w$. It follows that $w = h(w) \in \text{chase}_D(I)$. □

Theorem 5 has the following intuitive meaning. For a given relation I, the TTGDs $\langle w, I \rangle$ that are implied by D are represented by the tuples that appear in $\text{chase}_D(I)$. As shown above, this leads to regarding the chase as a decision procedure for implication. In general, this procedure has the exponential time bound derived in the preceding section. However, in some special cases, the set of tuples in the chase can be described succinctly. For such cases, a more efficient procedure can be found. Since our interest now is in the implication of TTGDs, we can assume without loss of generality, by Theorem 3, that all the given dependencies are TTGDs.

Let $I = \{w_1, w_2\}$, and let $X = \{A \mid w_1[A] = w_2[A]\}$. Obviously, for all $w \in \text{chase}_D(I)$, we have $w[X] = w_1[X] = w_2[X]$. We can characterize each tuple $w \in \text{chase}_D(I)$ by the set $\text{FIRST}(w) = \{A \in \bar{X} \mid w[A] = w_1[A]\}$, since $w[\bar{X} - \text{FIRST}(w)] = w_2[\bar{X} - \text{FIRST}(w)]$. By definition, $\text{FIRST}(w_1) = \bar{X}$ and $\text{FIRST}(w_2) = \emptyset$.

LEMMA 9. *Let $I = \{w_1, w_2\}$, and let D be a set of TTGDs. For all $w', w'' \in chase_D(I)$, there exist tuples $t_1, t_2 \in chase_D(I)$, such that*

(1) $FIRST(t_1) = FIRST(w') \cup FIRST(w'')$;
(2) $FIRST(t_2) = \bar{X} - FIRST(w')$.

PROOF

(1) Let h be a valuation on I such that $h(w_1) = w'$ and $h(w_2) = w''$. By Theorem 5, $\langle w', I \rangle(\text{chase}_D(I)) = \text{chase}_D(I)$, so $h(w') \in \text{chase}_D(I)$. But

$$\begin{aligned}
\text{FIRST}(h(w')) &= \{A \in \bar{X} \mid h(w')[A] = w_1[A]\} \\
&= \{A \in \text{FIRST}(w') \mid h(w')[A] = w_1[A]\} \\
&\quad \cup \{A \in \bar{X} - \text{FIRST}(w') \mid h(w')[A] = w_1[A]\} \\
&= \{A \in \text{FIRST}(w') \mid w'[A] = w_1[A]\} \\
&\quad \cup \{A \in \bar{X} - \text{FIRST}(w') \mid w''[A] = w_1[A]\} \\
&= \text{FIRST}(w') \cup (\text{FIRST}(w'') - \text{FIRST}(w')) \\
&= \text{FIRST}(w') \cup \text{FIRST}(w'').
\end{aligned}$$

So take t_1 to be $h(w')$.

(2) Let h be a valuation on I such that $h(w_1) = w_2$ and $h(w_2) = w_1$. By Theorem 5, $h(w') \in \text{chase}_D(I)$. But

$$\begin{aligned}\text{FIRST}(h(w')) &= \{A \in \bar{X} \mid h(w')[A] = w_1[A]\} \\ &= \{A \in \text{FIRST}(w') \mid h(w')[A] = w_1[A]\} \\ &\quad \cup \{A \in \bar{X} - \text{FIRST}(w') \mid h(w')[A] = w_1[A]\} \\ &= \{A \in \text{FIRST}(w') \mid w_2[A] = w_1[A]\} \\ &\quad \cup \{A \in \bar{X} - \text{FIRST}(w') \mid w_1[A] = w_1[A]\} \\ &= \varnothing \cup (\bar{X} - \text{FIRST}(w')) = \bar{X} - \text{FIRST}(w').\end{aligned}$$

So take t_2 to be $h(w')$. □

Note that $\langle w, \{w_1, w_2\}\rangle$ is equivalent to the MVD $X \longrightarrow\!\!\!\!\!\rightarrow \text{FIRST}(w)$. Lemma 9 then is simply a restatement of the Boolean properties of the MVDs implied by D, as given in [9].

The Boolean closure property can be used to test implications efficiently.

THEOREM 6. *Let D be a set of TTGDs, let $I = \{w_1, w_2\}$, and let $X = \{A \mid w_1[A] = w_2[A]\}$. There exists a partition of $\bar{X}$ into disjoint nonempty sets $W_1, \ldots, W_m$, $1 \le m$, such that for any tuple w, with $VAL(w) \subseteq VAL(I)$, we have that $w \in \text{chase}_D(I)$ if and only if for all i, $1 \le i \le m$, either $W_i \subseteq FIRST(w)$ or $W_i \cap FIRST(w) = \varnothing$.*

PROOF. By Lemma 9 the collection $\{\text{FIRST}(w) \mid w \in \text{chase}_D(I)\}$ is a cover of $\bar{X}$ that contains $\bar{X}$ and is closed under Boolean operations. The subcollection of minimal nonempty sets is the desired partition of $\bar{X}$ as can be easily verified. (This collection is the dependency basis of X with respect to D as defined in [9] and [23].) □

To test if $D \models \langle w, I\rangle$, one has to construct $W_1, \ldots, W_m$, and check for the condition of the theorem. This can be done in polynomial time. Theorem 6 is the key to the efficient algorithms for inferences of MVDs of [6], [34], [52], and [53].

4. *A Proof Procedure for Dependencies*

4.1. TUPLE-GENERATING DEPENDENCIES. As noted earlier, we cannot express *embedded multivalued dependencies* (EMVDs) by TTGDs. An EMVD is a statement $X \longrightarrow\!\!\!\!\!\rightarrow Y \mid Z$. It is satisfied by a relation I if, for all $t_1, t_2 \in I$, we have that if $t_1[X] = t_2[X]$, then there exists a tuple $t \in I$ such that $t[XY] = t_1[XY]$ and $t[XZ] = t_2[XZ]$.

Tuple-generating dependencies (TGD) generalize TTGDs and EMVDs. A TGD says that if some tuples, fulfilling certain conditions, exist in the database, then some other tuples (possibly with some unknown values), fulfilling certain conditions, must also exist in the database. Formally, a TGD is a pair of finite relations $\langle I', I\rangle$. It is satisfied by a relation J if for any valuation h, such that $h(I) \subseteq J$, there is an extension h' of h to I' so that $h'(I') \subseteq J$.

The class of TGDs and EGDs is equivalent to Fagin's class of *embedded implicational dependencies* [25], and is also shown in [54] to be equivalent to the class of *algebraic dependencies*.

Example 4. Let I and L be the relations:

I:

A	B	C	D
$a0$	$b0$	$c1$	$d0$
$a0$	$b1$	$c0$	$d1$

L:

A	B	C	D
$a1$	$b0$	$c0$	$d0$
$a1$	$b1$	$c1$	$d1$

$\langle L, I\rangle$ is a TGD. A relation J satisfies $\langle L, I\rangle$ if, for all $t_1, t_2 \in J$, we have that if $t_1[A] = t_2[A]$; then there exist tuples $s_1, s_2 \in J$ such that $s_1[BD] = t_1[BD]$, $s_1[C] = t_2[C]$, $s_2[BD] = t_2[BD]$, and $s_2[C] = t_1[C]$. □

If $\mathrm{VAL}(I') \subseteq \mathrm{VAL}(I)$, then $\langle I', I\rangle$ is *total*. If for all tuples $w \in I'$ we have $\mathrm{VAL}(w) \not\subseteq \mathrm{VAL}(I)$, then $\langle I', I\rangle$ is *partial*. (Note that a TGD may be nontotal and nonpartial.) Viewing a TTGD $\langle w, I\rangle$ as a TGD $\langle \{w\}, I\rangle$, we see that a TTGD is a special case of a TGD. By the following lemma there is no loss of generality in assuming that every total TGD is of the form $\langle w, I\rangle$.

LEMMA 10. *Let $\langle I', I\rangle$ be a total TGD. Then $\langle I', I\rangle \models \exists \{\langle w, I\rangle \mid w \in I'\}$.*

PROOF. Let J be a relation, and h a valuation, on I (hence, also on I') such that $h(I) \subseteq J$. If $J \in \mathrm{SAT}(\langle I', I\rangle)$, then $h(I') \subseteq J$. That is, $h(w) \in J$ for all $w \in I'$. It follows that $J \in \mathrm{SAT}(\{\langle w, I\rangle \mid w \in I'\})$. If $J \in \mathrm{SAT}(\{\langle w, I\rangle \mid w \in I'\})$, then $h(w) \in J$ for all $w \in I'$. That is, $h(I') \subseteq J$. It follows that $J \in \mathrm{SAT}(\langle I', I\rangle)$. □

COROLLARY. *Let $\langle I', I\rangle$ be a total TGD. Then we can effectively construct a TTGD $\langle w, J\rangle$, such that $\langle I', I\rangle \models \exists \langle w, J\rangle$.*

PROOF. The claim follows from the above lemma by Lemma 2. □

This justifies our definition of TTGD in Section 3.

The following condition for triviality generalizes Lemma 5(1).

LEMMA 11. *A TGD $\langle I', I\rangle$ is trivial if and only if there exist a valuation h, which is the identity on I, such that $h(I') \subseteq I$.*

PROOF

(*If*). Let J be a relation, and g a valuation on I such that $g(I) \subseteq J$. Consider the valuation $g \circ h$. Since h is the identity on I, $g \circ h$ is an extension of g to I'. But $h(I') \subseteq I$, so $g \circ h(I') \subseteq g(I) \subseteq J$.

(*Only if*). Let h be the identity on I. Since $I \in \mathrm{SAT}(\langle I', I\rangle)$ and $h(I) \subseteq I$, it must be the case that h can be extended to I' so that $h(I') \subseteq I$. □

Theorem 3 can be generalized to deal with implications of TGDs by TGDs and EGDs. Unlike the proof-theoretic proof of Theorem 3, the proof of the following theorem is model theoretic.

THEOREM 7. *Let D be a set of TGDs and EGDs, and let d be a TGD. $D \models d$ if and only if $D^* \models d$.*

PROOF

(*If*). By Lemma 6, $D \models D^*$. The claim follows.

(*Only if*). Suppose that $D^* \not\models d$; then there is a relation $J \in \mathrm{SAT}(D^*) - \mathrm{SAT}(d)$. We construct a relation $J' \in \mathrm{SAT}(D) - \mathrm{SAT}(d)$ to show that also $D \not\models d$.

Let $\equiv_J$ be an equivalence relation on Dom as follows: For each attribute $A \in U$ and values $a_1, a_2 \in \mathrm{DOM}(A)$, $a_1 \equiv_J a_2$, iff for all tuples u and v such that $u[\bar{A}] = v[\bar{A}]$, $u[A] = a_1$, and $v[A] = a_2$, we have that $u \in J$ iff $v \in J$. Intuitively, a_1 and a_2 are equivalent if they look identical when seen from "within" J. We denote the equivalence class of a by $[a]_J$, and we let a' be a representative element of $[a]_J$ (a' is independent of a). Let g be a valuation such that $g(a) = a'$, for all $a \in$ Dom. Let $J' = g(J)$. Observe that g is the identity on J'.

CLAIM 1. $J' \subseteq J$.

Let $w \in J$. Choose an attribute A, and define a tuple v: $v[A] = g(w[A])$, $v[\bar{A}] = w[\bar{A}]$. Since $w[A] \equiv_J g(w[A]) = v[A]$, $v \in J$. By induction it follows that if we replace any number of values in w by their images under g, the resulting tuple will be in J. In particular, $g(w) \in J$. It follows that $J' \subseteq J$.

CLAIM 2. *If* $J \in SAT(\langle K, I\rangle)$, *then* $J' \in SAT(\langle K, I\rangle)$.

Suppose that $J \in \text{SAT}(\langle K, I\rangle)$. Let h be a valuation on I such that $h(I) \subseteq J'$. By Claim 1, $h(I) \subseteq J$; hence, there is an extension h' of h to K such that $h'(K) \subseteq J$. Since h and h' agree on I, $h(I) \subseteq J'$, and g is the identity on J', we have that h and $g \circ h'$ agree on I. That is, $g \circ h'$ is an extension of h to K. But $h'(K) \subseteq J$, so $g \circ h'(K) \subseteq J'$.

CLAIM 3. *If* $J' \in SAT(\langle K, I\rangle)$, *then* $J \in SAT(\langle K, I\rangle)$.

Suppose that $J' \in \text{SAT}(\langle K, I\rangle)$. Let h be a valuation on I such that $h(I) \subseteq J$. Now $g \circ h(I) \subseteq g(J) = J'$, so there is an extension f of $g \circ h$ to K such that $f(K) \subseteq J' \subseteq J$. Let h' be an extension of h to K such that, h' agrees with f on $\text{VAL}(K) - \text{VAL}(I)$. We now show that $h'(K) \subseteq J$. Let $w \in K$. We know that $f(w) \in J$. Choose an attribute A and define a tuple v: $v[\bar{A}] = f(w)[\bar{A}]$ and $v[A] = h'(w)[A]$. If $w[A] \in I[A]$, then $f(w)[A] = g \circ h(w)[A]$ and $h'(w)[A] = h(w)[A]$; so $v[A] \equiv_J f(w)[A]$. If $w[A] \in K[A] - I[A]$, then $v[A] = f(w)[A]$. In either case $v \in J$. As before, it follows that $h'(w) \in J$.

CLAIM 4. *Let e be an EGD, and let e_1 and e_2 be the TTGDs associated with e. If $J \in SAT(\{e_1, e_2\})$, then $J' \in SAT(e)$.*

Let e be $\langle (a_1, a_2), I\rangle$, and suppose that $J \in \text{SAT}(\{e_1, e_2\})$. Let h be a valuation on I such that $h(I) \subseteq J' \subseteq J$. Let u and v be tuples such that $u[\bar{A}] = v[\bar{A}]$, $u[A] = h(a_1)$, and $v[A] = h(a_2)$. We can extend h to w_1 and w_2 so that $h(w_1) = u$ and $h(w_2) = v$. Since $J \in \text{SAT}(\{e_1, e_2\})$, it follows that $u \in J$ iff $v \in J$. That is, $h(a_1) \equiv_J h(a_2)$ and $g(h(a_1)) = g(h(a_2))$. But $h(a_1), h(a_2) \in J'[A]$, so $h(a_1) = g(h(a_1)) = g(h(a_2)) = h(a_2)$.

CLAIM 5. $J' \in SAT(D) - SAT(d)$.

Since $J \in \text{SAT}(D^*) - \text{SAT}(d)$, the claim follows by Claims 2–4. □

We have seen that the chase can be used to test implication of TTGDs by TTGDs and EGDs. It can also be used to decide implication of TGDs by TTGDs and EGDs.

THEOREM 8. *Let D be a set of TTGDs and EGDs. Then $D \models \langle I', I\rangle$ if and only if $\langle I', chase_{D^*}(I)\rangle$ is trivial.*

PROOF. By Theorem 7 we can assume that $D = D^*$ (i.e., D is a set of TTGDs).

(*If*). Let $J \in \text{SAT}(D)$, and let h be a valuation on I such that $h(I) \subseteq J$. By Lemma 3, $h(\text{chase}_D(I)) \subseteq J$. But $\langle I', \text{chase}_D(I)\rangle$ is trivial, so there is a valuation h' on $I' \cup I$, which is the identity on I such that $h'(I') \subseteq \text{chase}_D(I)$. Thus, $h \circ h'$ is an extension of h to I', and $h \circ h'(I') \subseteq h(\text{chase}_D(I)) \subseteq J$.

(*Only if*). Since $D \models \langle I', I\rangle$ and $\text{chase}_D(I) \in \text{SAT}(D)$, it follows that $\text{chase}_D(I) \in \text{SAT}(\langle I', I\rangle)$. Let h be the identity valuation on I. Now $h(I) = I \subseteq \text{chase}_D(I)$, so there is an extension h' of h to I' such that $h'(I') \subseteq \text{chase}_D(I)$. That is, $\langle I', \text{chase}_D(I)\rangle$ is trivial. □

COROLLARY. *$D \models \langle I', I\rangle$ if and only if there exist a total TGD $\langle J, I\rangle$ such that $D \models \langle J, I\rangle$, and for some valuation h which is the identity on I, $h(I') = J$.*

PROOF. $D \models \langle I', I\rangle$ iff $\langle I', \text{chase}_{D^*}(I)\rangle$ is trivial iff there is a valuation h, which is the identity on I, such that $h(I') \subseteq \text{chase}_{D^*}(I)$ iff there are tuples $w_1, \ldots, w_k \in \text{chase}_{D^*}(I)$, $k > 0$, such that $h(I') = \{w_1, \ldots, w_k\}$ iff there are TTGDs $\langle w_1, I\rangle, \ldots, \langle w_k, I\rangle$ such that $D \models \{\langle w_i, I\rangle \mid 1 \le i \le k\}$ and $h(I') = \{w_1, \ldots, w_k\}$ iff $D \models \langle\{w_1, \ldots, w_k\}, I\rangle$ and $h(I') = \{w_1, \ldots, w_k\}$. □

Note that since $h(I') = J$, $\langle J, I\rangle \models \langle I', I\rangle$. But h is the identity on I, so I' is in fact J with some values replaced by new values. We can view $\langle I', I\rangle$ as a "weakened version" of $\langle J, I\rangle$. Thus, the corollary above states that a TGD d is implied by a set of total dependencies iff there is a total TGD d' such that $D \models d'$, and d is a "weakened version" of d'. This generalizes the projection completeness theorems in [14] and [48].

Remark. Arguments similar to that of Section 3.3 show that deciding whether $D \models \langle I', I\rangle$ can be done in time $O(sm^{2n+s+e})$, where s, m, and n are as in Theorem 2, and e is the number of values that occur in I' but not in I. □

4.2. A NONTERMINATING CHASE. The implication problem for TGDs is known to be recursively unsolvable [13, 19, 49]; that is, the set $\{\langle D, d\rangle \mid D \models d\}$ is not recursive. However, since dependencies can be expressed as first-order sentences for which the implication problem is partially solvable, the above set is recursively enumerable. In this section we generalize the chase to TGDs. This generalized chase constitutes a proof procedure for the implication problem

Trying to generalize our TT-rule to TGDs, we encounter difficulties, because the new tuples, whose existence in the database is implied by the existence of some other tuples, are only partly known. The solution is to replace each unknown value by a new distinct value. Let $\langle I', I\rangle$ be a TGD and h a valuation on I. A *distinct extension* of h to I' is an extension h' of h to I', where $h'(a)$ is a new distinct value for each $a \in \text{VAL}(I') - \text{VAL}(I)$. There can be many distinct extensions of h to I', so we assume that there is some rule on how to select one of them when it is needed.

$\langle I', I\rangle$ defines an operation on relations as follows:

$$\langle I', I\rangle(J) = \bigcup \{f(I') \mid f \text{ is a distinct extension of a valuation } h \text{ such that } h(I) \subseteq J \text{ but, for no extension } g \text{ of } h \text{ to } I', g(I') \subseteq J\} \cup J.$$

Obviously $\langle I', I\rangle(J)$ is defined up to renaming of the new values. Note that $J \subseteq \langle I', I\rangle(J)$.

Our generalized chase rule is now:

T-rule (for a $\langle J', J\rangle \in D$): I_{n+1} is $\langle J', J\rangle(I_n)$.

Since this rule introduces new values, the chase may be infinite. We say that a dependency is *applicable* in a chase if applying it produces a new relation. The goal of the chase by D is to produce a relation in SAT(D). However, unlike the case when D consists of TTGDs and EGDs where every chase by D produces a relation in SAT(D), this is not so when D consists also of TGDs. Thus we require the following:

(1) Whenever an EGD is applied, it should be applied as long as it is applicable.
(2) Every dependency that is applicable infinitely many times should be applied infinitely many times.

We also assume that the domains are not only totally ordered, but also well founded, so no value can change infinitely many times. Let $\mathbf{I} = I_0, \ldots, I_n, \ldots$ be

a chase of I by D. We define the result of this chase as

$$\text{chase}(\mathbf{I}) = \{w \mid \text{there is some } n \text{ such that for all } m \geq n,\ w \in I_m\}.$$

Observe that, because the domains are well founded, chase(**I**) is nonempty. Obviously, if all dependencies in D are TGDs then chase (**I**) $= \bigcup_{j \geq 0} I_j$.

Lemma 3 still holds for the generalized chase, as the reader can easily verify. Lemma 4 does not necessarily hold.

LEMMA 12. *Let* **I** *be a chase of I by D. Then chase*(**I**) $\in SAT(D)$.

PROOF. Let $\langle J', J\rangle$ be a TGD in D, and h a valuation on J such that $h(J) \subseteq$ chase(**I**). But then, there are some n such that for all $m \geq n$, $h(J) \subseteq I_m$, and, by our assumption on the order of rule application, there must be some $m \geq n$ and an extension h' of h to J' such that $h'(J') \subseteq I_m$. We claim now that there is a sequence of valuations $h_0, h_1, \ldots$, which are the identity on $h(J)$, such that $h_i \circ h_{i-1} \circ \cdots \circ h_0 \circ h'(J') \subseteq I_{m+i}$. The proof is by induction on i. For $i = 0$, h_0 is just the identity valuation. Suppose now that the claim holds for i. If I_{m+i+1} is obtained from I_{m+i} by an application of a TGD, then $I_{m+i} \subseteq I_{m+i+1}$, and h_{i+1} is the identity valuation. Otherwise, I_{m+i+1} is obtained from I_{m+i} by identifying some value a_2 with another value a_1. If $h_i \circ \cdots \circ h_0 \circ h'(J) \subseteq I_{m+i+1}$, then take h_{i+1} to be the identity valuation; otherwise, take $h_{i+1}(a_2) = a_1$ and $h(a) = a$ for $a \neq a_2$. Since no value can be changed infinitely many times, there is some $p \geq m$ such that for all $i \geq p$, h_i is the identity valuation. That is, for all $i \geq p$, $h''(J') \subseteq I_i$, where $h'' = h_p \circ \cdots \circ h_0 \circ h'$. So h'' is an extension of h to J', and $h''(J') \subseteq$ chase(**I**). Let $\langle (a_1, a_2), J\rangle$ be an EGD in D, and let h be a valuation on J such that $h(J) \subseteq \text{chase}_D(I)$ but $h(a_1) \neq h(a_2)$. But then, there is some n such that for all $m \geq n$, $h(J) \subseteq I_m$ and $h(a_1) \neq h(a_2)$—contradicting our requirement for rules application. □

THEOREM 9. *Let D be a set of dependencies. Then*

(1) $D \models \langle I', I\rangle$ *if and only if for all chases* **I** *of I by* D^* *there is a valuation h that is the identity on I and* $h(I') \subseteq$ *chase*(**I**), *if and only if there are a chase* **I** *of I by* D^* *and a valuation h that is the identity on I and* $h(I') \subseteq$ *chase*(**I**).

(2) *Let e be a nontrivial A-EGD* $\langle (a_1, a_2), I\rangle$. *Then* $D \models e$, *if and only if for all chases* **I** *of I by D we have* $a_2 \notin VAL(\text{chase}(\mathbf{I}))$, *if and only if for some chase* **I** *of I by D we have* $a_2 \notin VAL(\text{chase}(\mathbf{I}))$, *if and only if for all chase* **I** *of I by* D^* *there are a nontrivial A-EGD* $\langle (a_i, a_j), J\rangle \in D$ *and a valuation h such that* $h(J) \subseteq$ *chase*(**I**), $h(a_i) = a_1$ *and* $h(a_j) = a_2$, *if and only if for some chase* **I** *of I by* D^* *there are a nontrivial A-EGD* $\langle (a_i, a_j), J\rangle \in D$ *and a valuation h such that* $h(J) \subseteq$ *chase*(**I**), $h(a_i) = a_1$ *and* $h(a_j) = a_2$, *if and only if* $D^* \models e_1$ *and there is a nontrivial A-EGD in D.*

PROOF. The arguments are very similar to those in the proofs of Theorems 1, 4, and 8, and are left to the reader. □

Obviously, the conditions of the theorem are not effectively testable for infinite chases. In practice, we can replace in the theorem "chase (**I**)" by "some I_n in **I**." To test implication we compute the relations of the chase until the conditions of the theorem are satisfied. If $D \models d$, then we are bound to get a positive answer. But if $D \not\models d$, then we may chase forever.

4.3. SOLVABLE CASES. As noted before, the implication problem for TGDs is recursively unsolvable. In this section we describe several solvability results.

In some cases solvability follows from the fact that the answer to the implication problem is trivially negative.

LEMMA 13. *In the following cases $D \models d$ if and only if d is trivial:*

(1) *D is a set of TGDs and d is an EGD.*
(2) *D is a set of partial TGDs and d is a TTGD.*

PROOF

(1) The claim generalizes Lemma 7 and follows from Theorem 9.
(2) Let D be a set of partial TGDs and d be a TTGD $\langle w, I\rangle$. Suppose that d is not trivial; that is, $w \notin I$. Then $D \models d$ iff for some chase $\mathbf{I}$ of I by D we have $w \in$ chase($\mathbf{I}$). But for every TGD $\langle J', J\rangle \in D$ and a tuple $u \in J'$, there is an attribute A such that $w[A] \notin J[A]$. Thus, for every valuation h on J, if h' is a distinct extension of h to J', then $h'(u) \neq w$. It is clear that no application of a T-rule can produce w, so $w \notin$ chase($\mathbf{I}$) and $D \not\models d$. □

Though in general chases by TGDs are infinite, in some cases they are necessarily finite. An attribute A is *partial* in a TGD $\langle \mathrm{I}', \mathrm{I}\rangle$ if $I'[A] \not\subseteq I[A]$. It is partial in a set D of dependencies if it is partial in some TGD in D. Let d be the TGD $\langle J, I\rangle$. We define a binary relation $\equiv_d$ on the tuples of J as follows:

(1) $u \equiv_d u$
(2) If for some attribute A, $u[A] = v[A] \notin I[A]$, then $u \equiv_d v$.
(3) If $u \equiv_d v$ and $v \equiv_d w$, then $u \equiv_d w$.
(4) $u \equiv_d v$ only if it so follows from application of the above clauses.

Clearly, $\equiv_d$ is an equivalence relation.

LEMMA 14. *Let $d = \langle J, I\rangle$ be a TGD and $J_1, \ldots, J_k$ be the partition of J to equivalence classes as induced by $\equiv_d$, then $\langle J, I\rangle \models \rightrightarrows \{\langle J_i, I\rangle \mid 1 \leq i \leq k\}$.*

PROOF. To see that the first direction of the claimed equivalence holds, note that $\langle J_i, \langle J, I\rangle(I)\rangle$ is trivial, because $J_i \subseteq J \subseteq \langle J, I\rangle(I)$. So $\langle J, I\rangle \models \langle J_i, I\rangle$. We now prove the other direction. Let $D = \{\langle J_i, I\rangle \mid 1 \leq i \leq k\}$. Let h be the identity valuation on I. We can construct a valuation h' that is a distinct extension of h to J and also a distinct extension to all of the J_i's because if $u \in J_i$ and $v \in J_j$, $i \neq j$, then $u[A] = v[A]$ entails $u[A] \in I[A]$. It follows that there are a chase $\mathbf{I}$ of I by D and a valuation g that is the identity on I and $g(J) \subseteq$ chase($\mathbf{I}$), so $D \models \langle J, I\rangle$. □

THEOREM 10. *Let D be a set of dependencies such that at most one attribute A is partial in D, and let I be a finite relation. Then all chases of I by D are finite.*

PROOF. Let $\langle J', J\rangle \in D$. By Lemma 14, we can assume without loss of generality that $| J'[A] | = 1$. A T-rule for a TGD $\langle J', J\rangle \in D$ produces tuples with new values if A is partial in $\langle J', J\rangle$, and for some relation I_n in the chase, $h(J) \subseteq I_n$, but for all extension h' of h to J', $h'(J') \not\subseteq I_n$. However, for all attributes $B \in \bar{A}$, $J'[B] \subseteq J[B]$; so h' agrees with h on $J'[B]$. It can not be the case that there is a relation $K \subseteq I_n$ such that $h(J)[\bar{A}] = K[\bar{A}]$ and $| K[A] | = 1$, because then we can define $h'(J')[A] = K[A]$ and have $h'(J') \subseteq I_n$. However, we have such $K \subseteq I_{n+1}$, because in applying $\langle J', J\rangle$ we add $h'(J')$ to I_n. Clearly, this can happen only a finite number of times, so only a finite number of new values can be added. It follows that all chases are finite. □

Our next result concerns *template dependencies* (TD). A TD is a TGD $\langle I', I\rangle$, where $| I' | = 1$. Note that a TD must be either total or partial. Template dependencies were introduced by Sadri and Ullman, who also developed the chase as a proof procedure for functional and template dependencies [43, 44]. The implication problem for TDs was shown to be recursively unsolvable in [29] and [50]. We now

prove a solvability results for TDs. The idea is that we can force finiteness of the chase by applying the rules in a specific order.

For each TD $d = \langle \{w\}, I \rangle$ we define an attribute set TOTAL(d) = $\{A \mid w[A] \in I[A]\}$. Let d be a TGD $\langle \{w\}, I \rangle$, X = TOTAL(d), and $Y \subseteq X$. Define w_Y as a tuple such that $w_Y[Y] = w[Y]$, and all other values of w_Y are new values; and define w'_Y as a tuple such that $w'_Y[X] = w[X]$, and $w'_Y[\bar{X}] = w_Y[\bar{X}]$. We associate with d a TTGD $d_Y = \langle w'_y, I \cup \{w_Y\} \rangle$.

Example 5. Let d be the TD $\langle \{w, I\} \rangle$:

	A	B	C	D	E	F
w:	$a0$	$b0$	$c0$	$d0$	$e0$	$f0$
	$a0$	$b0$	$c1$	$d0$	$e1$	$f1$
I:	$a0$	$b1$	$c0$	$d1$	$e0$	$f2$
	$a1$	$b0$	$c0$	$d2$	$e2$	$f3$

TOTAL(d) is $\{A, B, C, D, E\}$. Let $Y = \{A, B, C\}$. Then d_Y is

	A	B	C	D	E	F
w:	$a0$	$b0$	$c0$	$d0$	$e0$	$f4$
	$a0$	$b0$	$c1$	$d0$	$e1$	$f1$
	$a0$	$b1$	$c0$	$d1$	$e0$	$f2$
I:	$a1$	$b0$	$c0$	$d2$	$e2$	$f3$
	$a0$	$b0$	$c0$	$d3$	$e3$	$f4$

THEOREM 11. *Let D be a set of EGDs and TDs, and let Y be an attribute set such that, for all TDs $d \in D$, we have $Y \subseteq$ TOTAL(d) and $D \models d_Y$. Then we can construct a set D' of dependencies such that $D' \models\!\models D$ and for any relation I there is a finite chase of I by D'.*

PROOF. Let $D' = D \cup \{d_Y \mid d$ is a TD in $D\}$. Clearly, $D' \models\!\models D$. We now show that there is a finite chase of I by D'. To this end, before applying a TD $d \in D$, we apply d_Y as long as it is applicable; that is, if we apply a TD $d = \langle \{w\}, J \rangle$ to I_n, then $\langle w'_Y, J \cup \{w_Y\} \rangle(I_n) = I_n$. Suppose that $t \in I_{n+1} - I_n$; that is, there is a valuation h on J such that $h(J) \subseteq I_n$, for all extensions h' of h to w, $h'(w) \notin I_n$, and $t = h''(w)$ where h'' is a distinct extension of h to w. We claim that $t[Y] \notin I_n[Y]$. Suppose not. Then for some $u \in I_n$, $t[Y] = u[Y]$. Then h can be extended to w_Y so that $h(w_Y) = u$ and $h(I \cup \{w_Y\}) \subseteq I_n$. It follows that $h(w'_Y) \in I_n$. But then we can define an extension h' of h to w so that $h'(w)[\bar{X}] = h(w'_Y)[\bar{X}]$, where X = TOTAL(d), and we have $h'(w) \in I_n$—contradiction. Now note that since $Y \subseteq$ TOTAL(d) for all TGDs $d \in D$, $I_n[A] \subseteq I[A]$ for all $A \in Y$. It follows that only a finite number of T-rule applications introduce new values. Hence, this chase is finite. □

COROLLARY. *Let D be a set of EGDs and TDs, and let Y be an attribute set such that for all partial TDs $d \in D$, Y = TOTAL(d). Then all chases by D are finite.*

PROOF. Let $d = \langle \{w\}, I \rangle$ be a TD in D. If d is total, then TOTAL(d) = U, so $w'_Y = w$ and $d_Y = d$. Otherwise, TOTAL(d) = Y, so $w_Y = w'_Y$ and d_Y is a trivial TGD. In both cases $D \models d_Y$. It follows that all chases by D are finite. □

The condition of Theorem 11 looks quite arbitrary and difficult to apply. The corollary, however, is a straightforward application of Theorem 11, and we now show another example in which the theorem can be applied quite straightforwardly.

Example 6. A *first-order hierarchical decomposition* (FOHD) [22] is a TTGD $\langle w, I\rangle$ satisfying

(1) for some $Y \subseteq U$, $w[Y] = u[Y]$ for all $u \in I$.
(2) for all attributes $A \in \bar{Y}$, and distinct tuples $u, v \in I$, $u[A] \neq v[A]$.

Let $\langle u, J\rangle$ be an FOHD, $I = \{u_1, \ldots, u_k\}$. Define $Y = \{A \mid J[A] = u[A]\}$, and $X_i = \{A \in \bar{Y} \mid u_i[A] = u[A]\}$, $1 \leq i \leq k$. Note that $Y, X_1, \ldots, X_k$ is a partition of U.[7] Let $d = \langle\{w\}, I\rangle$ be a TD such that TOTAL(d) is a union of Y and some of the X_i's. That is:

(1) $Y \subseteq$ TOTAL(d);
(2) for all $1 \leq i \leq k$, either $X_i \subseteq$ TOTAL(d) or $X_i \cap$ TOTAL(d) $= \emptyset$.

Let $D = \{\langle u, J\rangle, d\}$. We claim that $D \vDash d_Y$. Assume, without loss of generality, that for $1 \leq i \leq m$, $X_i \subseteq$ TOTAL(d), and for $m < i \leq k$, $X_i \cap$ TOTAL(d) $= \emptyset$, where $0 \leq m \leq k$. We show how to get w'_Y in a chase of $I \cup \{w_Y\}$ by D. Let h be a distinct extension of the identity valuation on I to w; by applying d to $K_0 = I \cup \{w_Y\}$ we get $t = h(w) \in K_1$, where t[TOTAL(d)] $= w$[TOTAL(d)]. Since $Y \subseteq$ TOTAL(d), $w_Y[Y] = t[Y]$. We can now define a valuation g on J so that $g(u_i) = t$ for $1 \leq i \leq m$, and $g(u_i) = w_Y$ for $m < i \leq k$, and we have that $g(u) = w'_Y$. So by applying $\langle u, J\rangle$ to K_1, we get w'_Y. It follows that if D is a set of dependencies such that $\langle u, J\rangle \in D$, and all TGDs in D satisfy conditions (1) and (2), then Theorem 11 is applicable. This entails solvability for the problem of testing preservation of dependencies under hierarchical decompositions [11, 32].

We have observed that the set $\{\langle D, d\rangle \mid D \vDash d\}$ is recursively enumerable. Thus, to show that it is recursive, it suffices to show that its complement $\{\langle D, d\rangle \mid D \nvDash d\}$ is also recursively enumerable. Let I be a relation and $w \in I$. The value $w[A]$ is *nonrepeating* in I, if for all other tuples $u \in I$, $u[A] \neq w[A]$. An attribute A is *nonrepeating* in I, if for all tuples $w \in I$, $w[A]$ is nonrepeating in I. For a set of TGDs D and a TGD $\langle I', I\rangle$ we define the following property:

(*) For each attribute A, if A is partial in D then:

(a) if $w \in I'$ and $w[A] \notin I[A]$, then $w[A]$ is nonrepeating in I';
(b) if $\langle J', J\rangle \in D$, then A is nonrepeating in J.

THEOREM 12. *The set* $\{\langle D, d\rangle \mid D \nvDash d$ *and* $\langle D, d\rangle$ *satisfies* (*)$\}$ *is recursively enumerable.*

PROOF. It suffices it to show that if $\langle D, d\rangle$ satisfies (*) and $D \nvDash d$, then there is a finite relation $K \in$ SAT(D) $-$ SAT(d). Let $d = \langle I', I\rangle$, and let **I** be a chase of I by D. By Lemma 12, chase(**I**) $\in$ SAT(D) $-$ SAT(d). For each attribute A, let $a' \in$ DOM(A) be some fixed value such that $a' \notin I[A]$. Let h be a valuation on chase(**I**) defined as follows: For all attributes A and values $a \in$ chase(**I**)$[A]$, if $a \in I[A]$, then $h(a) = a$; otherwise $h(a) = a'$. Clearly, h is the identity on I, $K = h$(chase(**I**)) is finite, and $I \subseteq K$.

CLAIM 1. $K \notin SAT(d)$.

Suppose that $K \in$ SAT(d). Let f and g be the identity valuation on I. Since $g(I) \subseteq K$, there is an extension g' of g to I' such that $g'(I') \subseteq K$. We extend f to I' in the following manner. Choose any tuple $w \in I'$ and an attribute $A \in U$. Since

[7] The notation [22] for $\langle u, J\rangle$ is in fact $Y \twoheadrightarrow X_1, \ldots, X_k$.

$g'(w) \in K = h(\text{chase}(\mathbf{I}))$, $g'(w) = h(u)$ for some $u \in \text{chase}(\mathbf{I})$. If $g'(w)[A] \notin I[A]$, then also $u[A] \notin I[A]$ and $w[A] \notin I[A]$. Since in this case A is partial in D, $w[A]$ is nonrepeating in I', and we can define $f(w[A]) = u[A]$. Finally, we get $f(I') \subseteq \text{chase}(\mathbf{I})$. By Theorem 9, $D \models \langle I', I\rangle$—contradiction.

CLAIM 2. $K \in SAT(D)$.

Let $\langle J', J\rangle \in D$, and let g be a valuation on J such that $g(J) \subseteq K$. We define a valuation f on J in the following manner. Choose any tuple $w \in J$ and an attribute $A \in U$. Since $g(w) \in K = h(\text{chase}(\mathbf{I}))$, $g(w) = h(u)$ for some $u \in \text{chase}(\mathbf{I})$. If $g(w)[A] \in I[A]$, then $f(w[A]) = g(w)[A]$, and if $g(w)[A] \notin I[A]$, then A is partial in D and $w[A]$ is nonrepeating in J; so we let $f(w[A]) = u[A]$. Thus, $f(J) \subseteq \text{chase}(\mathbf{I})$, and $h \circ f = g$. But since $\text{chase}(\mathbf{I}) \in \text{SAT}(D)$, there is an extension f' of f to J' such that $f'(J') \subseteq \text{chase}(\mathbf{I})$. $h \circ f'$ is an extension of g to J' and $h \circ f'(J') \subseteq h(\text{chase}(\mathbf{I})) = K$. □

Theorem 12 can be slightly modified to include EGDs as well.

Example 7. A *cross dependency* [14, 40, 48] is a TD $\langle\{w\}, I\rangle$ such that all attributes $A \in U$ are nonrepeating in I. (A cross dependency states that the relation is a cross product of some of its projections.) It is easy to see that if D is a set of cross dependencies and d is a cross dependency, then $\langle D, d\rangle$ satisfies (*). By Theorem 12, the implication problem for cross dependencies is solvable. □

We refer the reader to [30] and [46] for some other solvability results for TGDs.

5. *Concluding Remarks*

The framework established in the preceding sections enables us to study several possible extensions.

5.1. A MORE GENERAL DATABASE MODEL. The database model considered here is quite restricted. First, we assumed that the database consists of a single relation—the so-called *universal relation assumption* [8]. Also, we required that distinct attributes have disjoint domains. Namely, we assumed that our relations are *typed* or *many-sorted*. In practice, however, databases may consist of several relations, and distinct attributes need not have disjoint domains, for example, consider the attributes EMPLOYEE# and MANAGER#. Furthermore, one may wish to express constraints like "every manager is an employee." These constraints, called *inclusion dependencies* [24], are not expressible in our framework. It turns out that our framework can be generalized to the unrestricted model, and many of all our results carry over with minor modifications. We refer the reader to [1].

5.2. CONSTANTS. The values in our formalism are uninterpreted in the sense that they serve as variables. Thus, although we can express the constraint "a borrower can borrow at most one book" by the functional dependency BORROWER# → BOOK#, we cannot express the constraint "borrower #12345 can borrow at most one book". To this end, we associate with each attribute A a set of constants $\text{CON}(A) \subseteq \text{DOM}(A)$. For any valuation h and a constant $a \in \text{DOM}(A)$, we require $h(a) = a$. Although dependencies without constants have the property that, for any relation I and a set of dependencies D, I can be "chased" by D into $\text{SAT}(D)$, this is not the case for dependencies with constants, since it may happen that an application of an E-rule requires the identification of two constants.

5.3. BOUNDED DOMAINS. In practice some domains may be bounded, not only practically but also conceptually, for example, | DOM(SEX) | = 2. When dealing with EGDs and TTGDs, this poses no problem. Our T-rule of Section 4 presupposes, however, that in each domain there is an infinite supply of values. To account for the boundedness of a domain, we should be careful not to introduce more values than available. (Clearly, if all domains are bounded, then the implication problem is solvable.) Implication of join dependencies with bounded domains is treated in [24] and [31].

5.4. DISJUNCTIVE DEPENDENCIES. Recently, it has been suggested that dependencies with disjunctive conclusion may be of interest [5, 45]. For example, a *disjunctive functional dependency* is a statement $X \rightarrow Y + Z$. It is satisfied by a relation I, if for all tuples $t_1, t_2 \in I$, if $t_1[X] = t_2[X]$, then either $t_1[Y] = t_2[Y]$ or $t_1[Z] = t_2[Z]$. The domain boundedness constraint discussed above can also be expressed as a disjunctive dependency. Our formalism can be generalized to include disjunctive dependencies. For example, a disjunctive TTGD is a pair $\langle I', I\rangle$ of finite relations I and I' such that $\mathrm{VAL}(I') \subseteq \mathrm{VAL}(I)$. It is satisfied by a relation J if for any valuation h, such that $h(I) \subseteq J$, there is a tuple $w \in I'$ such that $h(w) \in J$. We can also generalize the chase to disjunctive dependencies by treating it as a tree search rather than as a straight-line search.

5.5. FINITE IMPLICATION. Since a database is inherently finite, there is a strong justification to define a relation as a finite set of tuples. We say that a set of dependencies D *finitely implies* a dependency d, denoted $D \models_f d$, if d is satisfied by every finite relation that satisfies all dependencies in D. The finite implication problem is to decide, for a given D and d, whether $D \models_f d$. Unfortunately, the finite implication problem for TGDs is recursively unsolvable [13, 19, 49]. Furthermore, it is not even partially solvable, which means that there can be no proof procedure for finite implication. However, for all solvable cases dealt with in this paper, we actually have shown that implication and finite implication coincide.

5.6. INFINITE SETS OF DEPENDENCIES. Throughout, we have assumed that D is a finite set, an assumption that can be justified on practical grounds. It turns out that there is a theoretical interest in characterizing implication by infinite sets of dependencies, for example, in dealing with projection of TGD classes (see [25]). Looking carefully at our definition of a nonterminating chase in Section 4.2, we see that we have not used the finiteness of D. Thus, the chase is also a proof procedure of implication from infinite sets of dependencies.

5.7. FORMAL SYSTEMS. A lot of effort has been devoted to the development of formal systems for dependencies [4, 5, 9, 17, 18, 35, 38–40, 47, 48]. Formal systems enjoy several advantages over the chase. A formal system allows us to infer new dependencies from given dependencies, whereas the chase only allows us to check if a dependency is implied by given dependencies, but it does not tell us how to generate new dependencies. It turns out that the chase is very useful in developing sound and complete formal systems. In [15] we develop several formal systems for TGDs and EGDs.

5.8. DEPENDENCIES AS FIRST-ORDER SENTENCES. We have already noted that dependencies can be expressed as first-order sentences. Thus, one may wonder about the relationship between our proof procedure, the chase, and known proof procedures for first-order logic. It turns out that there is indeed a very strong

connection between our procedure and the procedure of resolution and paramodulation [20]. This connection will be described in a future paper.

ACKNOWLEDGMENTS. We appreciate helpful comments from the two anonymous referees.

REFERENCES

1. ABITEBOUL, S., AND VARDI, M. Y. Formal systems for untyped data dependencies. To appear.
2. AHO, A. V., BEERI, C., AND ULLMAN, J. D. The theory of joins in relational databases. *ACM Trans. Database Syst. 4*, 3 (Sept. 1979), 297–314.
3. AHO, A. V., SAGIV, Y., AND ULLMAN, J. D. Equivalence among relational expressions. *SIAM J. Comput. 8* (1979), 218–246.
4. ARMSTRONG, W. W. Dependency structure in data base relationships. In *Proceedings of IFIP 74*, Elsevier North-Holland, New York, 1974, pp. 580–583.
5. ARMSTRONG, W. W., AND DELOBEL, C. Decompositions and functional dependencies in relations. *ACM Trans. Database Syst. 5*, 4 (Dec. 1980), 404–430.
6. BEERI, C. On the membership problem for functional and multivalued dependencies in relational databases. *ACM Trans. Database Syst. 5*, 3 (Sept. 1980), 241–259.
7. 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), 30–59.
8. BEERI, C., BERNSTEIN, P. A., AND GOODMAN, N. A sophisticate's introduction to database normalization theory. In *Proceedings of the 4th International Conference on Very Large Data Bases* (West Berlin, Germany, Sept. 13–15). ACM, New York, 1978, pp. 113–124.
9. BEERI, C., FAGIN, R., AND HOWARD, J. H. A complete axiomatization for functional and multivalued dependencies in database relations. In *Proceedings of the 3rd ACM-SIGMOD International Conference on Management of Data* (Toronto, Ontario, Canada, Aug. 3–5). ACM, New York, 1977, pp. 47–61.
10. BERRI, C., MENDELZON, A. O., SAGIV, Y., AND ULLMAN, J. D. Equivalence of relational database schemes. *SIAM J. Comput. 10* (1981), 647–656.
11. BEERI, C., AND RISSANEN, J. Faithful representation of relational database schemes. IBM Res. Rep. IBM, San Jose, 1980.
12. BEERI, C., AND VARDI, M. Y. On the complexity of testing implication of data dependencies. Res. Rep. Dept. Computer Science, The Hebrew University of Jerusalem, Jerusalem, Israel, 1980.
13. BEERI, C., AND VARDI, M. Y. The implication problem for data dependencies. In *Proceedings of the 8th International Colloquium for Automata Languages and Programming* (Acre, Israel, July 13–17). Lecture Notes in Computer Science, vol. 115. Springer-Verlag, New York, 1981, pp. 73–85.
14. BEERI, C., AND VARDI, M. Y. On the properties of join dependencies. In *Advances in Database Theory*, H. Gallaire, J. Minker, and J. M. Nicolas, Eds. Plenum Press, New York, 1981, pp. 25–72.
15. BEERI, C., AND VARDI, M. Y. Formal system for tuple and equality generating dependencies. *SIAM J. Comput. 13* (1984), 76–98.
16. BERNSTEIN, P. A. Synthesizing third normal form relations from functional dependencies. *ACM Trans. Database Syst. 1*, 4 (Dec. 1976), 277–298.
17. BISKUP, J. On the complementation rule for multivalued dependencies in data base relations. *Acta Inf. 10* (1978), 297–305.
18. BISKUP, J. Inferences of multivalued dependencies in fixed and undetermined universe. *Theor. Comput. Sci. 10* (1980), 93–105.
19. CHANDRA, A. K., LEWIS, H. R., AND MAKOWSKY, J. A. Embedded implicational dependencies and their inference problem. In *Proceedings of the 13th Annual ACM Symposium on the Theory of Computing* (Milwaukee, Wisc., May 11–13). ACM, New York, 1981, pp. 342–354.
20. CHANG, C. L., AND LEE, C. R. T. *Symbolic Logic and Mechanical Theorem Proving*. Academic Press, New York, 1973.
21. CODD, E. F. Further normalization of the data base relational model. In *Data Base Systems* (R. Rustin, Ed.). Prentice-Hall, Englewood Cliffs, N.J., 1972, pp. 33–64.
22. DELOBEL, C. Normalization and hierarchal dependencies in the relational data model. *ACM Trans. Database Syst. 3*, 3 (Sept. 1978), 201–222.
23. FAGIN, R. Multivalued dependencies and a new normal form for relational databases. *ACM Trans. Database Syst. 2*, 2 (June 1977), 262–278.

24. FAGIN, R. A normal form for relational databases that is based on domains and keys. *ACM Trans. Database Syst. 6*, 3 (Sept. 1981), 387–415.
25. FAGIN, R. Horn clauses and database dependencies. *J. ACM 29*, 4 (Oct. 1982), 952–985.
26. FAGIN, R., MAIER, D., ULLMAN, J. D., AND YANNAKAKIS, M. Tools for template dependencies. *SIAM J. Comput. 12* (1983), 36–59.
27. GRAHAM, M. H. A new proof that the chase is a Church–Rosser replacement system. In *Proceedings of the XP1 Workshop on Relational Database Theory* (Stony Brook, June) 1980.
28. GRANT, J., AND JACOBS, B. E. On the family of generalized dependency constraints. *J. ACM 29*, 2 (Oct. 1982), 986–997.
29. GUREVICH, Y., AND LEWIS, H. R. The inference problem for template dependencies. In *Proceedings of the ACM Symposium on Principles of Database Systems* (Los Angeles). ACM, New York, 1982, pp. 221–229.
30. ITO, M., TANIGUCHI, K., AND KASAMI, T. Membership problem for embedded multivalued dependencies under some restricted conditions. *Theor. Comput. Sci. 23* (1983), 175–194.
31. KANELLAKIS, P. C. On the computational complexity of cardinality constraints in relational databases. *Inf. Proc. Lett. 11* (1980), 98–101.
32. MAIER, D., MENDELZON, A. O., SADRI, F., AND ULLMAN, J. D. Adequacy of decompositions of relational databases. In *Advances in Database Theory*, H. Gallaire, J. Minker, and J. M. Nicolas, Eds. Plenum Press, New York, 1981, pp. 101–114.
33. MAIER, D., MENDELZON, A. O., AND SAGIV, Y. Testing implications of data dependencies. *ACM Trans. Database Syst. 4*, 4 (Dec. 1979), 455–469.
34. MAIER, D., SAGIV, Y., AND YANNAKAKIS, M. On the complexity of testing implications of functional and join dependencies. *J. ACM 28*, 4 (Oct. 1981), 680–695.
35. MENDELZON, A. O. On axiomatizing multivalued dependencies in relational databases. *J. ACM 26*, 1 (Jan. 1979), 37–44.
36. NICOLAS, J. M. First order logic formalization for functional, multivalued and mutual dependencies. In *Proceedings of the ACM-SIGMOD International Conference on Management of Data* (Austin, Tex., May 31–June 2). ACM, New York, 1978, pp. 40–46.
37. NICOLAS, J. M. Mutual dependencies and some results on undecomposable relations. In *Proceedings of the 4th International Conference on Very Large Data Bases* (West Berlin, Germany, Sept. 13–15). ACM, New York, 1978, pp. 360–367.
38. PAREDAENS, J. Transitive dependencies in a database scheme. *RAIRO Inf./Comput. Sci. 14* (1980), 149–164.
39. PAREDAENS, J. The interaction of integrity constraints in an information system. *J. Comput. Syst. Sci. 20* (1980), 310–329.
40. PAREDAENS, J. A universal formalism to express decomposition, functional dependencies and other constraints in a relational database. *Theor. Comput. Sci. 19* (1982), 143–160.
41. PAREDAENS, J., AND JANSSENS, D. Decompositions of relations—A comprehensive approach. In *Advances in Database Theory*, H. Gallaire, J. Minker, and J. M. Nicolas, Eds. Plenum Press, New York, 1981, pp. 73–100.
42. RISSANEN, J. Theory of relations for databases—A tutorial survey. In *Proceedings of the 7th Symposium on Mathematical Foundations of Computer Science* (Zakopane, Poland, Sept. 4–8) Lecture Notes in Computer Science, vol. 64. Springer-Verlag, New York, 1978, pp. 537–551.
43. SADRI, F., AND ULLMAN, J. D. Template dependencies: A large class of dependencies in relational databases and its complete axiomatization. *J. ACM 29*, 2 (Apr. 1982), 363–372.
44. SADRI, F., AND ULLMAN, J. D. The theory of functional and template dependencies. *Theor. Comput. Sci. 17* (1982), 317–332.
45. SAGIV, Y., DELOBEL, C., PARKER, D. S., AND FAGIN, R. An equivalence between relational database dependencies and a fragment of propositional calculus. *J. ACM 28*, 3 (July 1981), 435–453.
46. SAGIV, Y., AND WALECKA, S. F. Subset dependencies and a completeness result for a subclass of embedded multivalued dependencies. *J. ACM 29*, 1 (Jan. 1982), 103–117.
47. SCIORE, E. A complete axiomatization of full join dependencies. *J. ACM 29*, 2 (Apr. 1982), 373–393.
48. VARDI, M. Y. Axiomatization of functional and join dependencies in the relational model. M.Sc. Thesis, Dept. Applied Mathematics, Weizmann Institute of Science, Rehovot, Israel, 1980.
49. VARDI, M. Y. The implication problem for data dependencies in relational databases. Ph.D. Thesis (in Hebrew), The Hebrew University of Jerusalem, Jerusalem, Israel, 1981.
50. VARDI, M. Y. The implication and the finite implication problems for typed template dependencies. *J. Comput. System Sci. 28*, 1 (Feb. 1984), 3–28.

51. VARDI, M. Y. On the decomposition of relational databases. In *Proceedings of the 23rd IEEE Symposium on the Foundations of Computer Science*, (Chicago, Nov. 3–5). IEEE, New York, 1982, pp. 176–187.
52. VARDI, M. Y. Inferring multivalued dependencies from functional and join dependencies. *Acta Inf. 19* (1983), 305–324.
53. VARDI, M. Y. Inferring multivalued dependencies from tuple and equality generating dependencies. To appear.
54. YANNAKAKIS, M., AND PAPDIMITRIOU, C. Algebraic dependencies. *J. Comput. System Sci. 21* (1982), 2–41.
55. ZANIOLO, C. Analysis and design of relational schemata for database systems. Tech. Rep. UCLA-ENG-7769, Dept. Computer Science, UCLA, Los Angeles, July 1976.

RECEIVED FEBRUARY 1981; REVISED APRIL 1984; ACCEPTED APRIL 1984