Null Inclusion Dependencies in Relational Databases

Mark Levene
Department of Computer Science
University College London
Gower Street
London WC1E 6BT, U.K.
Email: mlevene@cs.ucl.ac.uk

George Loizou
Department of Computer Science
Birkbeck College
Malet Street
London WC1E 7HX, U.K.
Email: george@dcs.bbk.ac.uk

Abstract

Functional dependencies (FDs) and inclusion dependencies (INDs) are the most fundamental integrity constraints that arise in practice in relational databases. We introduce null inclusion dependencies (NINDs) to cater for the situation when a database is incomplete and contains null values. We show that the implication problem for NINDs is the same as that for INDs. We then present a sound and complete axiom system for null functional dependencies (NFDs) and NINDs, and prove that the implication problem for NFDs and NINDs is decidable and EXPTIME-complete. By contrast, when no nulls are allowed, this implication problem is undecidable. This undecidability result has motivated several researchers to restrict their attention to FDs and noncircular INDs in which case the implication problem was shown to be EXPTIME-complete. Our results imply that when considering nulls in relational database design we need not assume that NINDs are noncircular.

1 Introduction

Functional dependencies (FDs) [ULLM88, ATZE93] generalize the notions of *entity integrity* and *keys* [CODD79] and inclusion dependencies (INDs) [MITC83, CASA84] generalize the notions of *referential integrity* and *foreign keys* [CODD79, DATE86]. In this sense FDs and INDs are the most fundamental data dependencies that arise in practice.

Codd [CODD79] suggested the addition to the database domains of an unmarked null value, whose meaning is "value at present unknown", which we denote by *unk*, in order to extend relations so that they can model situations when the information is incomplete. We call such relations, whose tuples may contain the null value *unk*, *incomplete relations* and we call a sequence of incomplete relations an *incomplete database* (or simply a database). Following Codd's proposal, incomplete information is represented in SQL by using *unk* as a distinguished null value [DATE93].

The theory of data dependencies in relational databases has been generalized in order to deal with incomplete databases. In particular, FDs have been generalized to *null functional dependencies* [LIEN82, ATZE86, LEVE96], multivalued dependencies [ULLM88, ATZE93] have been generalized to *null multivalued dependencies* [LIEN82] and join dependencies [ULLM88, ATZE93] have been generalized to *null join dependencies* [LEVE92].

To our knowledge INDs have not yet been generalized to *null inclusion dependencies*

(NINDs) in order to cater for the situation when a database is incomplete; herein we investigate such a generalization.

To illustrate the need for NINDs consider a database containing three incomplete relations, r_1, r_2 and r_3, shown in Tables 1, 2 and 3, respectively, which are over relation schemas R_1, R_2 and R_3, respectively (relation schemas are sequences of attributes).

STUD	DEPT
Iris	Computing
Susi	Computing
Reuven	*unk*
Eli	*unk*
Naomi	Maths

Table 1: The incomplete relation r_1

HEAD	DEPT
Dan	Computing
Annette	Philosophy
unk	History
Brian	*unk*

Table 2: The incomplete relation r_2

LECT	DEPT
Dan	Computing
Hanna	Maths
unk	Maths
Raphael	*unk*
Annette	Philosophy
Brian	History

Table 3: The incomplete relation r_3

The semantics of this incomplete database are: in r_1 each STUDent studies in one DEParTment, in r_2 each DEPT has one HEAD and in r_3 each LECTurer teaches in one DEPT. Furthermore, each DEPT of r_1 is included in the set of DEPTs of r_3 and each HEAD and DEPT of r_2 is included in the set of LECTs and DEPTs of r_3.

Thus, the semantics of the database schema, $\{R_1, R_2, R_3\}$, can be captured by a set of NFDs, F, and a set of NINDs I, where F = $\{R_1$: STUD $\rightarrow$ DEPT, R_2 : DEPT $\rightarrow$ HEAD, R_3 : LECT $\rightarrow$ DEPT$\}$ and I = $\{R_1$[DEPT] $\subseteq$ R_3[DEPT], R_2[HEAD, DEPT] $\subseteq$ R_3[LECT, DEPT]$\}$.

Intuitively, a NFD R : X $\rightarrow$ Y is satisfied in an incomplete relation r over R, if there exists a relation without nulls emanating from r which satisfies the NFD on using the classical definition of FD satisfaction [ULLM88, ATZE93].

Correspondingly, we now explain the meaning of a NIND R[X] $\subseteq$ S[Y] being satisfied in an incomplete database. Intuitively, an occurrence of *unk* is *less informative* than an occurrence of a nonnull value, and a tuple t_1 over a relation schema $R_1 = < A_1, \ldots, A_m >$ is less informative than a tuple t_2 over a relation schema $R_2 = < B_1, \ldots, B_m >$, written $t_1 \sqsubseteq t_2$, if $\forall$ i $\in$ {1,...,m}, $t_1[A_i]$ is less or equally informative than $t_2[B_i]$.

The NIND R[X] $\subseteq$ S[Y] is satisfied in an incomplete database, d, containing relations r over R and s over S, if $\forall t_1 \in$ r, $\exists t_2 \in$ s such that t_1[X] $\sqsubseteq$ t_2[Y]. This definition of a NIND can be seen to faithfully generalize the definition of an IND, since it implies that there exists a database without nulls emanating from d which satisfies the NIND on using the classical definition of IND satisfaction [MITC83, CASA84].

Consider next the example: let BOSS be a relation schema having two attributes, EMP and MGR, and let r be the relation over BOSS shown in Table 4. The fact that a manager is also an employee is captured by the NIND BOSS[MGR] $\subseteq$ BOSS[EMP]. This type of NIND gives rise to the notion of *circular* NINDs [SCIO86], which leads to the following problem. In order to enforce the satisfaction of this NIND over BOSS in a relation without nulls we get into a circular argument implying that a relation that satisfies the NIND has an infinite number of tuples (we allow only finite relations), unless we allow employees to manage themselves. In the presence of incomplete information, as can be verified in Table 4, this semantic problem does not arise.

EMP	MGR
Jack	John
John	Jill
Jill	*unk*

Table 4: The incomplete relation r

A more serious problem that arises with INDs is that the implication problem for FDs together with INDs holding in a database without nulls was shown to be undecidable [MITC83, CHAN85]. This has motivated several researchers [MANN86] to restrict their attention to noncircular INDs, since the implication problem for FDs and noncircular INDs is decidable and was shown to be EXPTIME-complete [COSM85].

We show that this restriction can be relaxed in the case of incomplete relations, since the implication problem for NFDs together with NINDs holding in incomplete databases is decidable. In fact, we prove that the implication problem for NFDs and NINDs is EXPTIME-complete. Due to the fact that in practice it is very difficult to avoid null values (as was already mentioned SQL supports the null value *unk*), our result implies that database designers need not restrict their attention to any subclass of NFDs and NINDs.

We derive a sound and complete axiom system for NFDs and NINDs based on Mitchell's sound and complete axiom system for FDs together with INDs [MITC83]. Our axiom system consists of Lien's and Atzeni's axiom system for inferring NFDs from NFDs [LIEN82, ATZE86], Casanova et al.'s axiom system [CASA84] for inferring NINDs from NINDs, the *pullback* inference rule [MITC83] for inferring NFDs from NFDs and NINDs and the *null collection* inference rule, which is a new inference rule, motivated by the *collection* inference rule [MITC83], for inferring NINDs from NFDs and NINDs.

In our motivating example we observe that the NFD R_2 : HEAD $\rightarrow$ DEPT is implied by the NFD R_3 : LECT $\rightarrow$ DEPT and the NIND R_2[HEAD, DEPT] $\subseteq$ R_3[LECT, DEPT], on using the pullback inference rule.

A desirable goal in relational database design is for a database schema to be in *Boyce-Codd normal form* (BCNF) [ULLM88, ATZE93], which intuitively means that the left-hand sides of all nontrivial FDs (in our case NFDs) are supersets of keys. In order to incorporate NINDs into database design we define the notion of *key-based* NINDs [MANN86, MANN88], which generalize the notion of foreign key as follows.

A NIND R[X] $\subseteq$ S[Y] is said to be key-based if Y is a key for S. In this case we consider X to be a *foreign key* of R with respect to S (cf. [DATE86]). We show that if a database schema is in BCNF with respect to a set of NFDs, F, and all the NINDs in a set of NINDs, I, are key-based, then we have the desirable property that any NFD that is logically implied by F $\cup$ I is logically implied by F on its own and any NIND that is logically implied by F $\cup$ I is logically implied by I on its own (cf. [MANN88]). In this case we say that F and I have *no interaction*. As a consequence, when F and I have no interaction, the implication problem for NFDs and NINDs reduces to the separate implication problems for NFDs and NINDs. This result suggests that a good design principle in the presence of NFDs and NINDs is a BCNF database schema together with a set of key-based NINDs. Our result generalizes that of [MANN86], since we do *not* assume that the set of NINDs is noncircular.

The rest of the paper is organized as follows. In Section 2 we formalize incomplete relations and incomplete databases, and define a partial order on tuples of such relations. In Section 3 we define the notions of NFD and NIND and their satisfaction in the context of incomplete databases. In Section 4 we extend the *chase procedure*, defined over relations without nulls with respect to FDs [MAIE79, ATZE93] and INDs [JOHN84, MANN88], to incomplete relations with respect to NFDs and NINDs. In Section 5 we utilize the chase procedure in order to derive a sound and complete axiom system for NFDs together with NINDs. In Section 6 we study the complexity of the implication problem for NFDs together with NINDs. In Section 7 we investigate the special case of *key-based* NINDs, which formalize the notion of referential integrity, and consider the problem of interaction between NFDs and such NINDs. Finally, in Section 8 we give our concluding remarks and in Appendix A we present proofs of some complexity results.

2 Relations that Model Incomplete Information

In this section we introduce the underlying database model used throughout the paper.

Definition 2.1 (Relation and database schema) Let Att be a countably infinite set of attribute names (or simply attributes). A *relation schema* (or simply a schema) is a relation symbol R together with an associated similarity type, denoted by type(R), such that type(R) $\in \omega$, where ω is the set of all natural numbers. We refer to such a schema simply as R. For convenience we associate with each relation schema R a total and one-to-one mapping, att: {1,...,type(R)} $\rightarrow$ Att, which allows us to name the type(R) components of a relation schema R; we denote the sequence <att(1), ..., att(type(R))> by schema(R).

A *database schema* is a set $\mathbf{R} = \{R_1, \ldots, R_n\}$, n $\in \omega$, such that each $R_i \in \mathbf{R}$ is a relation schema.

From now on we will refer to a sequence of attributes as a shorthand for a sequence of distinct attributes (we assume that sequences of attributes do not contain any repeated attributes). As usual uppercase letters (which may be subscripted) appearing at the end of the alphabet such as X, Y, W, Z will be used to denote sequences of attributes, while those at the beginning of the alphabet such as A, B, C, D will be used to denote single attributes. When convenient we will write $A_1, \ldots, A_n$ instead of $< A_1, \ldots, A_n >$ to describe a sequence and when no ambiguity arises we will refer to a sequence of attributes as a set of attributes.

We take A $\in < A_1, ..., A_n >$ to mean A $\in \{A_1, ..., A_n\}$ and $< A_1, ..., A_n > \subseteq < B_1, ..., B_m >$, with n $\leq$ m, to mean $\{A_1, ..., A_n\} \subseteq \{B_1, ..., B_m\}$.

We will denote the fact that two sequences have the same elements, i. e. that both X $\subseteq$ Y and Y $\subseteq$ X hold, by X $\approx$ Y, and will denote the concatenation of two sequences X and Y by XY; we will assume that, unless otherwise stated, when we concatenate two sequences of attributes these sequences have no common attributes, i. e. they are disjoint. The difference between two sequences of attributes, denoted by X$-$Y, is the sequence resulting from removing all the common attributes in X and Y from X while maintaining the original order of the attributes remaining in X. If the sequences X, Y are *not* disjoint we define their concatenation XY to be (X$-$Y)Y. The intersection of two sequences of attributes, denoted by X $\cap$ Y, is a shorthand for X $-$ (X $-$ Y). For simplicity we will not distinguish between the empty sequence, <>, and the empty set, $\emptyset$.

For a sequence or set X, |X| will denote the cardinality of X. Finally, the *size* of a set or sequence X is defined to be the cardinality of a standard encoding [GARE79] of X.

Definition 2.2 (Relation and database) Let *Dom* be a countably infinite set of constants and let $unk, inc \in Dom$ be two distinguished constants, denoting the null values "unknown" and "inconsistent", respectively.

A *type(R)-tuple* (or simply a tuple whenever type(R) is understood from context) is a member of $Dom \times \ldots \times Dom$ (type(R) times).

An *incomplete relation* (or simply a relation) over R is a finite (possibly empty) set of type(R)-tuples. An *incomplete database* (or simply a database) over $\mathbf{R} = \{R_1, \ldots, R_n\}$ is a sequence d $= < r_1, \ldots, r_n >$ such that each $r_i \in$ d is a relation over $R_i \in \mathbf{R}$.

We call constants in $Dom - \{unk, inc\}$ nonnull values and the constants *unk* and *inc* (unmarked) null values. We let DB($\mathbf{R}$) denote the countably infinite set of all databases over $\mathbf{R}$. From now on we will assume that d $= < r_1, \ldots, r_n >$ is a database over a database schema $\mathbf{R} = \{R_1, \ldots, R_n\}$ and that r is a relation over a relation schema R.

Definition 2.3 (Projection) The *projection* of a tuple t $\in$ r onto the attribute A_i = att(i), denoted by t$[A_i]$, is the i-coordinate of t, i. e. t(i). We further extend the notion of projection to a sequence of attributes, Y $= < A_1, \ldots, A_n > \subseteq$ schema(R), as follows: the projection of t onto Y (also called the Y-value of t), denoted by t[Y], is defined by t[Y] = <t(A_1), ..., t(A_n)>. The projection of r onto Y, denoted by π_Y(r), is defined by π_Y(r) = {t[Y] | t $\in$ r}.

Definition 2.4 (Active domain) The *active domain* of a relation r over R, denoted by adom(r), is defined by

$$adom(r) = \{v \mid \exists A \in schema(R), \exists t \in r \text{ such that } t[A] = v\} \cup \{unk, inc\}.$$

The active domain of a database d $= < r_1, \ldots, r_n >$ over $\mathbf{R}$, denoted by adom(d), is defined by

$$adom(d) = \cup\{adom(r_i) \mid r_i \in d\}.$$

Definition 2.5 (Complete and consistent relations and databases) A type(R)-tuple t is said to be *complete* if $\forall A_i \in$ schema(R), t$[A_i] \neq unk$ and t$[A_i] \neq inc$, i. e. t$[A_i]$ is a nonnull value, otherwise t is said to be *incomplete*; t is said to be *inconsistent* if $\exists A_i \in$ schema(R), such that t$[A_i] = inc$, otherwise t is said to be *consistent*.

A relation r over R is said to be *complete* if $\forall t \in r$, t is complete, otherwise r is said to be *incomplete*. A relation r over R is said to be *inconsistent* if $\exists t \in r$ such that t is inconsistent, otherwise r is said to be *consistent*.

A database d over R is said to be *complete* if all the relations r in d are complete, otherwise d is said to be *incomplete*. A database d over R is said to be *inconsistent* if there exists a relation r in d such that r is inconsistent, otherwise d is said to be *consistent*.

We let COMPLETE($\mathbf{R}$) denote the countably infinite set of all complete databases over $\mathbf{R}$.

Definition 2.6 (Less informative constants) Let r be a relation over R. We define a partial order in *Dom*, denoted by $\sqsubseteq$, as follows:

$$v \sqsubseteq u \text{ if and only if } u = v \text{ or } v = unk \text{ or } u = inc, \text{ where } v, u \in Dom.$$

Definition 2.7 (Less informative tuples) Let schema(R_1) = $< A_1, \ldots, A_m >$ and schema(R_2) = $< B_1, \ldots, B_m >$, i. e. type(R_1) = type(R_2) = m. We extend $\sqsubseteq$ to be a partial order in the set of type(R_1)-tuples and type(R_2)-tuples, as follows: where t_1 is a type(R_1)-tuple and t_2 is a type(R_2)-tuple, t_1 is *less informative than* t_2 (or equivalently t_2 is *more informative than* t_1), written $t_1 \sqsubseteq t_2$, if

$$\forall i \in \{1, \ldots, m\}, t_1[A_i] \sqsubseteq t_2[B_i].$$

We say that t_1 and t_2 are *information-wise equivalent*, written $t_1 \cong t_2$, if $t_1 \sqsubseteq t_2$ and $t_2 \sqsubseteq t_1$.

We note that for two type(R)-tuples, t_1 and t_2, $t_1 \cong t_2$ if and only if $t_1 = t_2$. We further observe that the set of all type(R)-tuples is a *complete lattice* having no infinite chains [DAVE90], whose bottom element is <*unk*, ..., *unk*> and whose top element is <*inc*, ..., *inc*>. We next define the *join* operator [DAVE90] of this complete lattice of tuples.

Definition 2.8 (The join operator) Let r be a relation over R. We define the *join* operator, denoted by $\sqcup$, as a mapping from an ordered pair, (v_1, v_2), of constants in $Dom \times Dom$ to a single value in Dom as follows:

$v_1 \sqcup v_2$ is the least upper bound of v_1 and v_2 with respect to $\sqsubseteq$.

We extend $\sqcup$ to be a mapping from an ordered pair, (t_1, t_2), of type(R)-tuples to a single type(R)-tuple as follows:

$t_1 \sqcup t_2$ = t, where t is a type(R)-tuple and $\forall A_i \in$ schema(R), t$[A_i] = t_1[A_i] \sqcup t_2[A_i]$.

It can easily be verified that $t_1 \sqcup t_2$ returns the least upper bound of t_1 and t_2, namely the join operator realises the lattice theoretic join.

Definition 2.9 (The set of possible worlds of a relation) The set of all *possible worlds* relative to a relation r over R, denoted by POSS(r), is defined by

$$POSS(r) = \{s \mid s \text{ is a relation over R and there exists a total and onto mapping } f : r \rightarrow s \text{ such that } \forall t \in r, t \sqsubseteq f(t) \text{ and } f(t) \text{ is complete}\}.$$

We extend POSS to a database d = $< r_1, \ldots, r_n >$ over **R**, as follows:

$$POSS(d) = \{< s_1, \ldots, s_n > \mid s_1 \in POSS(r_1) \text{ and } \ldots \text{ and } s_n \in POSS(r_n)\}.$$

The following proposition states the relationship between the concept of possible worlds and consistency.

Proposition 2.1 *The following two statements are true:*

1. *A relation r over R is inconsistent if and only if POSS(r)* $= \emptyset$.
2. *A database d over* **R** *is inconsistent if and only if POSS(d)* $= \emptyset$. □

From now on we assume that unless otherwise stated relations and databases are consistent.

3 Null Functional and Null Inclusion Dependencies

In this section we introduce the null data dependencies we investigate in the paper.

Definition 3.1 (Null Functional Dependency) A *null functional dependency* (or simply a NFD) over a database schema **R** is a statement of the form R : X $\rightarrow$ Y (or simply X $\rightarrow$ Y whenever R is understood from context), where R $\in$ **R** and X, Y $\subseteq$ schema(R). The NFD X $\rightarrow$ Y is said to be *trivial* if Y $\subseteq$ X; it is said to be *standard* if X $\neq \emptyset$.

Hereafter we will assume that all NFDs are standard. We also assume that a database schema **R** has a set of NFDs associated with it, denoted by F. We let NFD(**R**) be the set of all possible sets F of NFDs over **R**.

The following definition is a syntactic definition of NFD satisfaction.

Definition 3.2 (Satisfaction of a NFD) A NFD R : X $\rightarrow$ Y over **R** is satisfied in a database d, over **R**, denoted by d $\models$ R : X $\rightarrow$ Y, if, where $r \in d$ is the relation over R $\in$ **R**, r is consistent and $\forall t_1, t_2 \in$ r, if t_1[X] and t_2[X] are complete and t_1[X] = t_2[X], then $t_1[Y] \sqcup t_2[Y]$ is consistent.

From now on we will assume that when R : X $\rightarrow$ Y $\in$ F and d is a database over **R**, then r $\in$ d is the relation over R $\in$ **R**.

We observe that if r $\in$ d is a complete relation, then d $\models$ X $\rightarrow$ Y if and only if the classical definition of functional dependency is satisfied, i. e. $\forall t_1, t_2 \in$ r, if t_1[X] = t_2[X], then t_1[Y] = t_2[Y] [ULLM88, ATZE93].

The following proposition gives a semantic characterization of NFD satisfaction in terms of possible worlds.

Proposition 3.1 *Let d be a database over* **R** *and $r_1 \in d$ be the relation over $R_1 \in$* **R** *such that r_1 is consistent. Then $d \models R_1 : X \rightarrow Y$ if and only if $\exists r_2 \in$ POSS(r_1) such that $\forall t_1, t_2 \in r_2$, if t_1[X] = t_2[X], then t_1[Y] = t_2[Y].* □

As defined above a NFD is *nonstandard* if its left-hand side is the empty set, i. e. if it is of the form R : $\emptyset \rightarrow$ Y. This implies that the projection of a relation r over R satisfies such a NFD if and only if the projection of r onto any attribute A $\in$ Y contains at most one nonnull value. Herein we do not consider such cardinality constraints and have therefore assumed that all NFDs are standard.

Definition 3.3 (Null Inclusion Dependency) A *null inclusion dependency* (or simply a NIND) over a database schema **R** is a statement of the form R[X] $\subseteq$ S[Y], where R, S $\in$ **R** and X $\subseteq$ schema(R), Y $\subseteq$ schema(S) are sequences of attributes such that |X| = |Y|. A NIND is said to be *trivial* if it is of the form R[X] $\subseteq$ R[X].

We assume that a database schema **R** has a set of NINDs associated with it, denoted by I, and let NIND(**R**) be the set of all possible sets I of NINDs over **R**.

Definition 3.4 (Noncircular sets of NINDs) A set of NINDs over **R** is *circular* [SCIO86] if either

1. There exists a nontrivial NIND R[X] $\subseteq$ R[Y] $\in$ I.
2. There exist m distinct relation schemas, $R_1, R_2, R_3, \ldots, R_m \in \mathbf{R}$, with m > 1, such that I contains the NINDs: $R_1[X_1] \subseteq R_2[Y_2]$, $R_2[X_2] \subseteq R_3[Y_3]$, ..., $R_m[X_m] \subseteq R_1[Y_1]$.

A set of NINDs is *noncircular* if it is not circular.

The following definition is a syntactic definition of NIND satisfaction.

Definition 3.5 (Satisfaction of a NIND) A NIND R[X] $\subseteq$ S[Y] over $\mathbf{R}$ is satisfied in a database d over $\mathbf{R}$, denoted by d $\models$ R[X] $\subseteq$ S[Y], if d is consistent and $\forall t_1 \in$ r, $\exists t_2 \in$ s such that t_1[X] $\sqsubseteq$ t_2[Y], where r $\in$ d is the relation over R $\in$ $\mathbf{R}$ and s $\in$ d is the relation over S $\in$ $\mathbf{R}$.

From now on we will assume that when R[X] $\subseteq$ S[Y] $\in$ I and d is a database over $\mathbf{R}$, then r $\in$ d is the relation over R $\in$ $\mathbf{R}$ and s $\in$ d is the relation over S $\in$ $\mathbf{R}$.

We observe that if d is a complete database, then d $\models$ R[X] $\subseteq$ S[Y] if and only if the classical definition of inclusion dependency (IND) is satisfied, i. e. π_X(r) $\subseteq$ π_Y(s) [CASA84]. The justification for our definition of NIND satisfaction is that the use of the notion of less informative tuples leads to the most natural generalisation of set-inclusion.

The following proposition gives a semantic characterization of NIND satisfaction in terms of possible worlds.

Proposition 3.2 *Let d_1 be a database over* $\mathbf{R}$ *and R[X] $\subseteq$ S[Y] be a NIND over* $\mathbf{R}$. *If $d_1 \models$ R[X] $\subseteq$ S[Y], then $\exists d_2 \in$ POSS(d_1) such that $\pi_X(r) \subseteq \pi_Y(s)$, where $r \in d_2$ is the relation over $R \in$* $\mathbf{R}$ *and $s \in d_2$ is the relation s over $S \in$* $\mathbf{R}$. $\square$

We note that the converse of the above proposition does not hold. For example, let $d_1 = < r_1, r_2 >$ be a database over $\mathbf{R} = \{R_1, R_2\}$, where $r_1 = \{< a, b >\}$ is the relation over R_1, with schema(R_1) = <A, B>, and $r_2 = \{< unk, d >\}$ is the relation over R_2, with schema(R_2) = <C, D>. Then $d_1 \not\models R_1$[A] $\subseteq$ R_2[C] but $d_2 \models R_1$[A] $\subseteq$ R_2[C], where $d_2 = < r_1, s_2 > \in$ POSS(d_1) is a database over $\mathbf{R}$, with $s_2 = \{< a, d >\}$ being a relation over R_2, and $\pi_{<A>}(r_1)$ $\subseteq$ $\pi_{<C>}(s_2)$. Thus the existence of a possible world of a database, say d, that satisfies a classical inclusion dependency is not sufficient for a corresponding NIND to be satisfied in d.

Definition 3.6 (Logical implication) Let $\Sigma \subseteq$ F $\cup$ I be a set of NINDs and NFDs over $\mathbf{R}$. We say that Σ *logically implies* a NFD or a NIND α over $\mathbf{R}$, written $\Sigma \models \alpha$, whenever for all databases, d over $\mathbf{R}$, the following condition is true:

$$\text{if } \forall \beta \in \Sigma, d \models \beta \text{ holds then } d \models \alpha \text{ also holds.}$$

Definition 3.7 (Axiom system) An *axiom system* for NFDs and NINDs is a set of inference rules (or simply rules) that can be used to derive NFDs and NINDs from a given set Σ of NFDs and NINDs over $\mathbf{R}$. Given an axiom system for NFDs and NINDs, a *proof* of a NFD or a NIND α over $\mathbf{R}$, obtained from Σ, is a finite sequence of NFDs and NINDs over $\mathbf{R}$, whose last element is α, and where each NFD or NIND in the said sequence is either in Σ or follows from a finite number of previous NFDs and NINDs in the sequence by one of the inference rules. We denote by $\Sigma \vdash \alpha$ the fact that there exists a proof of α from Σ.

Definition 3.8 (Sound and complete axiom system) Let $\Sigma \subseteq \mathrm{F} \cup \mathrm{I}$ be a set of NINDs and NFDs over **R** and α be a NFD or a NIND over **R**. An axiom system is *sound* for NFDs and NINDs if $\Sigma \vdash \alpha$ implies that $\Sigma \models \alpha$ and it is *complete* for NFDs and NINDs if $\Sigma \models \alpha$ implies that $\Sigma \vdash \alpha$.

An axiom system which is sound and complete whenever Σ consists of NFDs only (respectively, NINDs only) and α is a NFD (respectively, a NIND) is said to be sound and complete for NFDs (respectively, NINDs).

We mention that Lien's and Atzeni's axiom system for NFDs contains the four inference rules: reflexivity, augmentation, union, and decomposition. Although Lien [LIEN82] and Atzeni and Morfuni [ATZE86] suggested the interpretation of "inapplicable" or "nonexistent", and "no information", respectively, for the unmarked null value, the axiom system for NFDs in the presence of *unk* remains essentially the same.

The following five inference rules constitute an axiom system which is sound and complete for NFDs [LIEN82, ATZE86]. (We have added the permutation rule to the original axiom system to take into account the fact that we are dealing with sequences of attributes rather than the usual sets of attributes.)

Definition 3.9 (Inference rules for NFDs) We define the following five inference rules for NFDs, where F is a set of NFDs over **R** and $\mathrm{R} \in \mathbf{R}$:

NFD1 Reflexivity: if $\mathrm{Y} \subseteq \mathrm{X} \subseteq \mathrm{schema(R)}$, then $\mathrm{F} \vdash \mathrm{R} : \mathrm{X} \rightarrow \mathrm{Y}$.

NFD2 Augmentation: if $\mathrm{F} \vdash \mathrm{R} : \mathrm{X} \rightarrow \mathrm{Y}$ and $\mathrm{W} \subseteq \mathrm{schema(R)}$, then $\mathrm{F} \vdash \mathrm{XW} \rightarrow \mathrm{YW}$.

NFD3 Union: if $\mathrm{F} \vdash \mathrm{R} : \mathrm{X} \rightarrow \mathrm{Y}$ and $\mathrm{F} \vdash \mathrm{R} : \mathrm{X} \rightarrow \mathrm{Z}$, then $\mathrm{F} \vdash \mathrm{R} : \mathrm{X} \rightarrow \mathrm{YZ}$.

NFD4 Decomposition: if $\mathrm{F} \vdash \mathrm{R} : \mathrm{X} \rightarrow \mathrm{YZ}$, then $\mathrm{F} \vdash \mathrm{R} : \mathrm{X} \rightarrow \mathrm{Y}$ and $\mathrm{F} \vdash \mathrm{R} : \mathrm{X} \rightarrow \mathrm{Z}$.

NFD5 Permutation: if $\mathrm{F} \vdash \mathrm{R} : \mathrm{X} \rightarrow \mathrm{Y}$, $\mathrm{W} \approx \mathrm{X}$ and $\mathrm{Z} \approx \mathrm{Y}$, then $\mathrm{F} \vdash \mathrm{W} \rightarrow \mathrm{Z}$.

The following three inference rules constitute an axiom system which is sound and complete for NINDs when only complete databases are considered [CASA84].

Definition 3.10 (Inference rules for INDs) We define the following three inference rules for INDs, where I is a set of NINDs over **R** and R, R_1, R_2, $R_3 \in \mathbf{R}$:

IND1 Reflexivity: if $\mathrm{X} \subseteq \mathrm{schema(R)}$, then $\mathrm{I} \vdash \mathrm{R[X]} \subseteq \mathrm{R[X]}$.

IND2 Projection and permutation: if $\mathrm{I} \vdash R_1[\mathrm{X}] \subseteq R_2[\mathrm{Y}]$, where $\mathrm{X} = < A_1, \ldots, A_m > \subseteq \mathrm{schema}(R_1)$, $\mathrm{Y} = < B_1, \ldots, B_m > \subseteq \mathrm{schema}(R_2)$ and $i_1, \ldots, i_k$ is a sequence of distinct integers from $\{1,\ldots,\mathrm{m}\}$, then $\mathrm{I} \vdash R_1[A_{i_1}, \ldots, A_{i_k}] \subseteq R_2[B_{i_1}, \ldots, B_{i_k}]$.

IND3 Transitivity: if $\mathrm{I} \vdash R_1[\mathrm{X}] \subseteq R_2[\mathrm{Y}]$ and $\mathrm{I} \vdash R_2[\mathrm{Y}] \subseteq R_3[\mathrm{Z}]$, then $\mathrm{I} \vdash R_1[\mathrm{X}] \subseteq R_3[\mathrm{Z}]$.

We now show that the above axiom system is also sound and complete for NINDs.

Theorem 3.3 *The axiom system comprising IND1, IND2 and IND3 is sound and complete for NINDs.*

Proof. It can be verified by the definition of the satisfaction of a NIND that the inference rules are sound.

In order to prove completeness of the axiom system we need to show that if I $\models \alpha$ then I $\vdash \alpha$, or equivalently, we need to show that if I $\not\vdash \alpha$ then I $\not\models \alpha$. Thus we need to exhibit a database d such that

$$\forall \beta \in I, d \models \beta \text{ but } d \not\models \alpha. \tag{1}$$

Now, by the completeness of the axiom system for INDs when only complete databases are involved [CASA84], $\exists d \in$ COMPLETE($\mathbf{R}$) satisfying (1). The result follows, since COMPLETE($\mathbf{R}$) $\subseteq$ DB($\mathbf{R}$). $\square$

In view of this theorem, in the sequel we shall refer to the aforesaid inference rules as inference rules for NINDs also.

4 The chase procedure for NFDs and NINDs

We extend the classical chase procedure defined over complete relations with respect to FDs [MAIE79, ATZE93] and INDs [JOHN84, MANN88] to incomplete relations with respect to NFDs and NINDs.

We show that the chase procedure can be used to test whether a database satisfies a set of NFDs and NINDs. We also show that, in general, the chase can add an exponential number of tuples to its input database. Finally, we show that the chase can always be computed by first applying the NIND chase rule (defined below) as long as possible and then applying the NFD rule (defined below) as long as possible.

We now define the said two *chase rules* (or simply rules), which are applied to a database d over $\mathbf{R}$.

NFD rule: if R : X $\rightarrow$ Y $\in$ F and $\exists t_1, t_2 \in r$ such that t_1[X] and t_2[X] are complete, t_1[X] = t_2[X] but $t_1[Y] \sqcup t_2[Y]$ is inconsistent, then change t_1[Y$-$X] and t_2[Y$-$X] to t_1[Y$-$X] $\sqcup$ t_2[Y$-$X].

NIND rule: if R[X] $\subseteq$ S[Y] $\in$ I and $\exists t_1 \in$ r such that $\nexists t_2 \in$ s with t_1[X] $\sqsubseteq$ t_2[Y], then add a tuple t_2 over S to s, with t_2[Y] = t_1[X] and $\forall$ A $\in$ schema(S) $-$ Y, t_2[A] = *unk*.

We denote the tuple t_2 such that t_2[Y] = t_1[X] and $\forall$ A $\in$ schema(S) $-$ Y, t_2[A] = *unk* by pad(t_1[X], Y, S) (cf. [MANN88]).

We now give the pseudo-code of an algorithm designated, CHASE(d, F, I), which, given an input database d over $\mathbf{R}$ together with a set of NFDs F and a set of NINDs I, applies the NFD and NIND rules to d as long as possible and returns the resulting database over $\mathbf{R}$. CHASE(d, F, I) will also denote the database resulting from the ensuing Algorithm 1.

Algorithm 1 (CHASE(d, F, I))

```
begin
  Result := d;
  Tmp := ∅;
  while Tmp ≠ Result do
    Tmp := Result;
    Apply the NFD rule or the NIND rule to Result;
```

7. **end while**
8. **while** $\exists r \in$ Result and $\exists t_1, t_2 \in r$ such that $t_1 \sqsubseteq t_2, t_1 \neq t_2$ and t_1 was added to r by an application of the NIND rule **do**
9. remove t_1 from r;
10. **end while**
11. **return** Result;
12. **end.**

Results analogous to Lemmas 4.1, 4.2 and Theorem 4.3 can be found in [JOHN84, MANN88] for FD and INDs without nulls (cf. [MAIE79, ATZE93]).

Lemma 4.1 *Algorithm 1 terminates.*

Proof. Algorithm 1 terminates, since adom(Result) $\subseteq$ adom(d), where Result denotes the state of the database just prior to the execution of line 11 of Algorithm 1. Thus no new values are generated during the execution of line 6 of Algorithm 1 implying that the algorithm must terminate after executing the while loop beginning at line 4 a finite number of times. □

Definition 4.1 (Insertability) Let d be a database over **R** with an associated set of NFDs F and an associated set of NINDs I. Furthermore, let Result be a state of the database d during the execution of Algorithm 1 and let r $\in$ Result be the relation over R $\in$ **R** and s $\in$ Result be the relation over S $\in$ **R**.

We say that a type(S)-tuple t is *insertable* into s $\in$ Result (or simply insertable if s is understood from context) if *one* of the following two conditions holds:

1. t = pad(t_1[X], Y, S), where R[X] $\subseteq$ S[Y] is a NIND in I and t_1 is a type(R)-tuple in r, and $\nexists t_2 \in$ s with $t_1[X] \sqsubseteq t_2[Y]$.
2. The same as condition (1) with t_1 being a type(R)-tuple that is insertable into r.

If a type(S)-tuple t is insertable into s as a result of condition (1) above, we also say that t is *directly insertable* into s as a result of the type(R)-tuple t_1 and the NIND R[X] $\subseteq$ S[Y].

Definition 4.2 (Degree of insertability) We define the *degree of insertability* (in Result) of a type(S)-tuple t over S, denoted by deg_ins(t), according to the two conditions of the definition of insertability being satisfied together with an additional third condition, namely

1. deg_ins(t) = 1;
2. deg_ins(t) = 1 + deg_ins(t_1);
3. otherwise, deg_ins(t) = 0.

We now show that if CHASE(d, F, I) is consistent then it is unique, i. e. the order in which the NFD rule and the NIND rule are applied to Result in Algorithm 1 does not effect the state of the returned database. We note that the order in which the rules are applied also includes the choice of the NFD in F to be used in the *if part* of the NFD rule and the choice of the NIND in I to be used in the *if part* of the NIND rule. We recall that the input database d is assumed to be consistent.

Lemma 4.2 *The database CHASE(d, F, I) returned by Algorithm 1 is unique provided CHASE(d, F, I) is consistent.*

Proof. Let Result be a state of the database d during the execution of Algorithm 1, which computes CHASE(d, F, I), and assume that d is consistent. We call an execution of line 6 in Algorithm 1 a *chase step*.

Firstly, we note that the NFD rules can be applied in any order since, if CHASE(d, F, I) is consistent, they do not modify any constants in the database.

We now make two useful observations about insertability and the degree thereof:

- Insertability of a type(S)-tuple t is preserved until the degree of insertability is reduced to zero; when the degree of insertability is one, then either t will be inserted into s by the definition of the NIND rule (possibly after some further chase steps), or a more informative tuple t', such that t $\sqsubseteq t'$, will be inserted into s (possibly after some further chase steps).
- If deg_ins(t) > 0, then it will be reduced but remain greater or equal to zero after further chase steps; in CHASE(d, F, I) deg_ins(t) $= 0$.

From the observations made above it follows that if deg_ins(t) > 0, then $\exists t' \in s'$, where s' is the final state of s in CHASE(d, F, I), such that t $\sqsubseteq t'$. The while loop of Algorithm 1 beginning at line 8 and ending at line 10 guarantees uniqueness, since if t was inserted into the relation over S in Result after one or more chase steps, then it will be removed from CHASE(d, F, I) only if a more informative tuple t', which was inserted into the relation over S in Result after some additional chase steps, were to be present in the final state of s. □

Theorem 4.3 *Let d be a consistent database over* **R** *and let* $\Sigma = F \cup I$ *be a set of NFDs and NINDs over* **R**. *Then the following statements are true:*

1. $\forall$ *R* : *X* $\rightarrow$ *Y* $\in$ *F*, *d* $\models$ *R* :*X* $\rightarrow$ *Y if and only if CHASE(d, F,* $\emptyset$*) is consistent.*
2. $\forall$ *R[X]* $\subseteq$ *S[Y]* $\in$ *I*, *d* $\models$ *R[X]* $\subseteq$ *S[Y] if and only if d = CHASE(d,* $\emptyset$*, I).*
3. $\forall \alpha \in \Sigma, d \models \alpha$ *if and only if CHASE(d, F, I) is consistent and d = CHASE(d, F, I).*
4. $\forall \alpha \in \Sigma, CHASE(d, F, I) \models \alpha$ *if and only if CHASE(d, F, I) is consistent.*

Proof. The results follow immediately from the definitions of the NFD and NIND satisfaction and Algorithm 1. □

As a result of Theorem 4.3 part (3) it can easily be shown that the condition $\forall \alpha \in \Sigma, d \models \alpha$ can be tested in polynomial time in the size of the input database d and Σ.

We observe that we do not need to make any assumption about the structure of the NINDs in I in order for CHASE(d, F, I) to terminate. When considering only complete relations then, in general, the chase does not terminate unless the set of NINDs is noncircular in which case the chase procedure is guaranteed to terminate (see [JOHN84, MANN88]).

The next theorem shows that the time complexity of CHASE(d, $\emptyset$, I) is, in general, exponential in the sizes of d and I; we measure the time complexity by the number of tuples added to the input database d by the NIND rule.

Theorem 4.4 *The number of tuples in CHASE(d, $\emptyset$, I) is, in general, exponential in the sizes of d and I.*

Proof. See Appendix A.

We note that the above theorem holds even if I is noncircular, since the set I of NINDs used in the proof is noncircular.

We can now obtain an exponential time upper bound for computing CHASE(d, F, I) as follows. Let q = |adom(d)| and let k denote the maximal cardinality of any schema(R_i), $R_i \in \mathbf{R}$. Then the maximal cardinality of CHASE(d, F, I) is bounded by q^k. Therefore, on using this polynomial time upper bound the following corollary follows from Theorem 4.4 and Lemma 4.2 together with the fact that CHASE(d, F, $\emptyset$) can be computed in polynomial time in the sizes of d and F (cf. [HONE82]).

Corollary 4.5 *The time complexity of CHASE(d, F, I) is, in general, exponential in the sizes of d, F and I.* □

We next show that for the chase we have defined we cannot apply the NFD rules first. Let $\mathbf{R} = \{R_1, R_2\}$ be a database schema, with schema(R_1) = <A, B, C> and schema(R_2) = $< A, B >$. Also, let d = $< r_1, r_2 >$ be a database over $\mathbf{R}$, with relations $r_1 = \{< a, b, c >\}$ over R_1 and $r_2 = \emptyset$ over R_2. Furthermore let F = $\{R_2 :$ A $\rightarrow$ B$\}$ and I = $\{R_1$[A, B] $\subseteq R_2$[A, B], R_1[A, C] $\subseteq R_2$[A, B]$\}$.

It can easily be verified that

$$CHASE(d, F, I) \neq CHASE(CHASE(d, F, \emptyset), \emptyset, I),$$

since the left-hand side of the equation results in an inconsistent database over $\mathbf{R}$ while the right-hand side of the equation results in a consistent database over $\mathbf{R}$.

We close this section by showing that the chase we have defined can be computed by first applying the NIND rule as long as possible and then applying the NFD rule as long as possible, even if the set of NINDs is circular (cf. [MANN88] for a similar result concerning noncircular INDs).

Lemma 4.6 *Let d be a database over* $\mathbf{R}$, *F be a set of NFDs over* $\mathbf{R}$ *and I be a set of NINDs over* $\mathbf{R}$. *Then the following two statements are true:*

1. *CHASE(d, F, I) is consistent if and only if CHASE(CHASE(d, $\emptyset$, I), F, $\emptyset$) is consistent.*
2. *if CHASE(d, F, I) is consistent, then CHASE(d, F, I) = CHASE(CHASE(d, $\emptyset$, I), F, $\emptyset$).*

Proof. The result follows by the definition of the NFD rule, since due to the consistency of CHASE(d, F, I) an application of the NFD rule to any state, say d', of the database d during the execution of Algorithm 1, which computes CHASE(d, F, I), does not modify d'. □

5 A sound and complete axiom system for NFDs and NINDs

Mitchell [MITC83] presented a sound and complete axiom system for FDs and INDs; in this axiom system attributes can be added to relation schemas by the *attribute introduction* inference rule. It turns out that an unbounded number of attributes may be added to a database schema during a proof using Mitchell's axiom system. This leads to the undecidability of the implication problem for FDs and INDs thus hindering the practical use of Mitchell's axiom system in any real-life application. Furthermore, Mitchell's axiom system contains the *pullback* inference rule, by which a new FD is derived from a FD and an IND, and the *collection* inference rule, which derives an IND from two INDs and a FD.

In the presence of incomplete information the pullback rule is also sound for NFDs and NINDs, namely

NFD-NIND1 Pullback: if $\Sigma \vdash$ R[WZ] $\subseteq$ S[XY], with |X| = |W|, and $\Sigma \vdash$ S : X $\rightarrow$ Y, then $\Sigma \vdash$ R : W $\rightarrow$ Z, where Σ = F $\cup$ I.

However, in the presence of incomplete information the collection rule is unsound for NFDs and NINDs. The collection rule states that the set of FDs and INDs, {R[UV] $\subseteq$ S[XY], R[UW] $\subseteq$ S[XZ], S : X $\rightarrow$ Y}, logically implies the IND R[UVW] $\subseteq$ S[XYZ] [MITC83]. Let F = {S : D $\rightarrow$ E} and I = {R[AB] $\subseteq$ S[DE], R[AC] $\subseteq$ S[DF]}, where schema(R) = <A, B, C> and schema(S) = <D, E, F>. Furthermore, let d = <r, s>, with r being the relation over R, shown in Table 5, and s being the relation over S, shown in Table 6. It can be verified that $\forall \alpha \in$ F $\cup$ I, d $\models \alpha$, but that d $\not\models$ R[ABC] $\subseteq$ S[DEF], which follows from the said collection rule when only complete databases are considered.

A	B	C
0	0	0

Table 5: The counterexample relation r

D	E	F
0	0	*unk*
0	*unk*	0

Table 6: The counterexample relation s

We next show that the example proof shown in [MITC83, pages 161, 162], which was used to justify the attribute introduction rule defined therein, is unsound for NFDs and NINDs. Let d = $< r >$, where r = $\{< a, b, unk, unk >\}$ is a relation over R with schema(R) = <A, B, C, D>, F = {R : C $\rightarrow$ D} and I = {R[AB] $\subseteq$ R[CD], R[BA] $\subseteq$ R[CD], R[B] $\subseteq$ R[A]}. By Theorem 4.3 CHASE(d, F, I), which is shown in Table 7, satisfies the NFD in F and the NINDs in I. Furthermore, it can be verified that CHASE(d, F, I) does not satisfy the NIND, R[AB] $\subseteq$ R[BA], and thus F $\cup$ I $\not\models$ R[AB] $\subseteq$ R[BA].

Although we have just shown that the collection rule is unsound for NFDs and NINDs, the following new inference rule, which we call the *null collection* inference rule, can be shown to be sound for NFDs and NINDs.

NFD-NIND2 Null collection: if $\Sigma \vdash$ S : $Z_3 \rightarrow Z_1$, $\Sigma \vdash$ R[W_2W_1W] $\subseteq$ S[Z_1Z_2Z], $\Sigma \vdash$ R[W_1W_3] $\subseteq$ S[Z_1Z_3], $\Sigma \vdash$ R[W_2W_3] $\subseteq$ S[Z_1Z_3] and $\forall$ i $\in$ {1,...,m}, $\Sigma \vdash$ R[X$-A_i$] $\subseteq$ S[Y$-B_i$], then $\Sigma \vdash$ R[X] $\subseteq$ S[Y], where Σ = F $\cup$ I, X = $W_1W_2W \subseteq$ schema(R), $W_3 \subseteq W$, Y = $Z_1Z_2Z \subseteq$ schema(S), $Z_3 \subseteq Z$, $|W_1| = |Z_1| = |W_2| = |Z_2| \geq 1$, $|W_3| = |Z_3| \geq 1$, $W_1W_2W_3 = < A_1, \ldots, A_m >$ and $Z_1Z_2Z_3 = < B_1, \ldots, B_m >$.

A	B	C	D
a	b	*unk*	*unk*
unk	*unk*	a	b
unk	*unk*	b	a
b	*unk*	*unk*	*unk*

Table 7: The counterexample database CHASE(d, F, I)

We note that we cannot drop the NFD or any of the NINDs in the *if part* of the null collection rule noting that S : $Z_3 \rightarrow Z_1$ is a nontrivial NFD. (We recall that at the beginning of Section 2 we have assumed that concatenated sequences of attributes have no common attributes.)

The NFD S : $Z_3 \rightarrow Z_1$ cannot be dropped, since in order to make the inference we need to establish the equality of the Z_1-values and the Z_2-values of type(S)-tuples. The NIND R$[W_2W_1W] \subseteq$ S$[Z_1Z_2Z]$ cannot be dropped because of Lemma 5.4 below. The NINDs R$[W_1W_3] \subseteq$ S$[Z_1Z_3]$ and R$[W_2W_3] \subseteq$ S$[Z_1Z_3]$ cannot be dropped, otherwise the NFD S : $Z_3 \rightarrow Z_1$ cannot be utilized due to the fact that, in general, it is not the case that if the ith attribute of W_3 is the jth attribute of W, then it is also the case that the ith attribute of Z_3 is the jth attribute of Z. Finally, we cannot drop any of R[X$-A_i$] $\subseteq$ S[Y$-B_i$], $\forall$ i $\in$ {1,. . .,m}, since otherwise by setting *unk* to be the A_i-value of type(R)-tuples we will not be able to utilize the NFD S : $Z_3 \rightarrow Z_1$ in the inference.

A special case of the null collection rule which we will utilise in the proof of Proposition 5.8 is the situation when W = W_3W_4 and Z = Z_3Z_4. In this case the NINDs R$[W_1W_3] \subseteq$ S$[Z_1Z_3]$ and R$[W_2W_3] \subseteq$ S$[Z_1Z_3]$ can be dropped from the definition of NFD-NIND2, since R$[W_1W_3]$ $\subseteq$ S$[Z_1Z_3]$ can be derived from R[X$-A_j$] $\subseteq$ S[Y$-B_j$], where $A_j \in W_2$ and $B_j \in Z_2$, and R$[W_2W_3] \subseteq$ S$[Z_1Z_3]$ can be derived from R$[W_2W_1W_3W_4] \subseteq$ R$[Z_1Z_2Z_3Z_4]$. Therefore, in this case a simpler version of the null collection rule obtains.

As an example of the use of null collection, let R = $< A_1, A_2, A_3 >$ and S = $< B_1, B_2, B_3 >$. Furthermore, let Σ = {R$[A_1, A_2] \subseteq$ S$[B_1, B_2]$, R$[A_1, A_3] \subseteq$ S$[B_1, B_3]$, R$[A_2, A_3] \subseteq$ S$[B_2, B_3]$, R$[A_2, A_1, A_3] \subseteq$ S$[B_1, B_2, B_3]$, S : $B_3 \rightarrow B_1$}. It can be verified that $\Sigma \models$ R$[A_1, A_2, A_3]$ $\subseteq$ S$[B_1, B_2, B_3]$ and in addition $\Sigma \vdash$ R$[A_1, A_2, A_3] \subseteq$ S$[B_1, B_2, B_3]$, on using null collection. Furthermore, the reader can verify that this NIND cannot be derived by the inference rules of our axiom system without using the null collection rule.

In order to provide a proof procedure for NFDs and NINDs we now modify the chase procedure, defined in Section 4, in order to deal with *marked nulls* [MAIE79, HONE82]. Such nulls afford us the ability to infer equality between attribute values.

Definition 5.1 The *marked domain* of a relation schema R, denoted by *Mdom*(R), is the finite set $\{unk, \perp_1, \ldots, \perp_q\}$, where q = type(R) +1; each $\perp_i$, i $\in$ {1,. . ., q}, is called a *marked null*. We extend the partial order $\sqsubseteq$ to *Mdom*(R) as follows:

$$\forall v \in Dom - Mdom(R), unk \sqsubseteq \perp_q \sqsubseteq \ldots \sqsubseteq \perp_1 \sqsubseteq v, v \not\sqsubseteq \perp_1 \not\sqsubseteq \ldots \not\sqsubseteq \perp_q \not\sqsubseteq unk.$$

Since we have assumed that herein we deal with only standard NFDs, it is obvious that type(R) +1 marked nulls suffice.

In the sequel we call a relation, which is defined over *Mdom*(R), a *marked relation* and a database consisting of a sequence of marked relations a *marked database*. We observe that a marked database is always consistent, since *Mdom*(R) does not contain any nonnull values.

We now modify the NFD rule as follows.

Modified NFD rule: if R : X $\rightarrow$ Y $\in$ F and $\exists t_1, t_2 \in$ r, where r is a marked relation in a marked database d such that $\forall$ B $\in$ X, t_1[B] and t_2[B] are marked nulls, t_1[X] = t_2[X] but $t_1[Y] \not\cong t_2[Y]$, then $\forall$ A $\in$ Y−X, if t_1[A] $\neq$ *unk* and t_2[A] $\neq$ *unk*, then change all occurrences of t_1[A] and t_2[A] *in d* to t_1[A] $\sqcup$ t_2[A].

In the sequel NFD rule and modified NFD rule will be used interchangeably, as long as no ambiguity arises. We now modify the chase procedure to take into account the modified NFD rule.

Definition 5.2 (Modified chase) The chase of a marked database d over **R** with respect to F and I, which is computed by Algorithm 1 and such that the modified NFD rule is applied instead of the NFD rule, is called the *modified chase* and is denoted by MCHASE(d, F, I).

We note that Lemmas 4.1 and 4.2 can be shown to hold for the modified chase. We briefly outline how this can be done.

Lemma 4.1, asserting that Algorithm 1 terminates when applying the modified NFD rule, holds since the modified chase does not introduce any new values during its computation. Now, since d does not contain any nonnull values, it is always the case that MCHASE(d, F, I) is consistent. Therefore, Lemma 4.2, asserting that the database MCHASE(d, F, I) returned by Algorithm 1 when applying the modified NFD rule is unique, holds since there is an equivalence relation amongst the marked nulls in adom(d) that are equated by the modified NFD rule and thus all the marked nulls in a particular equivalence class will be changed to the least upper bound of that equivalence class prior to the termination of Algorithm 1 (cf. [MAIE79, JOHN84, ATZE93]).

Lemma 4.6 does not hold for the modified chase and thus the modified chase cannot be computed by first applying the NIND rule as long as possible and then applying the NFD rule as long as possible. Let r, s_1 and s_2 be the relations shown in Tables 8, 9 and 10, respectively, and let d = $< r, s_1, s_2 >$ be a database over **R**, with $\mathbf{R} = \{R, S_1, S_2\}$ and schema(R) = $< A >$, schema(S_1) = $< B >$ and schema(S_2) = $< C, D >$. Furthermore, let F = {S_2 : C $\rightarrow$ D} and let I = {R[A] $\subseteq$ S_1[B]}. It can be verified that

$$MCHASE(d, F, I) \neq MCHASE(MCHASE(d, \emptyset, I), F, \emptyset),$$

since $\perp_3 \sqsubseteq \perp_2$ and thus MCHASE(d, $\emptyset$, I) = d.

A
$\perp_3$

Table 8: The counterexample relation r

B
$\perp_2$

Table 9: The counterexample relation s_1

C	D
$\perp_2$	$\perp_1$
$\perp_2$	$\perp_3$

Table 10: The counterexample relation s_2

The following theorem concerning the modified chase gives the result corresponding to Theorem 4.3.

Theorem 5.1 *Let d be a marked database over* **R** *and let* $\Sigma = F \cup I$ *be a set of NFDs and NINDs over* **R**. *Then the following two statements are true:*

1. $\forall$ *R[X]* $\subseteq$ *S[Y]* $\in$ *I, d* $\models$ *R[X]* $\subseteq$ *S[Y] if and only if d = MCHASE(d,* $\emptyset$*, I).*
2. $\forall \alpha \in \Sigma, MCHASE(d, F, I) \models \alpha$. □

We note that MCHASE(d, F, I) is consistent, since adom(d) does not contain any nonnull values. Furthermore, we note that by Corollary 4.5 the size of MCHASE(d, F, I) is, in general, exponential in the sizes of d and Σ.

Definition 5.3 (The database for a NFD and a NIND) Let σ be the NFD R : X $\rightarrow$ A over **R**, where A $\in$ schema(R). The *database for* σ, denoted by d_σ, with $|d_\sigma|$ = n, is the marked database $< \emptyset, \ldots, \emptyset, r_\sigma, \emptyset, \ldots, \emptyset >$, where $r_\sigma = \{t_1, t_2\}$ over R is the marked relation shown in Table 11, with |X| = m.

Let τ be the NIND R[X] $\subseteq$ S[Y] over **R**, with |X| = m. The *ith tuple* with respect to τ, denoted by t^i_τ, where i $\in \{0,\ldots, 2^{m-1}\}$ (or simply t_i whenever τ is understood from context), is the type(R)-tuple defined as follows:

$\forall A_j \in$ schema(R) $-$ X, $t_i[A_j]$ = *unk*, and $\forall A_j \in$ X, $t_i[A_j] = \perp_j$ if the jth bit of the binary representation of i using m bits is 0, otherwise $t_i[A_j]$ = *unk*.

The *ith database for* τ, where i $\in \{0,\ldots, 2^{m-1}\}$, denoted by d^i_τ, with $|d^i_\tau|$ = n, is the marked database $< \emptyset, \ldots, \emptyset, r_i, \emptyset, \ldots, \emptyset >$, where $r_i = \{t_i\}$ is the marked relation over R.

As an example the 0th database for R[X] $\subseteq$ S[Y] is shown in Table 12.

	X	A	schema(R) $-$ XA
t_1:	$\perp_1 \ldots \perp_m$	$\perp_{m+1}$	$unk \ldots unk$
t_2:	$\perp_1 \ldots \perp_m$	$\perp_{m+2}$	$unk \ldots unk$

Table 11: The database for X $\rightarrow$ Y

	X	schema(R) $-$ X
t_0:	$\perp_1 \ldots \perp_m$	$unk \ldots unk$

Table 12: The 0th database for R[X] $\subseteq$ S[Y]

The following lemma follows from the fact that NFD1, ..., NFD5 constitute a sound and complete axiom system for NFDs.

Lemma 5.2 *Let F be a set of NFDs over* **R**. *Then* $F \models \sigma$*, where* σ *is the NFD R : X* $\rightarrow$ *A, if and only if* $t'_1[A] = t'_2[A]$*, where* $t'_1, t'_2 \in r'_\sigma$ *are the final states of* t_1 *and* t_2*, respectively, and* r'_σ *is the final state of* r_σ *in MCHASE(*d_σ*, F,* $\emptyset$*).* □

For a similar result for FDs holding in complete relations see [MAIE79, Theorem 5].

The following lemma shows the corresponding result when only NINDs are present in Σ.

Lemma 5.3 *Let I be a set of NINDs over* $\mathbf{R}$*. Then* $I \models \tau$*, where* τ *is the NIND* $R[X] \subseteq S[Y]$*, if and only if* $\exists t' \in s'_\tau$ *such that* $t_0[X] \sqsubseteq t'[Y]$*, where* s_τ *is the relation over S in* d^0_τ*,* s'_τ *is the final state of* s_τ *in MCHASE(*d^0_τ*,* $\emptyset$*, I) and* $|X| = m$*.*

Proof. The *only if part* of the lemma follows directly from Theorem 5.1 part (1) and the definition of logical implication, since the marked database MCHASE(d^0_τ, $\emptyset$, I) can be shown to be isomorphic to a database (without marked nulls) that satisfies all the NINDs in I by mapping each marked null in MCHASE(d^0_τ, $\emptyset$, I) to a nonnull value in *Dom*.

We now proceed to prove the *if part* of the lemma. We assume that τ is a nontrivial NIND, otherwise the result trivially holds. By Theorem 5.1 part (2) $\forall \alpha \in$ I, MCHASE(d^0_τ, $\emptyset$, I) $\models \alpha$. Let d be a database over $\mathbf{R}$ such that $\forall \alpha \in$ I, d $\models \alpha$; we show that it is also the case that d $\models \tau$.

Let u $\in$ r, where r $\in$ d is the relation over R. We claim that $\exists$ w $\in$ s, where s $\in$ d is the relation over S, such that u[X] $\sqsubseteq$ w[Y] and thus d $\models \tau$ as required.

We conclude the result by induction on the number of chase steps, say k, required to compute MCHASE(d^0_τ, $\emptyset$, I).

(*Basis*): If k = 1, then the only chase step executed is an application of the NIND rule for a NIND R[W] $\subseteq$ S[Z] $\in$ I, where R[X] $\subseteq$ S[Y] can be obtained from R[W] $\subseteq$ S[Z] by the projection and permutation rule for NINDs. The result follows, since d $\models$ R[W] $\subseteq$ S[Z] holds by assumption.

(*Induction*): Assume the result holds when the number of chase steps required to compute MCHASE(d^0_τ, $\emptyset$, I) is k, with k $\geq$ 1; we then need to prove that the result holds when the number of chase steps required to compute MCHASE(d^0_τ, $\emptyset$, I) is k+1.

Let us consider the last chase step executed to obtain MCHASE(d^0_τ, $\emptyset$, I). The last chase step must have been an application of the NIND rule for a NIND T[W] $\subseteq$ S[Z], with Y $\subseteq$ Z. By inductive hypothesis there exists a NIND R[V] $\subseteq$ T[W] that is logically implied by I, with X $\subseteq$ V. Thus, the NIND R[X] $\subseteq$ S[Y] can be inferred from R[V] $\subseteq$ T[W] and T[W] $\subseteq$ S[Z] by the transitivity and projection and permutation rules for NINDs.

Therefore, by induction hypothesis $\exists u'$ in the relation, over T, in d such that u[V] $\sqsubseteq$ u'[W]. Furthermore, since d $\models$ T[W] $\subseteq$ S[Z], it follows that $\exists$ w $\in$ s such that u'[W] $\sqsubseteq$ w[Z]. By the transitivity of $\sqsubseteq$ the result follows. $\square$

The following lemma shows that if a NIND R[X] $\subseteq$ S[Y] can be inferred from a set of NFDs and NINDs, then there exists a NIND of the form R[W] $\subseteq$ S[Z] with X $\approx$ W that can be inferred from the set of NINDs on its own.

Lemma 5.4 *Let* $\Sigma = F \cup I$ *be a set of NFDs and NINDs over* $\mathbf{R}$ *and let* τ *be the NIND* $R[X] \subseteq S[Y]$ *over* $\mathbf{R}$*. In addition, assume that there does not exist a NIND over* $\mathbf{R}$ *of the form* $R[W] \subseteq S[Y]$*, with* $X \approx W$*, such that* $I \models R[W] \subseteq S[Y]$*. Then* $\Sigma \not\models \tau$*.*

Proof. In order to show that $\Sigma \not\models \tau$ we exhibit a database, d over $\mathbf{R}$, such that $\forall \alpha \in \Sigma$, d $\models \alpha$ but d $\not\models \tau$. Let d' be the database $< \emptyset, \ldots, \emptyset, r, s, \emptyset, \ldots, \emptyset >$ over $\mathbf{R}$, with $|d'|$ = n, where r $\in d'$ is the relation over R shown in Table 13 and s $\in d'$ is the relation over S shown in Table 14. (In Tables 13 and 14 we have assumed that |X| = |Y| = 4, schema(R) = X and schema(S) = Y; r and s generalise in a straightforward manner to the case where |X| = |Y| = m, X $\subseteq$ schema(R) and Y $\subseteq$ schema(S).)

Next let d be the database CHASE(d', $\emptyset$, I). The database d is consistent, since adom(d') = {a, *unk*} and thus by the definition of the satisfaction of a NFD, no NFD in F can be violated. Thus, by Theorem 4.3 part (3), d satisfies all the NFDs in F and all the NINDs in I.

The result now follows by the definition of the satisfaction of a NIND on using the definition of logical implication, since there does not exist a tuple of the form <a, a, a, a> in the relation over S in d. □

A_1	A_2	A_3	A_4
a	a	a	a

Table 13: The counterexample relation r

B_1	B_2	B_3	B_4
a	a	a	*unk*
a	a	*unk*	a
a	*unk*	a	a
unk	a	a	a

Table 14: The counterexample relation s

We now define the notion of a *containment mapping* which is a mapping from marked databases to databases (cf. [ATZE93]).

Definition 5.4 (Containment mapping) A *containment mapping*, ψ, is a mapping from *Mdom*(R) to *Dom* satisfying the following two conditions:

1. each $\perp_i$ is mapped by ψ to a nonnull value;

2. *unk* is mapped by ψ to either *unk* or to a nonnull value.

We extend ψ to type(R)-tuples t, where schema(R) = $< A_1, \ldots, A_{type(R)} >$, as follows: ψ(t) = $<$ ψ(t(A_1)), ..., ψ(t($A_{type(R)}$))$>$, where each occurrence of *unk* is mapped (separately) by ψ either to *unk* or to a nonnull value.

Correspondingly we extend ψ to marked relations r over R, as follows: ψ(r) = {ψ(t) | t $\in$ r}. Finally, we extend ψ to marked databases d = $< r_1, \ldots, r_n >$ over **R** as follows: ψ(d) = $<$ $\psi(r_1)$, ..., $\psi(r_n)$$>$.

The following theorem shows that the modified chase can be also be viewed as an inference procedure.

Theorem 5.5 *Let F be a set of NFDs over* **R**, *I be a set of NINDs over* **R** *and* $\Sigma = F \cup I$. *The following two statements are true:*

1. $\Sigma \models \sigma$, *where* σ *is the NFD* $R : X \rightarrow A$, *if and only if* $t'_1[A] = t'_2[A]$, *where* $t'_1, t'_2 \in r'_\sigma$ *are the final states of* $t_1, t_2 \in r_\sigma$, *respectively, and* r'_σ *is the final state of* r_σ *in MCHASE*(d_σ, *F, I).*

2. $\Sigma \models \tau$, *where* τ *is the NIND R[X]* $\subseteq$ *S[Y], if and only if* $\forall$ $i \in \{0, \ldots, 2^{m-1}\}$, $\exists t'_i \in s'_\tau$ *such that* $t_i[X] \sqsubseteq t'_i[Y]$, *where* s_τ *is the relation over S in* d^i_τ, s'_τ *is the final state of* s_τ *in MCHASE*(d^i_τ, *F, I) and* |X| = m.

Proof. We consider the NFD σ and the NIND τ separately. In the proof we let R, S, T $\in$ **R**.

(*If part for the NFD case*): We assume that σ is a nontrivial NFD, otherwise the result trivially holds. By Theorem 5.1 part (2) $\forall\alpha \in \Sigma$, MCHASE(d_σ, F, I) $\models \alpha$ since MCHASE(d_σ, F , I) is consistent due to the fact that adom(d_σ) does not contain any nonnull values. Next let d be a database over **R** such that $\forall\alpha \in \Sigma, d \models \alpha$; we show that it is also the case that d $\models \sigma$.

Let $u_1, u_2 \in$ r, where r $\in$ d is the relation over R, such that u_1[X] and u_2[X] are complete and u_1[X] = u_2[X]. We claim that u_1[A] $\sqcup$ u_2[A] is consistent and thus d $\models \sigma$ as required. Now, if one of u_1[A] or u_2[A] is *unk* then the result follows, so assume that both u_1[A] and u_2[A] are nonnull.

Next let ψ be a containment mapping from d_σ to d such that $\psi(t_1) = u_1$ and $\psi(t_2) = u_2$. In order to prove the result we need to show that ψ is also a containment mapping from MCHASE(d_σ, F, I) to d, since then $\psi(t_1'[\mathrm{A}]) = \psi(t_2'[\mathrm{A}])$ implies that u_1[A] = u_2[A], as required.

We prove the result by induction on the number of chase steps, say k, required to compute MCHASE(d_σ, F, I).

(*Basis*): If k = 1, then the only chase step executed is an application of the NFD rule for a NFD R : W $\rightarrow$ Z $\in$ F, where W $\subseteq$ X and A $\in$ Z. It follows that u_1[A] = u_2[A], as required, since d $\models$ R : W $\rightarrow$ Z holds.

(*Induction*): Assume the result holds when the number of chase steps required to compute MCHASE(d_σ, F, I) is k, where k $\geq$ 1; we then need to prove that the result holds when the number of chase steps required to compute MCHASE(d_σ, F, I) is k+1.

Let us consider the last chase step executed to obtain MCHASE(d_σ, F, I). There are two cases to consider.

In the *first case* the last chase step is an application of the NIND rule. Now, since no changes are made to the current states of t_1 and t_2, say $\hat{t_1}$ and $\hat{t_2}$, by the execution of the last chase step, it must be the case that $\hat{t_1}[\mathrm{A}] = \hat{t_2}[\mathrm{A}] = t_1'[\mathrm{A}] = t_2'[\mathrm{A}]$ prior to the execution of the last chase step. The result therefore follows by inductive hypothesis.

In the *second case* the last chase step is an application of the NFD rule for a NFD S : W $\rightarrow$ Z $\in$ F.

Let w_1 and w_2 be the two tuples in the relation over S in the current state of d_σ, prior to the execution of the last chase step, such that either w_1 or w_2 or both of w_1 and w_2 are modified by the application of the NFD rule. Thus, $\exists$ B $\in$ Z such that either w_1[B] or w_2[B] is modified by the application of the said NFD rule and thus both the values w_1[B] and w_2[B] are marked nulls. Furthermore, $w_1[\mathrm{B}] \sqcup w_2[\mathrm{B}] = t_1'[\mathrm{A}] = t_2'[\mathrm{A}]$.

Now, since w_1[W] and w_2[W] contain only marked nulls and w_1[W] = w_2[W], it follows by inductive hypothesis that $\psi(w_1[\mathrm{W}])$ and $\psi(w_2[\mathrm{W}])$ are complete, with $\psi(w_1[\mathrm{W}]) = \psi(w_2[\mathrm{W}])$, and that both $\psi(w_1[\mathrm{B}])$ and $\psi(w_2[\mathrm{B}])$ are also complete. Thus, $\psi(w_1[\mathrm{B}]) \sqcup \psi(w_2[\mathrm{B}]) = \psi(t_1'[\mathrm{A}]) = \psi(t_2'[\mathrm{A}]) = u_1[\mathrm{A}] = u_2[\mathrm{A}]$ also holds as required, since d $\models$ S : W $\rightarrow$ Z.

(*If part for the NIND case*): We assume that τ is a nontrivial NIND, otherwise the result trivially holds. By Theorem 5.1 part (2) $\forall\alpha \in \Sigma$, MCHASE(d_τ^i, F, I) $\models \alpha$ since MCHASE(d_τ^i, F , I) is consistent due to the fact that adom(d_τ^i) does not contain any nonnull values. Next let d be a database over **R** such that $\forall\alpha \in \Sigma, d \models \alpha$; we show that it is also the case that d $\models \tau$.

Let u $\in$ r, where r $\in$ d is the relation over R. We claim that $\exists$ w $\in$ s, where s $\in$ d is the relation over S, such that u[X] $\sqsubseteq$ w[Y] and thus d $\models \tau$ as required.

Now, let i $\in \{0,\ldots,2^{m-1}\}$ be the natural number such that the jth bit of the binary representation of i using m = |X| bits is 0 if and only if u[A_j] $\neq$ *unk*. (See Definition 5.3.) Furthermore, let ψ be a containment mapping from d^i_τ to d such that $\psi(t_i)$ = u. In order to prove the result we show that ψ is also a containment mapping from MCHASE(d^i_τ, F, I) to d, since then $\psi(t'_i)$ = w holds as required.

We prove the claim and thus conclude the result by induction on the number of chase steps, k, required to compute MCHASE(d^i_τ, F, I).

(*Basis*): If k = 1, then the only chase step executed is an application of the NIND rule for a NIND R[W] $\subseteq$ S[Z] $\in$ I, where R[X] $\subseteq$ S[Y] can be obtained from R[W] $\subseteq$ S[Z] by the projection and permutation rule for NINDs. The result follows, since d $\models$ R[W] $\subseteq$ S[Z] holds by assumption.

(*Induction*): Assume the result holds when the number of chase steps required to compute MCHASE(d^i_τ, F, I) is k, where k $\geq$ 1; we then need to prove that the result holds when the number of chase steps required to compute MCHASE(d^i_τ, F, I) is k+1.

Let us consider the last chase step executed to obtain MCHASE(d^i_τ, F, I). There are two cases to consider.

In the *first case* the last chase step is an application of the NIND rule for a NIND T[W] $\subseteq$ S[Z], with Y $\subseteq$ Z. Now, by inductive hypothesis there exists a NIND R[V] $\subseteq$ T[W] that is logically implied by Σ, with X $\subseteq$ V. It therefore follows that R[X] $\subseteq$ S[Y] can be inferred from R[V] $\subseteq$ T[W] and T[W] $\subseteq$ S[Z] by the transitivity and projection and permutation rules for NINDs.

Thus, by the induction hypothesis $\exists u'$ in the relation over T in d such that u[V] $\sqsubseteq$ u'[W]. Furthermore, since d $\models$ T[W] $\subseteq$ S[Z], it follows that $\exists$ w $\in$ s such that u'[W] $\sqsubseteq$ w[Z]. By the transitivity of $\sqsubseteq$ the result follows.

In the *second case* the last chase step is an application of the NFD rule for a NFD S : W $\rightarrow$ Z $\in$ F.

Let w_1 and w_2 be the two tuples in the relation over S in the current state of d^i_τ prior to the execution of the last chase step. We assume without loss of generality that w_1 is the current state of t'_i prior to the execution of the last chase step. We also assume without loss of generality that $\forall$ A $\in$ Z, w_1[A] is changed by the last chase step; otherwise we choose the maximal subset of Z satisfying this assumption to be the right-hand side of the NFD for which the NFD rule is applied.

By inductive hypothesis it follows that $\psi(w_1)$, $\psi(w_2)$ $\in$ s and also that $\psi(w_1)$[WZ] and $\psi(w_2)$[WZ] are complete, and $\psi(w_1[\mathrm{W}]) = \psi(w_2[\mathrm{W}])$. Furthermore, by inductive hypothesis, $\forall$ A $\in$ X, $\forall$ B $\in$ Y, such that t_i[A] $\neq$ *unk*, t_i[A] $\sqsubseteq$ w_1[B] if and only if u[A] $\sqsubseteq$ $\psi(w_1[\mathrm{B}])$. The result now follows, since d $\models$ S : W $\rightarrow$ Z implies that $\psi(w_1[\mathrm{Z}]) = \psi(w_2[\mathrm{Z}])$ and thus $\psi(t'_i)$ = w as required.

(*Only if part for the NFD case*): The only if part of the theorem follows directly from Theorem 5.1 part (2) and the definition of logical implication, since the marked database MCHASE(d_σ, F, I) can be shown to be isomorphic to a database (without marked nulls) that satisfies all the NFDs and NINDs in Σ by mapping each marked null in MCHASE(d_σ, F, I) to a nonnull value in *Dom*.

However, if the A-values are not equal in the final state of r_σ, then we can construct a counterexample database satisfying all $\alpha \in \Sigma$ but violating σ by using an appropriate containment mapping.

(*Only if part for the NIND case*): The only if part of the theorem follows directly from

Theorem 5.1 part (2) and the definition of logical implication, since the marked database MCHASE(d^i_τ, F, I) can be shown to be isomorphic to a database (without marked nulls) that satisfies all the NFDs and NINDs in Σ by mapping each marked null in MCHASE(d^i_τ, F, I) to a nonnull value in *Dom*.

However, if there exists an i $\in \{0,\ldots,2^{m-1}\}$ such that there does not exist a tuple t'_i in the final state of s_τ such that t_i[X] $\sqsubseteq t'_i$[Y], then we can construct a counterexample database satisfying all $\alpha \in \Sigma$ but violating τ by using an appropriate containment mapping. $\square$

The following corollary is an immediate result of Theorem 5.5.

Corollary 5.6 *Let F be a set of NFDs over* **R**, *I be a set of NINDs over* **R** *and* $\Sigma = F \cup I$. *The following two inference rules are sound and complete for NFDs and NINDs:*

NFD inference. *Chase NFD:* $\Sigma \vdash R : X \rightarrow Y$, *if* $\forall\ A \in Y$, *the* if part *of Theorem 5.5 part (1) holds, with* σ *being the NFD,* $R : X \rightarrow A$.

NIND inference. *Chase NIND:* $\Sigma \vdash R[X] \subseteq S[Y]$, *if the* if part *of Theorem 5.5 part (2) holds, with* τ *being the NIND,* $R[X] \subseteq S[Y]$. $\square$

The following theorem utilises the previous results to exhibit a more refined sound and complete axiom system for NFDs and NINDs than that in Corollary 5.6.

Theorem 5.7 *The axiom system comprising inference rules, NFD1 to NFD5, IND1 to IND3, NFD-NIND1 and NFD-NIND2 is sound and complete for NFDs and NINDs.*

Proof. Soundness has already been established, it thus remains to show that the axiom system is complete. That is, we need to show that given a set, Σ = F $\cup$ I, where F is a set of NFDs and I is a set of NINDs both over **R**, and a NFD or a NIND, α, $\Sigma \models \alpha$ implies that $\Sigma \vdash \alpha$. We prove the result by considering the two cases separately. As in the previous theorem, we let R, S, T $\in$ **R**.

(*The NFD case*): Let α be the NFD R : X $\rightarrow$ Y. We assume that Y $\not\subseteq$ X, otherwise $\Sigma \vdash \sigma$ follows by the reflexivity and permutation rules. By the decomposition and union rules we obtain that $\Sigma \vdash$ R : X $\rightarrow$ Y if and only if $\forall$ A $\in$ Y, $\Sigma \vdash$ R : X $\rightarrow$ A. Furthermore, $\Sigma \models$ R : X $\rightarrow$ Y if and only if $\forall$ A $\in$ Y, $\Sigma \models$ R : X $\rightarrow$ A. It therefore remains to show that $\forall$ A $\in$ Y, $\Sigma \models$ R : X $\rightarrow$ A implies $\Sigma \vdash$ R : X $\rightarrow$ A.

We prove the result, namely that $\Sigma \models \sigma$ implies $\Sigma \vdash \sigma$, where σ is the NFD R : X $\rightarrow$ A and A $\in$ Y, by showing that the *if part* of Theorem 5.5 part (1) implies $\Sigma \vdash \sigma$.

We obtain the result by induction on the number of chase steps, say k, required to compute MCHASE(d_σ, F, I).

(*Basis*): If k = 1, then the only chase step executed is an application of the NFD rule for a NFD R : W $\rightarrow$ Z $\in$ F, where W $\subseteq$ X and A $\in$ Z. The result follows by applying the augmentation rule to obtain $\Sigma \vdash$ R : X $\rightarrow$ Z and then the decomposition rule to obtain $\Sigma \vdash$ R : X $\rightarrow$ A.

(*Induction*): Assume the result holds when the number of chase steps required to compute MCHASE(d_σ, F, I) is k, where k $\geq$ 1; we then need to prove that the result holds when the number of chase steps required to compute MCHASE(d_σ, F, I) is k+1.

Let us consider the last chase step executed to obtain MCHASE(d_σ, F, I). There are two cases to consider.

In the *first case* the last chase step is an application of the NIND rule. As in the *if part* for the NFD case in the proof of Theorem 5.5 the result follows by inductive hypothesis, since the A-values of the final states of $t_1, t_2 \in r_\sigma$ must already be equal prior to the execution of the NIND rule.

In the *second case* the last chase step is an application of the NFD rule for a NFD S : W $\rightarrow$ Z $\in$ F. Thus, $\exists$ B $\in$ Z that causes the A-values of t_1 and t_2 to be equated. Furthermore, $\Sigma \vdash$ S : W $\rightarrow$ Z by induction hypothesis and thus $\Sigma \vdash$ S : W $\rightarrow$ B by the decomposition rule.

By inductive hypothesis $\Sigma \vdash$ R[VA] $\subseteq$ S[UB], where V $\subseteq$ X and U is a permutation of W, since |W| $\leq$ |X| must hold due to the fact that the W-values of the tuples in the current state of the marked relation over S must all be marked nulls prior to the application of the said NFD rule, and in addition we do not allow repeated attributes. Thus $\Sigma \vdash$ S : U $\rightarrow$ B follows by the permutation rule for NFDs and $\Sigma \vdash$ R : V $\rightarrow$ A follows by the pullback rule. The result that $\Sigma \vdash$ R : X $\rightarrow$ A now follows on using the augmentation, decomposition and permutation rules for NFDs.

(*The NIND case*): We prove that $\Sigma \models \tau$ implies $\Sigma \vdash \tau$, where τ is the NIND R[X] $\subseteq$ S[Y], by showing that the *if part* of Theorem 5.5 part (2) implies $\Sigma \vdash \tau$. We assume that it is not that case that both R = S and X = Y, otherwise $\Sigma \vdash \tau$ follows by the reflexivity rule for NINDs.

Let i $\in \{0,\ldots,2^{m-1}\}$ satisfy the condition that $\forall$ j $\in \{0,\ldots,2^{m-1}\}$, the number of chase steps required to compute MCHASE(d^i_τ, F, I) is greater than or equal to the number of chase steps required to compute MCHASE(d^j_τ, F, I).

We obtain the result by induction on the number of chase steps, say k, required to compute MCHASE(d^i_τ, F, I).

(*Basis*): If k = 1, then the only chase step executed is an application of the NIND rule for a NIND R[W] $\subseteq$ S[Z] $\in$ I, where R[X] $\subseteq$ S[Y] can be obtained from R[W] $\subseteq$ S[Z] by the projection and permutation rule for NINDs. Thus $\Sigma \vdash$ R[X] $\subseteq$ S[Y] holds.

(*Induction*): Assume the result holds when the number of chase steps required to compute MCHASE(d^i_τ, F, I) is k, where k $\geq$ 1; we then need to prove that the result holds when the number of chase steps required to compute MCHASE(d^i_τ, F, I) is k+1.

Let us consider the last chase step executed to obtain MCHASE(d^i_τ, F, I). There are two cases to consider.

In the *first case* the last chase step is an application of the NIND rule for a NIND T[W] $\subseteq$ S[Z], with Y $\subseteq$ Z. By inductive hypothesis there must exist a NIND R[V] $\subseteq$ T[W] such that $\Sigma \vdash$ R[V] $\subseteq$ T[W], with X $\subseteq$ V. It follows that R[X] $\subseteq$ S[Y] can be inferred from R[V] $\subseteq$ T[W] and T[W] $\subseteq$ S[Z] by the transitivity and projection and permutation rules for NINDs. Thus $\Sigma \vdash$ R[X] $\subseteq$ S[Y].

In the *second case* the last chase step is an application of the NFD rule for a NFD S : $Z_3 \rightarrow Z_1 \in$ F. Thus by inductive hypothesis $\Sigma \vdash$ S : $Z_3 \rightarrow Z_1$ holds. Without loss of generality we assume that Z_1 is the minimal set of attributes necessary for applying the last chase step.

By inductive hypothesis on using Lemma 5.4 we have that $\Sigma \vdash$ R[T] $\subseteq$ S[Y], with X $\approx$ T. Without loss of generality we assume that T = W_2W_1W, Y = Z_1Z_2Z and that the last chase step equates a W_1-value with a W_2-value. It follows that $|W_1| = |Z_1| = |W_2| = |Z_2| \geq 1$ must hold. Moreover, $\Sigma \vdash$ R[W_2W_1W] $\subseteq$ S[Z_1Z_2Z].

We next show that $Z_3 \subseteq Z$ also holds, assuming without loss of generality that $Z_3 \subseteq$ Y. Firstly, we can assume that $Z_3 \cap Z_1 = \emptyset$, since $\Sigma \models Z_3 \rightarrow Z_1$ if and only if $\Sigma \models Z_3 \rightarrow (Z_1 - Z_3)$. Secondly, if $Z_3 \cap Z_2 \neq \emptyset$, then it follows that in the case when $|Z_1| = 1$ (which by Theorem 5.5

can always be assumed without loss of generality) the said W_1-value and the said W_2-value can be equated in k chase steps contrary to our assumption. Thus $Z_3 \subseteq$ Z. It therefore follows that $\exists W_3 \subseteq W$ such that $|W_3| = |Z_3|$. Let $W_1W_2W_3 = < A_1, \ldots, A_q >$ and $Z_1Z_2Z_3 = < B_1, \ldots, B_q >$. We now claim that the maximal number of chase steps required to compute $\Sigma \vdash$ R[X $-A_p$] $\subseteq$ S[Y $-B_p$] is less than or equal to k, where p $\in$ {1,. . .,q}. Assume that this is not the case. Let us fix p $\in$ {1,. . .,q} and let u^i_τ be a type(R)-tuple that is defined as follows: $u^i_\tau[A_p] = unk$ and $\forall$ A $\in$ schema(R) $-A_p$, u^i_τ[A] $= t^i_\tau$[A]. We now replace the tuple $t^i_\tau \in r_i$ by the tuple u^i_τ, where r_i is the relation over R in d^i_τ.

If $A_p \in W_3$ then the NFD rule for S : $Z_3 \rightarrow Z_1$ cannot be applied, implying that MCHASE(d^i_τ, F, I) can computed in k chase steps contrary to the assumption made in the induction step. On the other hand, if $A_p \in W_1W_2$ then contrary to our assumption, Z_1 is not the minimal set of attributes necessary for applying the last chase step, since an A_p-value could not have been modified by the last chase step. This implies that the NFD rule for S : $Z_3 \rightarrow Z_1$ need not be applied and therefore MCHASE(d^i_τ, F, I) can computed in k chase steps. The claim now follows.

In a similar fashion it can be shown that the maximal number of chase steps required to compute $\Sigma \vdash$ R[W_1W_3] $\subseteq$ S[Z_1Z_3] and $\Sigma \vdash$ R[W_2W_3] $\subseteq$ S[Z_1Z_3] is less than or equal to k, since otherwise the NFD rule for S : $Z_3 \rightarrow Z_1$ cannot be applied.

In conclusion, by induction hypothesis, all the preconditions of the null collection rule hold. The result that $\Sigma \vdash$ R[X] $\subseteq$ S[Y] follows on using the null collection rule. □

The following proposition shows that we cannot, in general, put a bound on the number of NINDs used in a derivation of a new NIND that uses null collection. This means that for no natural number k $\geq$ 0 does there exist a k-ary complete axiomatization for NFDs and NINDs in the sense of [CASA84].

Proposition 5.8 *The null collection rule cannot be replaced by a finite number of rules in which the number of NINDs in the* if part *of all these rules is fixed for some natural number* $m \in \omega$.

Proof. Let **R** = {R, S}, where schema(R) = X = $W_1W_2W_3W_4$ and schema(S) = Y = $Z_1Z_2Z_3Z_4$. In addition, let $|W_1| = |Z_1| = |W_2| = |Z_2| \geq 1$, $|W_3| = |Z_3| \geq 1$, $W_1W_2W_3 = < A_1, \ldots, A_m >$ and $Z_1Z_2Z_3 = < B_1, \ldots, B_m >$.

Let Σ = F $\cup$ I, where F = {R : $W_3 \rightarrow W_1$, S : $Z_3 \rightarrow Z_1$} and I = {R[$W_2W_1W_3W_4$] $\subseteq$ S[$Z_1Z_2Z_3Z_4$]} $\cup$ {R[X$-A_i$] $\subseteq$ S[Y$-B_i$] | i $\in$ {1,. . .,m}} are sets of NFDs and NINDs over **R**. We observe that the cardinality of Σ is greater than m and that m $\geq$ 3.

It can now be verified that $\Sigma \vdash$ R[X] $\subseteq$ S[Y] on using the null collection rule. Furthermore, it can also be verified that $\forall \alpha \in \Sigma$, $\{\alpha\} \not\models$ R[X] $\subseteq$ S[Y].

We claim that if Δ is a set of at most m members of Σ, and if γ is either a NFD or a NIND over **R** such that $\Delta \models \gamma$, then $\exists \delta \in \Delta$ such that $\{\delta\} \models \gamma$. The claim follows on examining the inference rules of the axiom system we have shown to be sound and complete in Theorem 5.7 and the comment made after the definition of the null collection rule.

The result follows directly from [CASA84, Corollary 5.2], since all three sufficient conditions of this corollary are satisfied. □

It can be verified by the definitions of satisfaction of a NFD and a NIND that if we allow nonstandard NFDs we get the following interaction between NFDs and NINDs [COSM90]: {R : $\emptyset \rightarrow$ Y, R[X] $\subseteq$ R[Y]} $\models$ R : $\emptyset \rightarrow$ X. On the other hand, the relation shown in Table 15 provides a counterexample to the following interaction between NFDs and NINDs [COSM90]: $\{R : \emptyset \rightarrow Y, R[X] \subseteq R[Y]\} \models R[Y] \subseteq R[X]$.

X	Y
unk	*unk*
unk	a

Table 15: A counterexample relation

6 The implication problem for NFDs and NINDs

The implication problem is to determine the computational complexity of the logical implication $\Sigma \models \alpha$, where as before $\Sigma = F \cup I$ and α is a NFD or a NIND.

The implication problem for NFDs (i. e. when Σ consists solely of NFDs) can be solved in linear time [ATZE86] whilst by Theorem 3.3 and the result obtained in [CASA84, Theorem 3.3] the implication problem for NINDs (i. e. when Σ consists solely of NINDs) is PSPACE-complete [GARE79].

It follows that the implication problem for NFDs and NINDs is at least PSPACE-hard [GARE79]. Now, by using Corollary 4.5 and Corollary 5.6 (see the comment made after Theorem 5.1) we can establish an exponential time upper bound in the sizes of the schemas in **R** and the set Σ for the said implication problem.

The following theorem establishes an exponential time lower bound for the implication problem for NFDs and NINDs by also showing that the problem is EXPTIME-hard [GARE79]. The proof extends the technique used in [CASA84, Theorem 3.3] by reducing an alternating Turing machine working in space n [CHAN81] to the implication problem for NFDs and NINDs.

This result is interesting in view of the EXPTIME-completeness result that was established in [COSM85] for FDs and noncircular INDs holding in complete relations; the result was shown to hold even if the cardinality of the left-hand sides and right-hand sides of all the FDs is one.

Theorem 6.1 *The implication problem for NFDs and NINDs is EXPTIME-complete.*

Proof. See Appendix A.

As a special case we next show that if we do not use the null collection rule in our derivations, then the implication problem for NFDs and NINDs is in PSPACE; of course by Theorem 5.7 the axiom system without the null collection rule is no longer complete.

Theorem 6.2 *If we disallow the use of the null collection rule in dervations of NFDs and NINDs, then the implication problem for NFDs and NINDs is PSPACE-complete.*

Proof. See Appendix A.

In the process of relational database design several researchers have made the simplifying assumption that the INDs involved are noncircular [MANN86, SCIO86]. The main justification for this restriction is that the implication problem for FDs and INDs holding in complete relations was shown to be undecidable [MITC83, CHAN85]. Due to our complexity result for the implication problem of NFDs and NINDs, whenever incomplete information is supported (SQL supports the null value *unk* [DATE93]), we can relax the noncircularity restriction and consider general sets of NINDs in the process of database design.

7 Key-based NINDs and database design

Keys and foreign keys are fundamental to database design, since they capture the semantic notions of *entity* and *referential* integrity, respectively [CODD79, DATE86]. In normalization theory BCNF ensures that entity integrity is maintained [ULLM88, ATZE93] while key-based INDs ensure that referential integrity is preserved [MANN86].

Herein, we generalise a result in [MANN88] by showing that if a database schema is in BCNF and all the NINDs are key-based, then there is no interaction between the NFDs and NINDs. In contrast to [MANN88] our result implies that when designing a database, which takes incomplete information into consideration, the database designer need not assume that the NINDs are noncircular.

We now define the notions of key, superkey and Boyce-Codd normal form [ULLM88, ATZE93].

Definition 7.1 (Key and superkey) Let F be a set of NFDs over $\mathbf{R}$ and let $R \in \mathbf{R}$. A sequence of attributes $X \subseteq$ schema(R) is a *superkey* for R with respect to F (or simply a superkey for R if F is understood from context) if $F \models R : X \rightarrow$ schema(R).

A sequence of attributes $X \subseteq$ schema(R) is a *key* for R with respect to F (or simply a key for R if F is understood from context) if X is a superkey for R and there does not exist a proper subset Y of X such that Y is a superkey for R.

Definition 7.2 (Boyce-Codd normal form) A database schema $\mathbf{R}$ is in *Boyce-Codd normal form* (BCNF) with respect to a set of NFDs F over $\mathbf{R}$ (or simply in BCNF if F is understood from context) if for every nontrivial NFD $R : X \rightarrow Y \in F$, X is a superkey for R.

We next define the notion of key-based NINDs [MANN86, MANN88].

Definition 7.3 (Key-based NINDs) A NIND $R[X] \subseteq S[Y]$ over $\mathbf{R}$ is *key-based* if Y is a key for S. If $R[X] \subseteq S[Y]$ is a key-based NIND, then the set of attributes X is called a *foreign key* of R with respect to S (or simply a foreign key of R when S is understood from context). A set I of NINDs is key-based if every NIND in I is key-based.

We observe that the notion of key-based NINDs is closely related to the concept of *referential integrity* [CODD79, DATE86]. Assume that for each relation scheme in $\mathbf{R}$ we designate one of its keys as being a *primary key* [CODD79]. Then a referential integrity constraint can be defined as a key-based NIND of the form $R[X] \subseteq S[Y]$, where Y is the primary key of S.

We say that a set of NFDs F over $\mathbf{R}$ and a set of NINDs I over $\mathbf{R}$ have *no interaction* if for each NFD α over $\mathbf{R}$ and for each NIND β over $\mathbf{R}$, $F \cup I \models \alpha$ if and only if $F \models \alpha$ and $F \cup I \models \beta$ if and only if $I \models \beta$.

The following theorem shows that F and I have no interaction if $\mathbf{R}$ is in BCNF and I is key-based (cf. [MANN88]).

Theorem 7.1 *Let $\Sigma \subseteq F \cup I$ be a set of NFDs and NINDs over* **R**, *where* **R** *is in BCNF and I is key-based. Then F and I have no interaction.*

Proof. There are two cases to consider.

In the *first case* let α be a nontrivial NFD and suppose that $\Sigma \models \alpha$. We need to show that it is also the case that F $\models \alpha$.

By Theorem 5.7 it is sufficient to show that the pullback rule could not have been used in any proof of α from Σ in order to derive a nontrivial NFD which cannot be proved from F.

We use induction on the number of inference rules, k $\in \omega$, used to derive $\Sigma \vdash \alpha$.

(*Basis*): If k = 0, the result follows, since $\alpha \in$ F and therefore F $\models \alpha$ follows.

(*Induction*): Assume the result holds when the number inference rules used in order to derive $\Sigma \models \alpha$ is k, where k $\geq$ 0; we then need to prove that the result holds when the number of inference rules used to derive $\Sigma \vdash \alpha$ is k+1.

By inductive hypothesis it follows that the pullback rule must have been the last inference rule used in the proof of $\Sigma \vdash \alpha$. Let α be the NFD R : W $\rightarrow$ Z and assume without loss of generality that W is the minimal subset of schema(R) such that $\Sigma \vdash \alpha$ obtains. Thus it must be the case that $\Sigma \vdash$ R : W $\rightarrow$ Z follows from $\Sigma \vdash$ R[WZ] $\subseteq$ S[XY], with |X| = |W|, and $\Sigma \vdash$ S : X $\rightarrow$ Y by the pullback rule.

Furthermore, by inductive hypothesis we can deduce that F $\vdash$ S : X $\rightarrow$ Y and thus X is a superkey for S, since **R** is in BCNF. In addition, S : X $\rightarrow$ Y is a nontrivial NFD, since by assumption R : W $\rightarrow$ Z is a nontrivial NFD. It follows that XY is a proper superset of a key for S and thus the NIND R[WZ] $\subseteq$ S[XY] is not key-based. This leads to a contradiction, since it implies that there exists a NIND in I that is not key-based.

In the *second case* let β be a nontrivial NIND and suppose that $\Sigma \models \beta$. We need to show that it is also the case that I $\models \beta$.

By Theorem 5.7 it is sufficient to show that the null collection rule could not have been used in any proof of β from Σ in order to derive a nontrivial NIND which cannot be proved from I.

We use induction on the number of inference rules, k $\in \omega$, used to derive $\Sigma \vdash \beta$.

(*Basis*): If k = 0, the result follows, since $\beta \in$ I and therefore I $\models \beta$ follows.

(*Induction*): Assume the result holds when the number inference rules used in order to derive $\Sigma \models \beta$ is k, where k $\geq$ 0; we then need to prove that the result holds when the number of inference rules used to derive $\Sigma \vdash \beta$ is k+1.

Let β be the NIND R[X] $\subseteq$ S[Y]. It follows that the null collection rule must have been the last inference rule used in the proof of $\Sigma \vdash \beta$, with S : $Z_3 \rightarrow Z_1$ being a nontrivial NFD. In addition, Z_3 is a superkey for S, since S is in BCNF and we already have by inductive hypothesis that F $\vdash$ S : $Z_3 \rightarrow Z_1$. Again, by inductive hypothesis all the NINDs in the *if part* of the null collection rule must be key-based NINDs. In particular R[W_1W_3] $\subseteq$ S[Z_1Z_3] must be key-based. This leads to a contradiction, since Z_1Z_3 is a proper superset of a key for S. □

As a consequence of Theorem 7.1 we suggest that a good design principle is to obtain a BCNF database schema together with a set of key-based NINDs. In such a case the keys and foreign keys provide the user with the full semantics of the database schema. It is important to note that we have *not* assumed that the set I of NINDs is noncircular as is the case in [MANN86].

We conclude this section by explaining the semantics of an example taken from [SCIO86, page 149], which was used therein to show that circular sets of INDs give rise to unclear semantics.

Let BOSS be a relation schema with schema(BOSS) = {EMP, MGR} and let I = {BOSS[MGR] $\subseteq$ BOSS[EMP]} be a circular set of NINDs. The NIND in I states that every manager is also an employee. It follows that if r is a relation over BOSS and the tuple <John, Jack> is in r, then there must also exist a tuple of the form <Jack *unk*> in r. This does not cause any problems in the presence of incomplete information, since we may not know who Jack's boss is. On the other hand, if we only deal with complete relations then we get into a circular argument implying that r is an infinite relation (recall that we only allow finite relations) unless we add the tuple $< Jack, Jack >$ to r implying that Jack manages himself [SCIO86]. In both cases the semantics of the circular IND in I are unclear. By contrast, in the presence of nulls circular NINDs do not give rise to any semantic problems.

8 Concluding Remarks

We have introduced NINDs and studied their axiomatization and their implication problem in conjunction with NFDs. By employing a new inference rule, i. e. null collection, together with the pullback inference rule and the respective axiom systems for NFDs and NINDs, we have obtained a sound and complete axiom system for NFDs and NINDs. In order to establish our results we have made extensive use of the chase procedure as a theorem proving tool. We have also obtained the surprising result that the implication problem for NFDs and NINDs is decidable and EXPTIME-complete, thus allowing us to relax the restriction in database design that NINDs be noncircular. Finally, our results suggest that a good design principle is to obtain a BCNF database schema together with a set of key-based NINDs, since in this case NFDs and NINDs have no interaction.

Another important class of NINDs is the class of *unary NINDs*, that is NINDs of the form R[A] $\subseteq$ S[B], where A $\in$ schema(R) and B $\in$ schema(S) [COSM90]. In a similar fashion to the proof of Theorem 7.1, it can be shown that if F is a set of standard NFDs and I is a set of unary NINDs, then F and I have no interaction. Thus, on using the results in [COSM90] and the fact that the implication problem for NFDs can be solved in linear time, we deduce that the implication problem for NFDs and unary NINDs can also be solved in linear time.

A Appendix

Proof of Theorem 4.4. Let $\mathbf{R} = \{R_1, \ldots, R_n\}$, where $\forall$ i $\in$ {1,...,n}, schema(R_i) = $< A_1, \ldots, A_n >$. Also, let d = $< r_1, \ldots, r_n >$, where $r_1 = \{< a_1, \ldots, a_n >\}$, with $a_1, \ldots, a_n$ being n distinct nonnull values, and such that $\forall$ i $\in$ {2,...,n}, $r_i = \emptyset$. Also, let |CHASE(d, $\emptyset$, I)| = $\sum_{i=1}^{n} |r_i|$, where $r_i \in$ CHASE(d, $\emptyset$, I) is the relation over $R_i \in \mathbf{R}$. We exhibit a set of NINDs I over $\mathbf{R}$ such that |I| = 2(n$-$1) and |CHASE(d, $\emptyset$, I)| = $2^n - 1$.

In the following we let ID_i denote the NIND $R_i[A_1, \ldots, A_n] \subseteq R_{i+1}[A_1, \ldots, A_n]$, where i $\in$ {1,...,n$-$1}.

Informally, for each i $\in$ {1,...,n$-$1}, the set of NINDs I over $\mathbf{R}$ includes the NIND, ID_i, together with a NIND that swaps the ith attribute with the (i+1)th attribute in schema(R_i). Thus, formally I = $\cup$ {I_i | i $\in$ {1,...,n$-$1}}, where $I_i = \{ID_i\} \cup \{R_i[A_1, \ldots, A_{i+1}, A_i, \ldots, A_n] \subseteq R_{i+1}[A_1, \ldots, A_i, A_{i+1}, \ldots, A_n]\}$.

We obtain the result by induction on i $\in \{1,\ldots,n\}$ showing that for all i the cardinality of the relation $r_i \in$ CHASE(d, $\emptyset$, I) over $R_i \in \mathbf{R}$ is 2^{i-1}. This yields the required result, namely $|$CHASE(d, $\emptyset$, I)$| = 2^n - 1$.

(*Basis*): If i = 1, then trivially $|r_1| = 1$.

(*Induction*): Assume that the result holds for i ≥ 1. We need to prove that the result holds for i+1. By inductive hypothesis $|r_i| = 2^{i-1}$.

Furthermore, $\pi_{A_{i+1}}(r_i) = \{< a_{i+1} >\}$ and $\pi_{A_i}(r_i) \cap \pi_{A_{i+1}}(r_i) = \emptyset$ by the definition of I_i. Therefore r_{i+1} will have twice as many tuples as r_i due to I_{i+1}. $\square$

Proof of Theorem 6.1. By the discussion prior to the statement of this theorem it remains to establish that the implication problem is EXPTIME-hard. In order to prove this result we construct an alternating Turing machine (ATM) [CHAN81] working in space n, where n is the sum of the size of the input multiplied by the number of work tapes and the input tape; in the sequel we will view the collection of these tapes as a single tape of size n. For simplicity we allow the symbols of the input tape to be overwritten with blanks prior to the machine halting if it accepts its input.

Let Φ and Ψ be the alphabet and the set of states, respectively, of an ATM, say M. We let b $\in \Phi$ denote the blank symbol and assume for simplicity that there is a distinct distinguished start state, denoted by Start, and a distinct distinguished accepting state, denoted by Accept, such that the machine accepts its input if and only if it enters the accepting state, Accept. Without loss of generality we will assume that prior to entering the Accept state the contents of the input tape will be overwritten with blanks and the machine's head will be scanning the first cell thereof.

Let R be a relation schema whose attribute sequence, schema(R), includes the set of states Ψ, the set of natural numbers $\{1,\ldots,n\}$ and the set of ordered pairs $\{1,\ldots,n\} \times \Phi$.

Furthermore, let m = $|\Psi||\Phi|$n and assume that the sequences of additional attributes, $< A_1, \ldots, A_m >$, $< B_1, \ldots, B_m >$, $< D_1, \ldots, D_m >$, $< E_1, \ldots, E_m >$ together with the single attribute C are also contained in schema(R). It is evident that each attribute A_i, B_i, D_i and E_i corresponds to a triple representing a unique state in Ψ, a unique symbol in Φ and a unique cell position on the tape of bounded space n.

We can describe a configuration of M by the sequence of attributes:

$$< St_i, Pos_i, (1, Sym_1), \ldots (n, Sym_n) >,$$

where St_i is the current state of M, $Pos_i \in \{1,\ldots,n\}$ is the position of the cell on the said tape and $< (1, Sym_1), \ldots, (n, Sym_n) >$ describes the current contents of the tape. The initial configuration, designated by γ, is thus

$$< Start, 1, (1, Inp_1), \ldots (n, Inp_n) >,$$

where x = $< Inp_1, \ldots, Inp_n >$ denotes the input padded with blanks. Correspondingly, the accepting configuration is

$$< Accept, 1, (1, b), \ldots, (n, b) > .$$

Recall that the set of states Ψ of the ATM M contains two types of state: *universal* states and *existential* states. Informally, given an input, say x, we say that M accepts x if when M enters an existential state one of its successor states leads to the acceptance of x and when M enters a universal state all of its successor states lead to the acceptance of x (for the formal definition see [CHAN81]).

We will make several assumptions about the ATM, M, which do not restrict its expressive power. Firstly, if St_i has only one successor state then we always consider it to be an existential state. Secondly, if St_i has no successor states then we will assume, without loss of generality, that it is the accepting state, Accept. Finally, we will assume that the accepting state, Accept, can only be entered from an existential state.

Now, recall that each transition of an ATM, M, specifies that if M is in state St_1 scanning symbol Sym_1 in position i $\in$ {1,. . .,n} of the tape, then M enters state St_2 overwrites Sym_1 with symbol Sym_2 and then moves its head to any cell which is either to the left or to the right of the current position (the head may also remain stationary). Given such a transition we say that St_2 is the *successor* of St_1. For simplicity we will assume, without loss of generality, that there is no transition from a state St_i to the Start state.

Prior to continuing the proof we transform the ATM, M, into an equivalent ATM, M', (that is M' accepts an input, say x, if and only if M accepts x) by adding to M, for each universal state St^{old} of M that has at least one successor which is an existential state, an additional universal state St^{new} and several additional existential states, as follows.

Let St_i^{old}, which is an existential state, be a successor of St^{old}, and let St_j^{old} be the successor of St_i^{old}. Then we add St^{new} as a successor of St_i^{old} with the following semantics: St^{new} has two successors, St_1^{new} and St_2^{new}, both of which are existential states. St_1^{new} does nothing, that is, it overwrites the symbol the head is currently scanning with an identical symbol and moves the machine's head zero cells; it has one successor St_j^{old}. On the other hand, St_2^{new} overwrites the symbol the head is currently scanning with an identical symbol and then moves the machine's head to the end of the tape. The single successor of St_2^{new} overwrites the symbol the head is currently scanning with b and moves the machine's head one cell to the left. This process is repeated at most n$-$1 times at which point the machine accepts its input. (At the last such move the symbol at the beginning of the tape is overwritten with b.)

It can easily be verified that given an input x, M' accepts x if and only if M accepts x. For convenience in the sequel we will refer to the transformed ATM, M', simply as M.

We will now construct a set of NINDs that describe the transitions of M. There are two cases to consider for any transition from state St_1 to state St_2.

In the *first case* St_1 is an existential state and the transition of M can be described by a NIND of the form:

$$R[St_1, i, (i, Sym_i), Rest_i, Add_1] \subseteq R[St_2, j, (i, Sym_j), Rest_i, Add_2], \tag{2}$$

where i $\in$ {1,. . .,n}, j is calculated according to how many cells the machine's head moves to the left or to the right, and $Rest_i$ is an arbitrary ordering of ({1,. . .,n} $-$ {i}) $\times\Phi$. Furthermore, $Add_1 = < A_1, \ldots, A_m, B_1, \ldots, B_m, C >$, and if $St_2 \neq$ Accept then $Add_2 = Add_1$, otherwise (when $St_2 =$ Accept) $Add_2 = < D_1, \ldots, D_m, E_1, \ldots, E_m, C >$.

The sequence of pairs $Rest_i$ represents all possible cell contents other than the cell at position i (cf. [CASA84, Theorem 3.3]). We also assume that the NINDs only record *legal* transitions, i. e. transitions that do not move off the left end or off the right end of the tape.

In the *second case* St_1 is a universal state with k successors, $< St_1^1, \ldots, St_k^1 >$, with $2 \leq$ k $<$ m. Let $X_k = < A_1^1, \ldots, A_k^1 >$ be the sequence of A_i's corresponding to the k successors of St_1 according to the state, symbol and position induced by the transition, and let $Y_k = < B_1^1, \ldots, B_k^1 >$ be the sequence of B_i's corresponding to the k successors of St_1 according to the state, symbol and position induced by the transition.

For each St_p^1, where p $\in$ {1,...,k}, the transition of M can be described by two NINDs. The first NIND is of the form:

$$R[St_1, i, (i, Sym_i), Rest_i, Add_1^{A_p}] \subseteq R[St_p^1, j, (i, Sym_j), Rest_i, Add_2^{A_p}], \tag{3}$$

where i, j and $Rest_i$ are the same as described above. Furthermore, we let $Add_1^{A_p} = (< A_1, \ldots, A_m > -A_p^1) < B_1, \ldots, B_m, C >$ and $Add_2^{A_p} = Add_1^{A_p}$ (recall that since St_1 is a universal state then by assumption $St_2 \neq$ Accept).

The second NIND is of the form:

$$R[St_1, i, (i, Sym_i), Rest_i, Add_1^{B_p}] \subseteq R[St_p^1, j, (i, Sym_j), Rest_i, Add_2^{B_p}], \tag{4}$$

where i, j and $Rest_i$ are again the same as above, and for each p $\in$ {1,...,k}, $Add_1^{B_p} = < A_1, \ldots, A_m > (< B_1, \ldots, B_m, C > -B_p^1)$ and $Add_2^{B_p} = Add_1^{B_p}$.

In addition, we construct two additional NINDs. The first NIND is of the form:

$$\begin{array}{c} R[St_1, i, (i, Sym_i), Rest_i, A_1, \ldots, A_m, B_1, \ldots, B_m] \subseteq \\ R[Accept, 1, (i, b), Rest_i, D_1, \ldots, D_m, E_1, \ldots, E_m]. \end{array} \tag{5}$$

The second NIND is of the form:

$$\begin{array}{c} R[St_1, i, (i, Sym_i), Rest_i, A_1, \ldots, A_g, Y_k, A_{g+k+1}, \ldots, A_m, B_1, \ldots, B_g, X_k, B_{g+k+1}, \ldots, B_m, C] \subseteq \\ R[Accept, 1, (i, b), Rest_i, D_1, \ldots, D_m, E_1, \ldots, E_m, C], \end{array} \tag{6}$$

where $< A_1, \ldots, A_g, A_{g+k+1}, \ldots, A_m > = < A_1, \ldots, A_m > -X_k$ and $< B_1, \ldots, B_g, B_{g+k+1}, \ldots, B_m > = < B_1, \ldots, B_m > -Y_k$. (Each $A_p^1 \in X_k$ is swapped with the corresponding $B_p^1 \in Y_k$.)

Let Σ^M be the set of all NINDs that describe ATM transitions of M together with the NFD, R : C $\rightarrow < D_1, \ldots, D_m >$. We observe that the construction of Σ^M requires time polynomial in the size of the description of M.

By [CHAN81, Corollary 3.5] alternating polynomial space is equal to EXPTIME. Therefore in order to conclude the proof we show that ATM, M, accepts an input x = $< Inp_1, \ldots, Inp_n >$ if and only if $\Sigma^M \models \alpha$, where α is the NIND:

$$\begin{array}{c} R[Start, 1, (1, Inp_1), \ldots, (n, Inp_n), A_1, \ldots, A_m, B_1, \ldots, B_m, C] \subseteq \\ R[Accept, 1, (1, b), \ldots, (n, b), D_1, \ldots, D_m, E_1, \ldots, E_m, C]. \end{array}$$

M accepts x if and only if we can construct a *directed acyclic graph* (DAG) [BUCK90] whose nodes are labeled by configurations of M and such that an arc (v_1, v_2) denotes the fact that the configuration that labels node v_2 results from a transition emanating from the configuration that labels node v_1. The configuration γ labels node, say v_0. Let us denote such a DAG with respect to M and x by DAG(M, x). It can easily be verified that DAG(M, x) has a single *root* labeled by the initial configuration γ and a single *sink* labeled by the accepting configuration.

Assume that M accepts x. We prove that $\Sigma^M \models \alpha$ by an induction on the length, say q, of the longest path in DAG(M, x) from the root to its sink.

(*Basis*): If q = 1, then the result follows, since α can be obtained from the NIND given by (2) by using the projection and permutation rule for NINDs. (Note that q = 0 is not possible, since Accept is distinct from Start.)

(*Induction*): Assume that the result holds when the length of the longest path in DAG(M, x) from the root to its sink is q, where q $\geq$ 1; we then need to prove that the result holds when the length of the longest path in DAG(M, x) from the root to its sink is q+1. There are two cases to consider.

In the *first case* the start state is an existential state. Thus, by (2), Σ^M includes the NIND:

$$R[Start, 1, (1, Inp_1), Rest_1, A_1, \ldots, A_m, B_1, \ldots, B_m, C] \subseteq \\ R[St_2, j, (1, Sym_1), Rest_1, A_1, \ldots, A_m, B_1, \ldots, B_m, C].$$

Now, on using the projection and permutation rule for NINDs we can derive the NIND:

$$R[Start, 1, (1, Inp_1), \ldots, (n, Inp_n), A_1, \ldots, A_m, B_1, \ldots, B_m, C] \subseteq \\ R[St_2, j, (1, Sym_1), \ldots, (n, Inp_n), A_1, \ldots, A_m, B_1, \ldots, B_m, C],$$

corresponding to a transition of M from Start. Furthermore, by inductive hypothesis, with respect to the longest path in DAG(M, x) $-v_0$ [BUCK90], Σ^M logically implies the NIND:

$$R[St_2, j, (1, Sym_1), \ldots, (n, Inp_n), A_1, \ldots, A_m, B_1, \ldots, B_m, C] \subseteq \\ R[Accept, 1, (1, b), \ldots, (n, b), D_1, \ldots, D_m, E_1, \ldots, E_m, C].$$

The result that $\Sigma^M \models \alpha$ follows on using the transitivity rule for NINDs.

In the *second case* the start state is a universal state. Thus, by (3), on using the projection and permutation rule for NINDs, for each p $\in$ {1,. . .,k}, Σ^M includes the NIND:

$$R[Start, 1, (1, Inp_1), \ldots, (n, Inp_n), Add_1^{A_p}] \subseteq R[St_p^1, j, (1, Sym_1), \ldots (n, Inp_n), Add_2^{A_p}], \quad (7)$$

corresponding to a transition of M from Start.

Similarly, by (4), on using the projection and permutation rule for NINDs, for each p $\in$ {1,. . .,k}, Σ^M includes the NIND:

$$R[Start, 1, (1, Inp_1), \ldots, (n, Inp_n), Add_1^{B_p}] \subseteq R[St_p^1, j, (1, Sym_1), \ldots, (n, Inp_n), Add_2^{B_p}], \quad (8)$$

corresponding to a transition of M from Start.

Also, by (5), on using the projection and permutation rule for NINDs, Σ^M includes the NIND:

$$R[Start, 1, (1, Inp_1), \ldots, (n, Inp_n), A_1, \ldots, A_m, B_1, \ldots, B_m] \subseteq \\ R[Accept, 1, (1, b), \ldots, (n, b), D_1, \ldots, D_m, E_1, \ldots, E_m].$$

Finally, by (6), on using the projection and permutation rule for NINDs, Σ^M includes the NIND:

$$R[Start, 1, (1, Inp_1), \ldots, (n, Inp_n), \widehat{Add}_1] \subseteq R[Accept, 1, (1, b), \ldots, (n, b), \widehat{Add}_2],$$

where in this case $\widehat{Add}_1 =< A_1, \ldots, A_g, Y_k, A_{g+k+1}, \ldots, A_m, B_1, \ldots, B_g, X_k, B_{g+k+1}, \ldots, B_m, C >$ and $\widehat{Add}_2 =< D_1, \ldots, D_m, E_1, \ldots, E_m, C >$.

Now, by inductive hypothesis, with respect to the longest path in DAG(M, x) $-v_0$, it follows that, for each p $\in$ {1,. . .,k}, Σ^M logically implies the NIND:

$$R[St_p^1, j, (1, Sym_1), \ldots (n, Inp_n), Add_2^{A_p}] \subseteq R[Accept, 1, (1, b), \ldots, (n, b), Add_2^{D_p}], \quad (9)$$

where $Add_2^{D_p} = (< D_1, \ldots, D_m > - D_p^1) < E_1, \ldots, E_m, C >$ and D_p^1 is the attribute in the sequence of D_i's corresponding to A_p^1.

Thus, on using (7) and (9) and the transitivity rule for NINDs, for each p $\in$ {1,...,k}, Σ^M logically implies the NIND:

$$R[Start, 1, (1, Inp_1), \ldots, (n, Inp_n), Add_1^{A_p}] \subseteq R[Accept, 1, (1, b), \ldots, (n, b), Add_2^{D_p}].$$

In addition, by inductive hypothesis it follows that, for each p $\in$ {1,...,k}, Σ^M logically implies the NIND:

$$R[St_p^1, j, (1, Sym_1), \ldots (n, Inp_n), Add_1^{B_p}] \subseteq R[Accept, 1, (1, b), \ldots, (n, b), Add_2^{B_p}], \qquad (10)$$

where $Add_2^{B_p} = < D_1, \ldots, D_m > (< E_1, \ldots, E_m, C > - < E_p^1 >)$ and E_p^1 is the attribute in the sequence of E_i's corresponding to B_p^1.

Thus on using (8) and (10) and the transitivity rule for NINDs, for each p $\in$ {1,...,k}, Σ^M logically implies the NIND:

$$R[Start, 1, (1, Inp_1), \ldots, (n, Inp_n), Add_1^{B_p}] \subseteq R[Accept, 1, (1, b), \ldots, (n, b), Add_2^{B_p}].$$

Thus all the preconditions of the null collection rule hold. In particular, the subsequence of attributes X_k corresponds to W_1, the subsequence of attributes Y_k corresponds to W_2, the single attribute C corresponds to W_3 and the remaining sequence of attributes $< A_1, \ldots, A_m, B_1, \ldots, B_m > - (X_k Y_k < C >)$ concatenated with $< Start, 1, (1, Inp_1), \ldots, (n, Inp_n) >$ corresponds to W_4. Similarly, the corresponding subsequences of attributes contained in the sequence $< D_1, \ldots, D_m, E_1, \ldots, E_m, C >$ concatenated with $< Accept, 1, (1, b), \ldots, (n, b) >$ and having the same subscripts as the subsequences X_k and Y_k, correspond to Z_1, Z_2, Z_3 and Z_4, respectively. It follows that $\Sigma^M \vdash \alpha$ on using the null collection rule and thus by Theorem 5.7 $\Sigma^M \models \alpha$ as required.

Conversely, assume that $\Sigma^M \models \alpha$. Thus by Theorem 5.7 $\Sigma^M \vdash \alpha$ on using our sound and complete axiom system for NFDs and NINDs. We prove that DAG(M, x) can be constructed (that is, M accepts x) by induction on the minimal number of times, say q, the null collection rule is used in order to construct a proof of α from Σ^M. We assume without loss of generality that the length of a proof of α from Σ^M is minimal, i. e. no redundant NFDs and NINDs are included in the proof.

(*Basis*): Assume that q = 0. We note that if the null collection rule was not used in the proof of α from Σ^M, then the pullback rule was also not used in the proof, due to the fact that α is a NIND. Thus α is derived by using only the inference rules for NINDs.

We claim that in this case *all* NINDs in the proof of α from Σ^M are of the form given by (2), i. e. R[$St_1, \ldots, Add_1$] $\subseteq$ R[$St_2, \ldots, Add_2$], where St_1 and St_2 are existential states.

By assumption no redundant NINDs appear in the proof of α from Σ^M. Thus, the claim follows on inspecting our axiom system, since all of the NINDs in the proof of α from Σ^M must be of the form, R[$St_1, \ldots, Add_1$] $\subseteq$ R[$St_2, \ldots, Add_2$], where St_1 and St_2 are existential states and either $Add_1 = Add_2$, with Add_1 being a permutation of $< A_1, \ldots, A_m, B_1, \ldots, B_m, C >$, or $Add_1 =< A_1, \ldots, A_m, B_1, \ldots, B_m, C >$ and $Add_2 =< D_1, \ldots, D_m, E_1, \ldots, E_m, C >$ (see (2)); otherwise null collection would have to be used contradicting the fact that α is derived using only the inference rules for NINDs.

The result now follows by [CASA84, Corollary 3.2] noting that in this case DAG(M, x) consists of a single path such that all of the nodes in the path are labeled by configurations with existential states.

(*Induction*): Assume that the result holds when the minimal number of times the null collection rule is used to construct a proof of α from Σ^M is q, where q ≥ 0; we then need to prove that the result holds when the minimal number of times the null collection rule is used to construct a proof of α from Σ^M is q+1.

Let us consider the last time (i. e. the (q+1)th time) the null collection rule is used in the proof of α from Σ^M, assuming that the null collection rule is used to derive the NIND:

$$\begin{array}{c} R[St_1, i, (1, Sym_1), \ldots, (n, Sym_n), A_1, \ldots, A_m, B_1, \ldots, B_m, C] \subseteq \\ R[Accept, 1, (1, b), \ldots, (n, b), D_1, \ldots, D_m, E_1, \ldots, E_m, C]. \end{array} \tag{11}$$

It must be the case that the state St_1 in the NIND given by (11) is a universal state, since otherwise the result would follow by inductive hypothesis.

Now, by inductive hypothesis, we can construct a DAG, G, such that G can be extended to DAG(M, x) by adding to it a subgraph [BUCK90] whose root, say v_0, is labeled by the configuration $< St_1, i, (1, Sym_1), \ldots, (n, Sym_n) >$.

It therefore remains to show that we can extend G in order to obtain DAG(M, x) in such a way that all the paths starting from v_0 end at a single sink which is labeled by the accepting configuration.

We conclude the proof by tracing the NINDs that are necessarily used during the last time the null collection rule is used in order to obtain $\Sigma^M \vdash \alpha$.

We note that due to our transformation of M, prior to the construction of Σ^M, each set of NINDs described by (3), (4), (5) and (6) necessarily causes an instance of the null collection rule to be used in order to obtain $\Sigma^M \vdash \alpha$.

In particular, if q = 0, then the transformation does not alter the original ATM and the basis step obtains, so we can assume that q > 0. Now, if the previous universal state is not a new state that was added by the transformation, then St_1 must be its own successor. This implies that the null collection rule is necessary, otherwise the NINDs given by (3) and (4) would be trivial implying that the proof of α from Σ^M is not minimal. Therefore, we can assume that the previous universal state is a new universal state, St^{new}, which was added to the original ATM due to the said transformation.

We recall that St^{new} is the successor of an existential state St_i^{old} and moreover that St^{new} has two successors St_1^{new} and St_2^{new}, both of which are existential states. Thus, $St_i^{old} \neq St_1^{new}$ and therefore $\forall$ p $\in$ {1,. . .,k}, $A_p^1 \neq A_q$, where A_q is the attribute corresponding to St_1^{new} according to the state, symbol and position induced by the transition. It follows that the null collection rule is necessary in order to apply the transitivity rule for NINDs further on in the proof of α from Σ^M.

Now, for each p $\in$ {1,. . .,k}, where k is the number of successors of St_1, the proof of α from Σ^M must include two NINDs (see (3)). The first NIND is:

$$R[St_1, i, (1, Sym_1), \ldots, (n, Sym_n), Add_1^{A_p}] \subseteq R[St_p^1, i, (1, Sym_1), \ldots, (n, Sym_n), Add_1^{A_p}].$$

Since this is the last time the null collection rule is used, it follows that $\forall$ p $\in$ {1,. . .,k}, St_p^1 is an existential state and inductively that all the successors of St_p^1 are also existential states. Thus, the second NIND is:

$$R[St_p^1, i, (1, Sym_1), \ldots, (n, Sym_n), Add_1^{A_p}] \subseteq R[Accept, 1, (1, b), \ldots, (n, b), Add_2^{A_p}].$$

Since this is the last time the null collection rule is used, then by a similar argument to that made in the basis step the result that G can be extended to DAG(M, x) follows.

We note that in order to use the null collection rule we must first apply the transitivity rule for NINDs so that, for each p $\in$ {1,. . .,k}, the proof of α from Σ^M contains the NIND:

$$R[St_1, i, (1, Sym_1), \ldots, (n, Sym_n), Add_1^{A_p}] \subseteq R[Accept, 1, (1, b), \ldots, (n, b), Add_2^{A_p}].$$

We omit to mention the other NINDs and the NFD that are used to derive the NIND given in (11), since they are similar to those demonstrated in the earlier part of the proof. □

Proof of Theorem 6.2. Let Σ = F $\cup$ I be a set of NFDs and NINDs over **R**, and let α be the NFD R : W $\rightarrow$ B, where W $\subseteq$ schema(R).

Now, if $\Sigma \vdash \alpha$ can be derived *without* using the null collection rule then we write $\Sigma \vdash' \alpha$. We *claim* that $\Sigma \vdash' \alpha$ can be computed in polynomial space in the size of Σ, assuming without loss of generality that the size of Σ is at least as large as the size of **R**. On using the decomposition and union rules for NFDs, this claim implies that $\Sigma \vdash'$ R : W $\rightarrow$ Z, where Z $\subseteq$ schema(R), is also in PSPACE. Consider the ensuing algorithm.

Algorithm 2 (derive_NFD(Σ, R : W $\rightarrow$ B))

```
begin
  if B ⊆ W then
    return YES;
  end if
  for all nonempty subsets V ⊆ W do
    for all S ∈ R do
      for all subsets X ⊆ schema(S) such that |X| = |V| do
        for all A ∈ schema(S) − X do
          if F ⊢ S: X → A and I ⊢ R[V<B>] ⊆ S[X<A>] then
            return YES;
          end if
        end for
      end for
    end for
  end for
  return NO;
end.
```

Although there is an exponential number of subsets X $\subseteq$ schema(S), for the purpose of Algorithm 2 we need only generate the subsets one at a time and this can be done in polynomial space in the size of schema(S) by using a binary number comprising |W| bits.

The correctness of the algorithm relies on the claim that $\Sigma \vdash'$ R : W $\rightarrow$ B if and only if there exists a chain R[VB] $\subseteq S_1[X_1A_1] \subseteq \ldots \subseteq S_q[X_qA_q]$, q $\geq$ 1, such that I $\vdash$ R[VB] $\subseteq S_1[X_1A_1]$, $\forall$ i $\in$ {1,. . .,q−1}, I $\vdash S_i[X_iA_i] \subseteq S_{i+1}[X_{i+1}A_{i+1}]$ and F $\vdash S_q : X_q \rightarrow A_q$, with V $\subseteq$ W.

The *if part* of the above claim follows immediately on using the transitivity rule for NINDs, the pullback rule and the augmentation rule for NFDs. The *only if part* follows from the fact that if F is a set of NFDs over **R**, then F $\vdash$ R : W $\rightarrow$ B, where B $\not\subseteq$ W, if and only if

$\exists$ R : V $\rightarrow$ B $\in$ F such that V $\subseteq$ W. We note that this claim is not true in the absence of null values, since Armstrong's axiom system includes the transitivity rule for FDs. Therefore the for loop beginning at line 5 is sufficient in order to check whether the derivation holds. $\Box$

Acknowlegements. The authors would like to thank the referees for their constructive comments.

References

[ATZE86] P. Atzeni and N.M. Morfuni, Functional dependencies and constraints on null values in database relations. *Information and Control* **70**, (1986), 1-31.

[ATZE93] P. Atzeni and V. De Antonellis, *Relational Database Theory.* Redwood City, Ca., Benjamin/Cummings, 1993.

[BUCK90] F. Buckley and F. Harary, *Distance in Graphs.* Redwood City, Ca., Addison-Wesley, 1990.

[CASA84] M.A. Casanova, R. Fagin and C.H. Papadimitriou, Inclusion dependencies and their interaction with functional dependencies. *Journal of Computer and System Sciences* **28**, (1984), 29-59.

[CHAN81] A.K. Chandra, D.C. Kozen and L.J. Stockmeyer, Alternation. *Journal of the ACM* **28**, (1981), 114-133.

[CHAN85] A.K. Chandra and M.Y. Vardi, The implication problem for functional and inclusion dependencies is undecidable. *SIAM Journal on Computing* **14**, (1985), 671-677.

[CODD79] E.F. Codd, Extending the database relational model to capture more meaning. *ACM Transactions on Database Systems* **4**, (1979), 379-434.

[COSM85] S.S. Cosmadakis and P.C. Kanellakis, Equational theories and database constraints. In: *Proceedings of the ACM Symposium on Theory of Computing*, pp. 273-284, 1983.

[COSM90] S.S. Cosmadakis, P.C. Kanellakis and M.Y. Vardi, Polynomial-time implication problems for unary inclusion dependencies. *Journal of the ACM* **37**, (1990), 15-46.

[DATE86] C.J. Date, Referential integrity. In: *Relational Database: Selected Writings.* Addison-Wesley, Reading, Ma., pp. 41-63, 1986.

[DATE93] C.J. Date and H. Darwen, *A Guide to the SQL Standard*, third edition. Reading, Ma., Addison-Wesley, 1993.

[DAVE90] B.A. Davey and H.A. Priestly, *Introduction to Lattices and Order.* Cambridge, U.K., Cambridge University Press, 1990.

[GARE79] M.R. Garey and D.S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-completeness.* New York, Freeman, 1979.

[HONE82] P. Honeyman, Testing satisfaction of functional dependencies. *Journal of the ACM* **29**, (1982), 668-677.

[JOHN84] D.S. Johnson and A. Klug, Testing containment of conjunctive queries under functional and inclusion dependencies. *Journal of Computer and System Sciences* **28**, (1984), 167-189.

[LEVE92] M. Levene and G. Loizou, Inferring null join dependencies in relational databases. *BIT* **32**, (1992), 413-429.

[LEVE96] M. Levene and G. Loizou, The additivity problem for functional dependencies in incomplete relations. *Acta Informatica*, in press, 1996.

[LIEN82] Y.E. Lien, On the equivalence of database models. *Journal of the ACM* **29**, (1982), 333-362.

[MAIE79] D. Maier, A.O. Mendelzon and Y. Sagiv, Testing implications of data dependencies. *ACM Transactions on Database Systems* **4**, (1979), 455-469.

[MANN86] H. Mannila and K-J. Räihä, Inclusion dependencies in database design. In: *Proceedings of the IEEE Conference on Data Engineering*, Los Angeles, Ca., 713-718, 1986.

[MANN88] H. Mannila and K.-J. Raiha, Generating Armstrong databases for sets of functional and inclusion dependencies. Research Report A-1988-7, University of Tampere, Department of Computer Science, Tampere, Finland.

[MITC83] J.C. Mitchell, The implication problem for functional and inclusion dependencies. *Information and Control* **56**, (1983), 154-173.

[SCIO86] E. Sciore, Comparing the universal instance and relational data models. In: *Advances in Computing Research Vol. 3*, Eds. P.C. Kanellakis and F. Preparata, JAI Press, Greenwich, 1986, pp. 139-162.

[ULLM88] J.D. Ullman, *Principles of Database and Knowledge-Base Systems, Vol. I.* Rockville, Md., Computer Science Press, 1988.