ON LOSSLESS TRANFORMATION OF DATABASES SCHEMES NOT NECESSARILY SATISFYING UNIVERSAL INSTANCE ASSUMPTION

(preliminary version)

Tomasz Imielinski*
Computer Science Department
Rutgers University
New Brunswick,NJ 08903

and

Nicolas Spyratos
Universite de Paris-Sud
Centre d'Orsay
Laboratorie de Recherche en Informatique

ABSTRACT

Given a multirelational database scheme and a relational mapping f transforming it,an important question is whether the resulting scheme is equivalent to the original one. This question was addressed in the literature with respect to those relational schemes that satisfy the so called universal relation assumption; however, no study was ever concerned with multi-relational (data base) schemes that do not necessarily satisfy this assumption.

We present two general definitions of lossless transformation of the database scheme based on the so-called closed world and open world assumptions. While both definitions seem to be practically justified, the one based on the open world assumption is more "tractable" .We are able to test losslessness defined in such a way for a wide class of relational expressions and dependencies. An algorithm for testing losslessness of a mappings (which are arbitrary relational expressions built up from projections, cartesian products and restrictions) is presented in the paper. Moreover, given a lossless transformation, our algorithm enables us to explicitly construct an "inverted" mapping that restores the corresponding state of the original database. The application of the algorithm to schemes specified by differrent types of dependencies is described. In particular the application of the algorithm for schemes specified by inclusion dependencies is presented.In this case the algorithm works for families of inclusion dependencies having finite chase property.This class of inclusion dependencies is characterized in the paper.

* On leave from the Institute of Computer Science Polish Academy of Sciences

INTRODUCTION

The problem of lossless transformations of a database schemes has been extensively studied in the literature (see for example [U]). This is not surprising since the problem is of great practical importance. For example, the whole database design process can be viewed as a process of consecutive transformations of an input database scheme. The losslessness of these transformations should always be maintained in order to guarantee that what we obtain as the result of the design process is in fact a different form of the same database.

Most frequently the transformations under study were restricted to simple projections. The join operator was, in turn, used to restore the original database state from the new scheme.

Intuitively,we say that $f = \langle f_1, \ldots f_n \rangle$ a sequence of relational expressions is a lossless transformation of database scheme P if there exists some transformation g, such that $p = g(f(p))$ for any instance p of P. In other words, if there exists any transformation enabling us to restore the original state p from $\langle f_1(p) \ldots f_n(p) \rangle$, then this transformation must be a join.

This is a very convenient situation, since in order to test whether $\langle f_1 \ldots f_n \rangle$ is lossless we do not have to examine all possible transformations and check whether they restore the original database state; it suffices to examine the corresponding join.

Unfortunately, this is no longer true when the database scheme P is built up from more than one relational scheme and does not necessarily satisfy the universal relation assumption. Given a transformation $\langle f_1 \ldots f_n \rangle$, we have no "hints" about how the "inversion" restoring the original database state might look like (it need not even be a relational expression).

ndeed, suppose, for example, that P = <P_1(A,B,C), P_2(B,C,D), P_3(A,D,E)> where P_1, P_2, P_3 are relational schemes defined over ABC, BCD, and ADE respectively and let

$f = \langle f_1, f_2, f_3 \rangle$ where

$f_1 = \pi_{ACD}(P_1 \bowtie P_2)$

$f_2 = \pi_{ABE}(P_1 \bowtie P_3)$

$f_3 = \pi_{BCD}(P_2 \bowtie P_3)$

Then we do not have a definite candidate to restore the original state p of P from

$f(p) = \langle f_1(p), f_2(p), f_3(p) \rangle$

The candidate for inversion mapping could potentially look like

$g = \langle \pi_{ABC}(f_1(p) \bowtie f_2(p) \bowtie f_3(p)),$
$\pi_{BCD}(f_1(p) \bowtie f_2(p) \bowtie f_3(p)),$
$\pi_{ABE}(f_1(p) \bowtie f_2(p) \bowtie f_3(p)) \rangle$

but it could also look like

$g = \langle \pi_{AC}(f_1(p)) \bowtie \pi_{AB}(f_2(p)),$
$\pi_{CD}(f_1(p)) \bowtie \pi_{BC}(f_3(p)),$
$\pi_{BCD}(f_2(p) \bowtie f_3(p)) \rangle$

We may find some other reasonable candidates if we do not restrict ourselves to PJ-expressions, we can find even more of them.

It is clear that the intuitive definition of a lossless transformation, bansed on existence of some "inversion" mapping restoring the original state of the database, is not acceptable from the practical point of view.

Indeed in order to prove that a given transformation f is lossless we must find an "inversion" g.This is only way of proving its existence. Apparently we need some other formalization which would meet our intuitions and the same time would make the property of being lossless effectively decidable by an algorithm.

The existing literature lacks such a general approach,which would cover both single relational and multi-relational schemes. To our knowledge there is no study in the literature which concerns itself with lossless transformations of multi-relational schemes, not necessarily satisfying U.I.A (which are most frequent in practice).The existing research is restricted to decompositions of a single relational scheme,or schemes satisfying the U.I.A.

In this paper we are going to formulate two definitions of lossless transformations based on two different logical interpretations of a relational database, namely so-called open world and closed world assumptions.

Both definitions are somewhat natural but one based on the open world assumption is more "constructive".We are able to test it efficiently. The algorithm for testing losslessness of a transformation which are well formed arbitrary relational expression built up from projections and joins is presented in the paper. This algorithm,given a state of the new database scheme obtained by the lossless transformation, enables us to restore the corresponding original state of the database scheme. Therefore,in the case of a lossless transformation, we can construct inversion mapping explicitly.

1. BASIC NOTIONS

1.1 Relations and Tables

A relation r is a collection of functions (tuples) from a set of attributes (X) to some domain (D). For notational smplicity we assume one common (infinite) domain. By the type of a relation r we mean its set of attributes $\alpha(r)$. By a relation name we mean a symbol R with an associated type. We will say that relation r is the instance of relation name R if the type of r is equal to the type of R. We consider the usual relational operators: projection,cartesian product, restriction,union and selection (denoted by $\pi_Y(r)$, r s,$\sigma_{A=B}(r)$,rUs,$\pi_E(r)$ respectively). For any subset of relational operators such that project (P),cartesian product (C), restriction (R), selection (S) andunion (U) ,by a relational Ω- expression) we mean any well formed expression built up from relation names and the relational operators in Ω.For example ,a PCR-expression is built up using projection,cartesian product and restriction.

Further in the text we deal with "positional" version of relational algebra [Ul],defined by reffering to the columns of relations by numbers, not by names of attributes. In this way we avoid operation of renaming,needed for example to define cartesian product.

A database scheme P is the collection of relational names $P_1,..., P_n$ together with a set of dependencies. The instance of the database scheme is a sequence of relations $\langle r_1,...,r_n \rangle$ and it is called a multirelation. The number n of relations in a multirelation will be called an the index of the multirelation.The expression f will be called monotonic iff for every relations (multirelations) r,s $r \subseteq s \Rightarrow f(r) \subseteq f(s)$.

A table [IL1] is a relation with variables as well as constants allowed as entries. Let V be a countably infinite set of symbols called variables. Bya table of type X we mean any finite collection of tuples (i.e. functionsfrom X into D ∪ V).

By a valuation, we mean any mapping v:V→ D. A valuation can be extended over the set D (v(c)=c for all c in D) and over the set of tuplesand tables in the standard way.By rep (T) we will mean the function assigning to each table T the following set of relations:

{s : there exists v such that v(T)⊆s}

By a multitable we shall mean a sequence of tables T = $\langle T_1,... T_n \rangle$. Tables and multitables are just relations and

multirelations with null values. The variables play the role of indexed null values. Therefore, it is possible to represent the fact that the same two values, although unknown, are the same. By a minimal multitable T^i we will understand the multitable $<T_1 \dots T_n>$ such that $T_j = \phi$ for all $j \neq i$ and $T_i = \{t\}$, where t is the tuple built up only from variables. There are n different (up to renaming the variables) minimal multitables of an index n.

Dependencies

We will deal here with implicational dependencies defined as T/t, where T is a table and t is a tuple such that all the variables that occur in t also occur in T; furthermore, no symbol occurs in two different columns or $T/(e_1 = e_2)$,where e_1 and e_2 are two symbols suchthat at least one of them is a variable occurring in T.

Inclusion dependencies [CFP] will also play an important role in this paper. By a family of dependencies with finite chase we will mean any family Σ of implicational or inclusion dependencies such that $chase_\Sigma(T)$ is finite for any finite T. By $chase_\Sigma(T)$ we understand here the result,possibly infinite, of the well known process, with rules corresponding to implicational and inclusion dependencies,the latter ones requiring generation of additional tuples with new variables.

The Universal Instance Assumption

We will say that the instance $<r_1,\dots,r_n>$ of the scheme $R=<R_1,\dots,R_n>$ satisfies Universal Instance Assumption(UIA) iff there exist (universal) relation r over the set of all attributes occuring in the scheme such that each of relations r_i is a projection of r.

The Open World and the Closed World Assumptions

Both Closed World (CWA) and Open World(OWA) assumptions are two different ways of looking at a relational instance from a logical point of view. Under the CWA we treat the tuples not belonging to a relation r as expressing negative relationship, while under the OWA we admit the lack of knowledge about the "real" status of those tuples. In consequence, under the OWA we do not really know which of the relations, among those containing database state s, corresponds to the real one. This gives rise to the family of candidate relations denoted by rep(s) and defined as rep(s) = {r: r⊃s}.

Inversions of Relational Expressions

Let T be a (multi) table and let f be a relational expression such that the type of the result of (the set of altributes) is equal to the type of T.Let $f^{-1}(rep(T))$ denote the set of relations (multirelations) s such that $f(s) \in rep(T)$. In [IL1] it was proved thatif f is a relational expression built up from projections, cartesian products and restrictions, then there exists a table (multitable) such that $rep(U) = f^{-1}(rep(T))$. In [IL] the algorithm constructing thistable was given. Let us denote the table resulting from this algorithmby $f^{-1}(T)$. If $f = <f_1 \dots f_n>$ and $T = <T_1 \dots T_n>$ is the multitable with the type corresponding to the type of f, then

$$f_i^{-1}(rep(T)) = n^n_{i=1} f_i^{-1}(rep(T_i)) .$$

and again by [IL1],there exists(multitable) such that

$$rep(U) = f^{-1}(rep(T)) .$$

2. Local Properties

We introduce here the notion of locality (or k-locality) since it will be useful in further considerations.

Definition

Let $Q = \forall_R F(R)$ be some universal property, where R ranges over relations or multirelations. We say that Q is k-local iff

$$\forall_R F(R) \Leftrightarrow \forall_R (||R|| \leq k \Rightarrow F(R)) ,$$

where $||R||$ denotes the number of tuples in R. The k-locality reduces the infinitness of the problem to the finite, possibly tractable dimension and is certainly a desirable property. An example of a k-local property is the equivalence of two relational expressions not involving the difference operator. As shown in [IL3] for any two expressions f,g not involving the difference

$$\forall_R f(R) = g(R) \Leftrightarrow \forall_{R:\,||R|| \leq k} f(R) = g(R)$$

(k depends on the complexity of f and g).

It turns out that locality of many properties depends strictly on the general logical assumptions which are made about the database, more specifically, on whether OWA or the CWA is made. Generally speaking, the OWA is computationally more tractable than the CWA; for example, many properties become k-local under OWA.

The notion, which in many situations under the OWA implies k-locality, is the notion of distributivness defined as follows:

Definition

A set $\mathcal{S}$ of relations (multirelations) is distributive (k-distributive)if there exists k>0 such that for any $S \in \mathcal{S}$

$$S = \cup \{Q \in \mathcal{S} : Q \subseteq S \wedge ||Q|| \leq k\}$$

In the case of multirelations S, the cardinality $||\ ||$ is understood component-wise; i.e., if $S=<S_1,\dots S_n>$ then $k=<k_1,\dots k_n>$ and $||S||$ smaller than k means $||S_i|| \leq k_i$ for $i \in \{1\dots n\}$. k-distributivenesss frequently implies k-locality under the OWA. It will be the case in our problem.

3. Lossless transformations

Generally speaking, a transformation (function) f is lossless iff it is one to one, or, in other words, if $f^{-1}(f(x))=\{x\}$ for any x belonging to the domain of f. In this paper, we will adopt this definition, thus restricting ourselves to relational mappings f or sequences of them. The possibility of different definitions of losslessness for a given mapping f will be related to the different notions of database state, i.e., different structures of the domain of the function f.

The two different assumptions (CWA and OWA) lead to essentially different notions of a database state. While under the CWA the state of the database is simply a relation, under the OWA it is the set of relations (multirelations). In consequence, under the CWA the mapping f is the mapping between the relations, while under the OWA it is the mapping between sets of them.

Let $\mathcal{S}$ be a set of relational (multirelational) instances of some databasescheme. This set is usually defined by some finite set of dependencies. A transformation f is a lossless transformation of under the CWA iff: (1) For every s in the set of relations in that are mapped onto f(s) under f is equal to {s}. Formally, (1) could be restated as

$$(1')\quad \forall_{S\in\mathcal{S}}\ f^{-1}(f(S))\cap\mathcal{S}=\{S\}$$

We can repeat the same reasoning under the OWA obtaining the following definition of the lossless transformation under the OWA. (2) For every s in , the set of relations in that are mapped onto rep(f(s)) is equal to rep(s) Formally,

$$(2')\quad \forall_{S\in\mathcal{S}}\ f^{-1}(rep(f(S)))\cap\mathcal{S}= rep(S)\cap\mathcal{S}$$

In other words, under the OWA, we would again like to be able to restore the original state from the new schema. The only difference lies in the definition of this state.

In both definitions we do not specify how the mapping f and its inversion should look like. In this sense, our definitions are general. Further in the paper f will be understood as the sequence of relational expressions defining some new multirelational schema.

Again, both definitions can be justified. The definition based on the OWA is, however, (as all the definitions based on the OWA) more constructive as we will show further in the paper. It is easy to prove that the OWA losslessness implies the CWA losslessness as the following result shows.

Fact 1 For any monotone f,if f is lossless in the OWA sense then it is lossless in the CWA sense. Proof

$$f^{-1}(rep(f(s)))\cap\mathcal{S}=rep(s)\cap\mathcal{S}\ \Leftrightarrow\ \forall_q\ (f(s)\subset f(q)\Rightarrow s\subset q)$$

Now suppose that f is lossless in the OWA sense and that

$$f(s)=f(q),\ \text{then}\ f(q)\supset f(s)\ \text{and}\ f(s)\supset f(q),$$

But according to (1)

$$f(s)\supset f(q)\Rightarrow s\supset q$$
$$\text{and}\ f(s)\subseteq f(q)\Rightarrow s\subseteq q,$$
$$\text{hence}\ s=q\ \text{and}$$
$$f\ \text{is lossless in the CWA sense.}$$

In general, the converse is not true as can be easily shown.Indeed,for any f take $\mathcal{S}=\{s_1,s_2\}$ such that $f(s_1)\subsetneq f(s_2)$, $s_1\notin s_2$ nor $s_2\notin s_1$; then f is CWA lossless but not OWA lossless. One the other hand, we could not find (any "reasonable" set of database instances, i.e., those corresponding to some standard dependencies) an example of any transformations f that would be CWA lossless but not OWA lossless.

Our conjecture is that the OWA losslessness closely approximates the CWA losslessness

The result of Fact 1 is important, since even if somebody does not accept the OWA interpretation, the positive result of testing OWA losslessness, implies CWA lossless. We are now going to characterize the OWA losslessness more closely, using the notion of k-distributivity.

Theorem 1

Let $\mathcal{S}$ be a k-distributive set of database instances and let $f=<f_1,\ldots f_n>$ be monotonic then f is OWA lossless on $\mathcal{S}$ iff it is lossless for any $Q\in\mathcal{S}$ such that $||Q||\leq k$. In other words, the losslessness of mapping is the k-local property for k-distributive schemes.

Proof

It suffices to prove that

$$(3)\quad \cap_{i=1}^{n} f_i^{-1}(rep(f_i(Q)))\cap\mathcal{S}=rep(Q)\cap\mathcal{S}$$

for all $Q\in\mathcal{S}$ with $||Q||\leq k$ implies

$$\cap_{i=1}^{n} f_i^{-1}(rep(f_i(S)))\cap\mathcal{S}=rep(S)\cap\mathcal{S}\quad\text{for all } S\in\mathcal{S}$$

First of all let us notice that for any $S\in\mathcal{S}$

$$S\in\cap_{i=1}^{n} f_i(rep(f_i(S)))\cap\mathcal{S}$$

therefore

$$rep(S)\cap\mathcal{S}\subset\cap_{i=1}^{n} f_i^{-1}(rep(f_i(S)))\cap\mathcal{S}$$

On the other hand

$$\cup_{Q\subset S} f_i(Q)\subseteq f_i(S)$$

(by monotonicity of f_i) and

$$(4)\quad \cap_{i=1}^{n} f_i^{-1}(rep(f_i(S)))\cap\mathcal{S}$$
$$\subset \cap_{i=1}^{n} f_i^{-1}(rep(\cup_{Q\subset S} f_i(Q)))\cap\mathcal{S}$$

since

$$rep(\cup_{Q\subset S} f_i(Q))\supset rep(f_i(S))$$
$$rep(\cup_{Q\subset S} f_i(Q))=\cap_{Q\subset S} rep(f_i(Q))$$

and finally (4) can be transformed into the form

$$(5)\quad \cap_{Q\subset S}\cap_{i=1}^{n} f_i^{-1}(rep(f_i(Q)))\cap\mathcal{S}$$

but from (1)

$$\cap_{i=1}^{n} f_i^{-1}(rep(f_i(Q)))\cap =rep(Q)\cap\mathcal{S}$$

therefore, (5) can be transformed further into

$\cap_{Q\subset S} rep(Q) \cap = rep(S) \cap \mathcal{S}$

$S = \cup_{Q\subset S} Q$ by the decomposability of $\mathcal{S}$

Dependencies and losslessness of transformations

Lemma: For any set Σ of dependencies with finite chase there is a k such that Sat(Σ) is k-distributive

Proof:

Let Minimal(Σ) = {$chase_\Sigma(T^i)$ i=1 ...n}, where $T^{(i)}$ are minimal multitables for our database schema. We have then the following decomposition property

$\forall_{S\in Sat(\Sigma)} S = \cup \{v(U): U\in Minimal(\Sigma) \wedge v(U)\subseteq S\}$

The set Minimal (Σ) "parametrizes" the set of all minimal multirelations in Sat(Σ) i.e. multirelations of the form v(W) where W∈Minimal(Σ) and v is a certain valuation of variables in W. We are ready now to present our algorihm for testing losslessness.

ALGORITHM

Let $f=\langle f_1,...,f_n\rangle$ be the sequence of PCR transformations

Let Σ be the set of dependencies with a finite chase. In order to test whether $f=\langle f_1,...,f_n\rangle$ is lossless do the following:

(1) Generate the set of multitables Minimal(Σ)

(2) For each of the multitables W ∈ Minimal(Σ), compute $U=f^{-1}(f(W))$ treating variables in W as pairwise different constants. (3) Test whether $chase_\Sigma(U)$ is equivalent (in the sense of table or tableaux equivalence) to multitable $chase_\Sigma(W)$. If it is the case for every W∈Minimal(Σ), then $f=\langle f_1,...,f_n\rangle$ is lossless;if nct, then f is lossy.

Restoring the original database state

An important fact is that, given a losless transformation, our technique enables us to restore the original state of the database from the new state of the database. It turns out that our algorithm explicitely constructs the "inversion" .

Given a state S of the new scheme, the corresponding state of the original database scheme is computed as

$Q = Constant\ (chase_\Sigma f^{-1}(rep(S)))$

where Constant (T) is the set of all tuples of multitable T without variables (i.e., only constants occurring in them).

Sketch of the Proof of Correctness of the Algorithm

We will sketch here the proof of correctness, which will be presented in the full version of the paper. By Theorem 1 we can restrict our attention to test (2') only for multirelations of the form v(W) for some valuation v and some minimal multitable from the set Minimal (Σ). Therefore, we have to prove that

$f^{-1}(rep(f(v(W))))\cap Sat(\Sigma) = rep(v(W))\cap Sat(\Sigma)$

for every W ∈ Minimal(Σ). As we know $f^{-1}(rep(f(v(W)))) = rep(U)$ for some multitable U. Finally,

$rep(U)\cap Sat(\Sigma) = rep(v(W))\cap Sat(\Sigma)$

iff $rep(chase_\Sigma(U)) = rep(chase_\Sigma v(W))$.

What remains to be shown here is how to go from particular valuation v to any valuation v. This is simple, however, and corresponds actually to what the algorithm 1 is doing. We treat the variables occurring in W as parameters - generalized constants, compute f^{-1} and apply chase treating those paramters as if they were pairwise different constants. The proof of the correctness of this step is somewhat laborous and will be presented in the full version of the paper.

4. Applications of the algorithm: Multirelational database schemes specified by inclusion and functional dependencies

4.1 Inclusion dependencies

Inclusion dependencies (IDs) play the key role in our considerations about lossless transformations of multirealtional schemes. This is so simply because without inclusion dependencies only trivial transformations are lossless.

We will characterize here the families of IDs having finite chase property and, in consequence, k-distributivity property. This class contains Sciore's noncircular IDs and, referring to his argumentation, is a "reasonable" one.

Let $r[X]\subseteq s[Y]$ be an inclusion dependency where both X and Y are collections of column numbers. We will say that two relations R and S satisfy inclusion dependency $r[X]\subseteq s[Y]$ iff

$\pi_{\vec{X}}(R) \subseteq \pi_{\vec{Y}}(S)$

where, $\vec{X}$ and $\vec{Y}$ are orderings of the elements of X and Y in the increasing order. For example,$r[1,3,6]\subseteq s[2,9,7]$ will be satisfied by R and S iff

$\pi_{1,3,6}(R) \subseteq \pi_{2,7,9}(S)$

Notice that the order of columns is now important in making the projection.

Notice also that any inclusion dependency ID determines a one to one correspondence $\vec{ID}$ between elements of X and Y.That is ,$\vec{ID}(i)=j$ iff the ordinal of j in Y is the same as ordinal of i in X. For any subset X' of X $\vec{ID}(X')$ = {ID(i):i∈X} Following the idea from [SC] with each family D of inclusion dependencies we may associate a graph G_D . The nodes of this graph will correspond to the relational schemes and the edges (there could be many between two nodes) to the different inclusion dependencies. Each edge is labelled by the associated inclusion dependency.We will be interested here in those families of inclusion dependencies which have finite chase property.This will enable us to handle the problem

of testing loselessnsess of transformations efficiently. What is then the class of families of IDs having the finite chase property? Before giving the precise answer we will introduce the special class of families o inclusion dependencies.

Definition

A cycle $r \xrightarrow{ID_0} s_1 \xrightarrow{ID_1} s_2 \xrightarrow{ID_2} \ldots s_n \xrightarrow{ID_n} r$

where

$ID_0 = r[X_1] \subset s_1[Y_1]$

$ID_1 = s_1[X_2] \subset s_2[Y_2]$

.

.

.

$ID_n = s_n[X_{n+1}] \subset r[Y_{n+1}]$

is called a proper cycle iff

(i) $Y_{n+1} \subset X_1$

(ii) $\widetilde{ID}_0(Y_{n+1}) \subset X_2$,

$\widetilde{ID}_1(\widetilde{ID}_0(Y_{n+1})) \subset X_3$

.

.

.

$\widetilde{ID}_{n-1}(\widetilde{ID}_{n-2} \ldots \widetilde{ID}_0(Y_{n+1})) \subset X_{n+1}$

The family which is either acyclic or contains only proper cycles is called proper circular.

The proper circular families include noncircular families of Sciore [SC]]. They include circular families such as:

$\{r[1] \subseteq s[1]\ ; s[1] \subseteq r[1]\ \}$ or

$\{r[1,2] \subseteq s[1,2];$

$s[1,3] \subseteq w[1,3]$

$w[1] \subseteq r[1]\ \}$

Further in addition to Sciore's noncircular families of ID we will also allow proper circular families of the type described above.The importance of this families is showed in the following theorem.

Theorem 2

The family of inclusion dependencies has a finite chase iff it is proper circular.

Proof

Let us start the chase procedure by assuming that R contains one parameterized tuple $x=<x_1,\ldots x_n>$ where $x_1,\ldots x_n$ are variables. Inclusion dependencies associated with different edges of the cycle

$r \rightarrow s_1, s_2 \rightarrow s_3, \ldots s_n \rightarrow r$

will successvely introduce (via chase rules) new parameterized tuples to the relations $s_1,\ldots s_n$.If the resulting table after applying rules associated with edge $s_{n-1} \rightarrow s_n$ satisfies all inclusion dependencies then the chase will terminate, if not we will have to use inclusion dependencies associated with the edge $s_n \rightarrow r$ introducing a new tuple into r, and in consequence entering the infinite loop. Therefore we have to examine the conditions under which we do not enter the infinite loop which are also, the conditions for finite chase.

It is obvious that if the family of inclusion dependencies is not circular, then it must have a finite chase. Assume then that the family contains a proper cycle.

Then, however, we will not have to apply the rule associated with inclusion dependency $s_n[X_{n+1}] \subseteq r[Y_{n+1}]$ since after application of rules associated with $r \rightarrow s_1, \ldots s_{n-1} \rightarrow s_n$ the resulting multitable will satisfy all inclusion dependencies

$ID_0 \ldots ID_n\ ;\ (\ s_n[X_{n+1}] \subseteq r[Y_{n+1}])$

will be satisfied since $Y_{n+1} \subseteq X_1$ Assume now that our conditions are not met.Then there exists a non proper cycle

$r \rightarrow s_1 \rightarrow s_2 \ldots s_n \rightarrow r$

That is either

$Y_{n+1} \not\subseteq X_1$ or $Y_{n+1} \subseteq X_1$ but

for some k such that k=2,...,n-1

$\widetilde{ID}_k(\widetilde{ID}_{k-1} \ldots \widetilde{ID}_0(Y_{n+1})) \not\subseteq X_{k+2}$

If $Y_{n+1} \not\subseteq X_1$ than obviously after application of the rule of chase associated with ID_{n-1} the resulting multitable will not satisfy ID_n , so we have to apply the rule associated with ID_n introducing a new tuple into R and entering in that way the infinite loop.

If $Y_{n+1} \subseteq X_1$ but one of the other conditions is not satisfied then after application of the chase rule corresponding to ID_k we will have the situation

$s_{k+1}[\widetilde{ID}_k(\widetilde{ID}_{k-1}(\ldots \widetilde{ID}_0(Y_{n+1}))] \not\in r[Y_{n+1}]$

Since chase (by inclusion dependencies) always introduces new symbols,we will also have

$s_n[X_{n+1}] \not\in r[Y_{n+1}]$

and again it will be necessary to apply rule ID_n entering the infinte loop again. By $N(r_i,r_j)$ let us denote the number of different pathes from r_i to r_j in the graph G_D for the family of inclusion dependencies D and

let $k_D=<k_1 \ldots k_n>$

where n is then number of relational schemes in our database scheme,and

$k_j = Max\{N(r_i,r_j) : i=1..n\}$

By the neighbor of a node r_i we will mean any node r_j such that there is arc between r_i and r_j in the graph G_D.

Theorem 3

The family D of inclusion dependencies is decomposable iff it is proper circular.

Sketch of A Proof

Since a proper circular family has a finite chase, the set Minimal (D) is defined and each S Sat(D) can be represented as

$S = \cup\{v(W) : W \in Minimal(D) \text{ and } v(W) \subseteq S\}$

The cardinality of

$$W^i = chase_{\Sigma}(T^i) \in Minimal(D)$$

is equal to $k=(k_1^i,\ldots,k_n^i)$

where $k_j^i = N(r_i,r_j)$, therefore, the cardinality of any of v(W) in the decomposition of S is smaller than

$$k_D = <k_1,\ldots,k_n>$$

where $k_j = Max\{ N(r_i,r_j): i=1..n\}$. The proper circular families of inclusion dependencies have, therefore, the properties required by our algorithm.

Collorary 1

We can test the losslessness of the transformations $f=<f_1,\ldots f_n>$ of the database scheme specified by the family of proper circular inclusion dependencies using Algorithm 1. Since each family of functional dependencies is 1-distributive (i.e. each relation satisfying some family of functional dependencies can be viewed as the union of one tuple relations which trivially satisfy , we can also apply our algorithms to database scheme, specified by the class of proper circular inclusion dependencies and functional dependencies.

Collorary 2

We can test, using Algorithm 1, the losslessness of the transformation $f=<f_1,\ldots f_n>$ of any database scheme, specified by the family of functional and proper circular inclusion dependencies. We may also test schemes specified by arbitrary implicational and proper circular inclusion dependencies. However, in an arbitrary case the index ("k") of locality may be quite large depending on the degree of interaction between arbitrary implicational dependencies. We do not pursue this matter further since the general case does not seem to really occur in practice. Usually, each relation scheme of the database scheme is enforced to satisfy at most one total tuple generating dependency (join dependency). This situation could be handled,for example, by projecting the relation according to the join dependency, adding some inclusion dependencies and treating the resulting scheme as the original one with the join dependency replaced by the inclusion ones. There is an interesting subclass of inclusion dependencies which enables us to deal with the whole class of implicational dependencies still maintaining even 1-decomposability (i.e. k=1) and, in consequence, 1-locality of the equivalence problem.

Definition

By a tree structured family of IDs we mean any proper circular family such that for any two relational schemes r and s the number of different pathes in the graph G_D from r to s, (N(r,s))=1. As the immediate conclusion from Theorem 3 we obtain : Corollary 3 For any tree structured family D of IDs Sat(D) is 1-decomposable, that is, each S Sat(D) can be represented as

$S=\cup S'$ such that $S' \not< <1,\ldots 1>$ and $S' \in Sat(D)$.

Finally, since any family of implicational dependencies is 1-distributive we obtain.

Collorary 4

For any family Σ consisting of the set of arbitrary implicational dependencies and tree structured class of inclusion dependencies, the set $Sat(\Sigma)$ is 1-decomposable. In consequence, we can apply our algorithm to the database schemes specified by Σ . In this case the algorithm will be especially simple, since in all multitables from the set Minimal (Σ) each table contains at most one tuple. Tree structured families of inclusion dependencies correspond directly to tree structured ISA hierarchies and therefore are practically important subfamilies of IDs.

CONCLUSIONS

In this paper, we have presented a formal definition of a lossless transformations of a database under the closed and the open world assumptions, using the notions of k-locality and k-distributivity. We proposed an algorithm for testing whether a transformation,which is a vector of arbitrary relational expressions built up from projection, cartesian product and restriction, is lossless. The algorithm can be applied to database schemes specified by various types of dependencies including implicational and inclusion dependencies, and not necessarily satisfying the Universal Instance Assumption. Although we cannot deal with arbitrary families of ID's we can constructively deal with two important subfamilies: the so called proper circular IDs, which are slightly more general than Sciore's noncircular IDs, and tree structured IDs. Proper circularfamilies of inclusion dependencies are those which have finite chase inclusion dependencies; they play the crucial role in ourconsiderations. In the case where a given transformation turns out to be lossless our algorithm explicitly constructs the inversion mapping restoring the state of the original database scheme from any given state of the new transformed database scheme. There are some other related problems which, although not discussed here, are important from the practical point of view. They are:

a) Given a relational transformation f of a database scheme, what other information (given in the form of transformation g) should be added in order to make *transformation* <f,g> lossless? Transformation g could be treated as the complement of f (see [BS1]). This technique could be directly used for testing whether the transformation g is a complement of f (we must test whether <f,g> is lossless). The other, more difficult problem is the generation of complements of the transformation f. This could also be done by a similar

technique, and will be treated elsewhere.

b) For the two relational transformations f and g, is the independence of f and g in the sense of [BS2] a k-local property under the open world assumption. Between all invariant transformations of a given database scheme, those whose "components" are independent are of real practical interest; therefore, this problem deserves attention. We believe that our technique is applicable in this case and similar algorithms for testing independence could be created.

Acknowledgement:The authors are grateful to the Program Committee for pointing out the mistake in the earlier version of the paper.

REFERENCES

[ABU] Aho,A.V.,Beeri,C.,Ullman,J.D., The theory of joins in relational databases, ACM Trans. Database Syst. 4 (1979) 297-314.

[BMSU] Beeri,C., Mendelson,A.O., Sagiv,Y., Ullman, J.D., Equivalence of relational database schemes.SIAM J.Comput.10 (1981) 352-370.

[BS1] Banchillhon, F., Spyratos, N., Update semantics of relational views, ACM TODS, Dec.1981,557-575.

[BS2] Bancillhon, F., Spyratos, N., Independent components of databases, Proc. 7th International Conference on VLDB, Cannes, France, Sept.1981.

[BV] Berri, C., Vardi, M., A proof procedure for data dependencies,The Hebrew University, Dept. of Computer Science, Dec. 1980.

[CFP] Cassanova, M., Fagin, R., Papadimitriou, C., Inclusion dependencies and their interactions with functional dependencies, Proc. ACM PODS, 1982, pp. 171-176.

[F] Fagin,R., Horn clauses and database dependencies, Proc.1980 ACM SIGACT Symp. on Theory of Computing, pp. 123-134.

[IL1] Imielinski, T., Lipski, W., A technique for translating states between database schemata, Proc. 1982 ACM SIGMOD Conference, Orlando,Florida.

[IL2] Imielinski, T. Lipski, W., Inverting relational expressions - a natural and uniform technique for various database problems, Proc.1983 ACM SIGACT-SIGMOD Symp. on Principles of Database Systems (PODS), Atlanta, pp. 305-312.

[IL3] Imielinski,T.,Lipski,W.,A note of equivalence of relational expressions, Unpublished manuscript, Warsaw, 1982.

[R] Reiter, R., On closed world databases. In Logic and Databases,(M. Gallaire and J. Minker, Eds.) Plenum Press, New York, 1978, pp. 58-76.

[SC] Sciore,E.,ACM PODS Conference, Atlanta, 1983.

[U] Ullman,J.D., Principles of database systems, Computer Science Press, Potomac, MD, 1982 (Second Edition).