Detecting When UNNEST After a Sequence of NESTs and UNNESTs

can Always be Reversed with a Single NEST'

Hsiu-Hsen Yao and Guiltelk Ozsoyoglu

Department of Computer Engineering and Science
Case Western Reserve University
Cleveland, OH 44106

ABSTRACT

Algebraic database languages for nested relations use two operators, NEST and UNNEST, which change the nesting levels of relations. Consider a first-normal form relation scheme R and a sequence SEQ of NEST and UNNEST operators on R with the last operator being an UNNEST on attribute Y. We say that SEQ is single-nest-reversible if, for all instances r of R, we obtain the same relation instance we had before UNNEST if we nest SEQ(r) on attribute Y.

Single-nest-reversibility is a semantic integrity constraint for some database environments, and needs to be enforced. Also, the sequence SEQ induces some functional dependencies in relations, and the axiomatization of these dependencies directly use the single-nest-reversibility condition. In this paper, we give

(1) a necessary and sufficient condition that decides as to whether SEQ is single-nest-reversible or not;
(2) the algorithm TEST to check the condition in (1) in O(n^2) time where n is the length of SEQ; and
(3) sound and complete axioms for functional dependencies in one-level nested relations.

1. Introduction

Nested relations, also called Nonfirst-Normal-Form (N1NF) relations, of databases are relations containing relations. This paper considers a restricted class, namely, one-level nested relations [OzOM87] that have only simple-valued (atomic) and set-valued attributes. That is, arbitrary relations inside relations are not considered. We think that understanding the properties of one-level nested relations is an important step in understanding the properties of arbitrarily nested relations.

NEST and UNNEST are two relational algebra operators that change the nesting levels of nested relations [Maki77, RoKS88]. NEST groups an atomic (i.e., simple-valued) attribute into a set-valued attribute, and UNNEST decomposes a set-valued attribute into an atomic attribute. We append * to attribute Y in relation NEST(Y) to represent the fact that Y is now changed into a one-level nested attribute, i.e., Y* becomes a set-valued attribute.

We give an example.

Example 1. Consider R [Parent,Child] describing the binary relationship "Parent has Child", and a sequence SEQ(r) = NEST parent. NEST child. UNNEST parent. UNNEST child. NEST child(r). Relation instances r_0, r_3 and r_5 are given in figure 1.

![Figure 1. Examples of NEST and UNNEST](image-url)

The research in this paper is supported by the National Science Foundation under the grant DCR-8605554.
Nested relations may be views defined by different users for set-oriented updates or for flat relation restructuring. Therefore, for view management, it is important to understand the properties of SEQ.

It has been known [Jae82] that, given a set-valued attribute \( Y^* \) of a relation scheme \( R \), there may exist an instance \( r \) of \( R \) such that \( \text{NEST}_Y(UNNEST_Y^{-1}(r)) \neq r \). In such a case, \( UNNEST_Y^{-1} \) can be "reversed" only for the relation instance \( r \). We now give the definition of single-nest-irreversibility. Let \( R_0 \) be a flat (first-normal-form) relation scheme. Consider a NEST and UNNEST operator sequence \( \text{SEQ}(R_0) \) such that the last (say, the \( n^{th} \)) operator of \( \text{SEQ} \) is \( UNNEST_Y^{-1} \) getting as an input the relation scheme \( R_{n-1} \) and producing \( R_n \). We say that the sequence \( \text{SEQ}(R_0) \) is single-nest-irreversible if and only if there exists \( r_0 \), an instance of \( R_0 \), such that we can not nest the relation instance \( r_n \) with scheme \( R_n \) using \( \text{NEST}_Y \) back to the instance \( r_{n-1} \) of \( R_{n-1} \). Otherwise \( \text{SEQ} \) is said to be single-nest-reversible. Note that single-nest-irreversibility is a property of the sequence \( \text{SEQ}(R_0) \) rather than a property of relation instances. That is, single-nest-irreversibility depends on the order of operators in the sequence \( \text{SEQ}(R_0) \).

Example 2. Consider \( r_0 \) with scheme \( R_0[\text{Parent}, \text{Child}] \) in Example 1. Let \( n = 4 \). Then \( r_5 = r_{n-1} \) is not equal to \( r_5 = r_{n+1} = \text{UNNEST}_\text{child}(\text{NEST}_\text{child}(r_{n-1})) \). Therefore, \( \text{SEQ}_1(R_0) \) is single-nest-irreversible.

Single-nest-irreversibility sometimes destroys semantics associated with \( R_{n-1} \), and is undesirable. We give an example.

Example 3. Consider \( r_0 \) and \( \text{SEQ}_1(R_0) \) given in Example 1. Relation instances \( r_2 \) and \( r_3 \) have the additional semantics of "family" associated with each tuple. That is, the set of children in a given tuple form the children of a family. For example, John had a family once, and hence the first two tuples in \( r_0 \), and John later had a family with Nancy as a child. The "family" semantics is lost in \( r_5 \) due to the single-nest-irreversibility of \( \text{SEQ}_1(R_0) \).

Thus, we argue that single-nest-irreversibility is a semantic integrity constraint that needs to be enforced in nested relational databases. Users must be allowed to define an integrity constraint (to preserve the semantics) that asserts that "sequences that define views must be single-nest-reversible". Please note that users can always reconstruct \( r_{n-1} \) from \( r_{n+1} \) by first unnesting all attributes of \( r_{n+1} \) to obtain \( r_0 \) and then applying the first \((n-1)\) operators of \( \text{SEQ} \) to \( r_0 \). However, such an approach is prohibitively costly.

Example 4. Consider \( R_0[\text{Parent}, \text{Child}] \) and \( r_0 \) in Example 1. The sequence \( \text{SEQ}_0(R_0) = \text{NEST}_\text{parent}, \text{NEST}_\text{child}, \text{UNNEST}_\text{parent}, \text{UNNEST}_\text{child} \) is single-nest-irreversible. However, the sequence \( \text{SEQ}_2(R_0) = \text{NEST}_\text{parent}, \text{NEST}_\text{child}, \text{UNNEST}_\text{child} \) is single-nest-reversible. When the single-nest-reversibility is enforced, the view \( V_1 = \text{SEQ}_1(R_0) \) is not permitted, but the view \( V_2 = \text{SEQ}_2(R_0) \) is.

In this paper, we present (1) a necessary and sufficient condition that decides, using only the sequence \( \text{SEQ} \), as to whether \( \text{SEQ}(R_0) \) is irreversable or not, and (2) the algorithm \( \text{TEST} \) which implements the condition in (1) in \( O(n^2) \) time where \( n \) is the length of the sequence \( \text{SEQ} \).

The condition we derive for testing single-nest-reversibility is surprisingly complex, makes use of multidimensional geometric structures, and is of theoretical interest on its own. We now give applications in the database view management/update area.

Without the condition derived in this paper, the only way of enforcing single-nest-reversibility in view \( r_n \) with scheme \( R_n \) is by evaluating intermediate relation instances and checking at execution time a manifestation that \( r_{n-1} = \text{NEST}_Y( \text{UNNEST}_Y^{-1}(r_{n-1})) \). First, this test has to be made each time the base relation and the view instance are updated, and is costly. Second, if the single-nest-irreversibility is detected at run-time, there are only two after-the-fact actions, both of which are undesirable: (a) disable the requested view (or base relation) update, or (b) remove the view \( r_n = \text{SEQ}(R_0) \) from the set of existing views.

The single-nest-reversibility of \( \text{SEQ}(R_0) \) is also related to functional (and strong functional) dependencies. Let us extend functional dependencies (fd) into nested relations [FSTG85] as follows: A nested relation scheme \( R \) satisfies the functional dependency \( Y \rightarrow W \) iff, for all tuples \( t_1 \) and \( t_2 \) in any instance of \( R \), whenever \( t_1[V] = t_2[V] \) then \( t_1[W] = t_2[W] \).
the single-nest-reversibility condition. If, for all instances \( r_{n-1} \) of \( R_{n-1} \), \( r_n = \text{NEST}(\text{UNNEST}(r_{n-1})) \) (i.e., \( \text{SEQ}(R) \) is single-nest-reversible) then clearly (Attribute-set-of-\( r_{n-1} \cdot Y^n \)) \( \rightarrow \) \( Y^n \) in \( R_{n-1} \). We call such a dependency a sequence-induced functional dependency, and argue that sequence-induced fds in views must be enforced.

**Example 5.** Consider the view \( V_2 = R_2 \) in Example 1. Since both \( \text{UNNEST}_{\text{Child}}(R_2) \) and \( \text{UNNEST}_{\text{Parent}}(R_2) \) are single-nest-reversible, the functional dependency \( C^* \rightarrow P^* \) and \( P^* \rightarrow C^* \) hold in \( V_2 \) (\( P = \text{Parent} \) and \( C = \text{Child} \)). Thus, any view update must be verified whether it satisfies \( C^* \rightarrow P^* \) and \( P^* \rightarrow C^* \) before it is allowed.

Sequence-induced fds are direct consequences of single-nest-reversibility. Therefore, a test of single-nest-reversibility is also a test of the existence of \( \approx \) sequence-induced fd. In this paper, we give sound and complete axioms (i.e., inference rules) for functional (and strong functional) dependencies in one-level nested relations.

The literature related to this work either considers only a sequence of NEST operators \([\text{FiG84, FiG85}]\) or discusses only a very restricted version of NEST-UNNEST sequences in which \( \text{NEST}_x, \text{NEST}_y, \) subsequence is always followed by \( \text{UNNEST}_x, \text{UNNEST}_y, \) subsequence. \([\text{FiT85}]\) discusses how nesting preserves, alters, or destroys dependencies in 1NF relations.

The necessary and sufficient condition for single-nest-reversibility is given in section 3. Section 4 gives the algorithm \( \text{TEST} \) that checks whether \( \text{SEQ} \) is single-nest-reversible or not. Section 5 extends the algorithm \( \text{TEST} \) to incorporate functional dependencies. Section 6 extends Armstrong axioms into a sound and complete set of axioms in one-level nested relations. Complete proofs of all lemmas and theorems in this paper are in \([\text{YaoO89}]\).

### 2. Terminology and Definitions

For the sake of brevity, in the rest of the paper, we will use the term \((ir)\)-reversibility for the term single-nest-(ir) reversibility. The attribute set of \( R_i \) is denoted by \( U_i \), \( 0 \leq i \leq n \). We now give the definitions of NEST and UNNEST for one-level nested relations.

**Definition 1(a).** (One-level NEST) Consider the attribute set of relation scheme \( R \), denoted by \( \text{Attr}(R) \). Let \( Y \in \text{Attr}(R) \) be an atomic attribute, and \( CY = \text{Attr}(R) \cdot \{ Y \} \). Consider a relation instance \( r \) with scheme \( R \). For each tuple \( t \in r(CY) \), tuple \( w_i \) is defined as \( w_i[Y] = \{ t'[Y] \mid t' \in r \) and \( t'[CY] = t \} \). Then, the (one-level) NEST on the attribute \( Y \), denoted by \( \text{NEST}_Y \), is defined as \( \text{NEST}_Y(r) = \{ w_i \mid t \in r(CY) \} \).

**Definition 1(b).** (One-level UNNEST) Let \( r \) be a relation instance of \( R \). Let \( X^* \in \text{Attr}(R) \) be a set-valued attribute, and \( CX = \text{Attr}(R) \cdot \{ X^* \} \). For each tuple \( t' \in r \), the set \( U_i(t') \) is defined as follows \( U_i(t') = \{ t \mid t[X] = t'[X] \text{ and } t(CX) = t'[CX] \} \). Then, the (one-level) UNNEST on the attribute \( X^* \), denoted by \( \text{UNNEST}_{x^*} \), is defined as \( \text{UNNEST}_{x^*}(r) = \bigcup_{i} U_i(t') \).

Let \( \text{SEQ} = O_1O_2 \cdots O_s \) be a NEST and UNNEST sequence; \( [O_s, O_i] \) denotes a subsequence (range) of \( \text{SEQ} \) where \( s < t \). The following assumptions, both to be relaxed later, are added for simplicity.

**Assumptions:**

A1) The flat relation \( r_0 \) does not have any data dependency (e.g., FD, MVD, etc.) constraints.

A2) NEST-UNNEST operators operate on single nontrivial attributes. "Nontrivial" means that the NEST operator nests an unnested attribute, and the UNNEST operator unnesses a nested attribute. That is, we don't permit \( \text{UNNEST}_{x^*}, \text{UNNEST}_{x^*}, \text{NEST}_{x^*}, \text{NEST}_{x^*} \) subsequence and \( \text{NEST}_{x^*}, \text{NEST}_{x^*} \) operators in the sequence.

From now on, we will use the shorthand notation \( X \) and \( X \) for \( \text{NEST}_{x^*} \) and \( \text{UNNEST}_{x^*} \), respectively.

A tuple of \( r_0 \) is called a flat tuple. A flat tuple \( ft \) is contained in tuple \( t \) of \( r_i \) iff the set of tuples obtained by unnesting all nested attributes of \( t \) contains \( ft \). We say that two flat tuples are merged in \( r_i \) if there is a tuple of \( r_i \) that contains both flat tuples. Similarly, two flat tuples of \( r_0 \) are split in \( r_i \) if there is no tuple in \( r_i \) that contains both flat tuples.

Two (possibly flat) tuples \( t_i \) and \( t_j \) in \( r_i \), \( 0 \leq i \leq n \), are said to be almost the same except attribute \( A \) iff \( t_i[A] \neq t_j[A] \) and \( t_i[U_i \cdot A] = t_j[U_i \cdot A] \). To illustrate various concepts, we will occasionally use "the tuple graph" (of a relation instance) that has flat tuples as its nodes and that represents the "almost the same relationship".
The edge with label A denotes the fact that $f_{t1}$ and $f_{t2}$ are almost the same except A. Nodes (tuples) in a circle are merged. Therefore $f_{t2}$ and $f_{t3}$ are merged in $r_{1}$. Figure 2. Illustration of the Tuple Graph of a Relation Instance

Given a sequence $O_{1}O_{2}\cdots O_{n+1}$, the notation $Last_{x}$ ($O_{i}$) represents the last $NEST_{x}$ or $UNNEST_{x}$ operator before $O_{i}$ in the sequence $O_{1}O_{2}\cdots O_{n+1}$; if there are no such operators then $Last_{x}(O_{i})$ returns $O_{0}$ which is a pseudo-operator. Similarly, $Last_{x}(O_{1})$ and $Last_{x}(O_{i})$ represents the last $NEST_{x}$ operator and the last $UNNEST_{x}$ operator, respectively, before $O_{i}$ in the sequence $O_{1}O_{2}\cdots O_{n+1}$.

Example 6. Consider $Z_{1}Y_{2}X_{3}\bar{Z}_{4}\bar{Y}_{5}Z_{6}\bar{X}_{7}Y_{8}$, then $Last_{z}(\bar{Y}_{5}) = Z_{1}$, $Last_{z}(\bar{Y}_{5}) = Z_{4}$.

Definition 2. (SM-range) A range $[Y_{1}, Y_{2}]$ is a split-merge range (SM-range) of $Y$ iff there exists an instance $r_{0}$ with two flat tuples $f_{t1}$ and $f_{t2}$ that are almost the same except attribute $Y$, and $f_{t1}$ and $f_{t2}$ are split in $r_{1}$, and merged in $r_{2}$.

Definition 3. (MS-range) A range $[Y_{1}, Y_{2}]$ is a merge-split range (MS-range) of $Y$ iff there exists an instance $r_{0}$ with two flat tuples $f_{t1}$ and $f_{t2}$ that are almost the same except attribute $Y$, and $f_{t1}$ and $f_{t2}$ are merged in $r_{1}$, and split in $r_{2}$.

Merge-split (and, also, split-merge) range plays an important role in (single-nest-)irreversibility since it indicates that two "almost the same" tuples are split from the original single tuple that "contained" them.

3. Single-Nest-Irreversibility Property

3.1. SM-range

The following definition is equivalent to the statement that $SEQ(R_{0})$ is (single-nest-)irreversible iff $Y_{n+1}(\bar{Y}_{n})(r_{n+1})=y_{n+1}$.

Definition 4. (Single-Nest-Irreversibility) Consider a flat relation scheme $R_{0}$ and a sequence $SEQ(R_{0}) = O_{1}O_{2}\cdots O_{n}$, $O_{n} = \bar{Y}_{n}$. We say that $SEQ(R_{0})$ is (single-nest-)irreversible iff there exists an instance $r_{0}$ with scheme $R_{0}$, and $r_{i} = O_{i}(r_{i-1})$, $1 \leq i \leq n$, such that $r_{n-1}$ has two distinct tuples that are almost the same except attribute $Y$.

Theorem 1. $SEQ(R_{0})$ is irreversible iff $[Last_{y}(\bar{Y}_{n}), Y_{n+1}]$ is an SM-range of $y$.

Thus, we have converted the condition of irreversibility into the existence of a split-merge-range. Our general approach is to locate the cases that make the latter condition true. For this purpose, we will construct multidimensional geometric designs involving sets of "almost the same" tuples.

3.2. Necessity of SM-condition

In this section and the next two sections, we develop a necessary and sufficient condition based only on the sequence $O_{1}O_{2}\cdots O_{n}$ to check whether a range is an SM-(MS-) range.

Assume $[Y_{a}, \bar{Y}_{b}]$ is any subrange (ach) and there are no $Y$ or $\bar{Y}$ between $Y_{a}$ and $\bar{Y}_{b}$. Then we say that any operator $O_{1}, a<ch$, is in the Nested Zone of $Y$; otherwise, $O_{1}$ is located in the Unnested Zone of $Y$. We also denote $S_{x}$ as the set of all $NEST_{x}$ and $UNNEST_{x}$ operators between $O_{x}$ and $O_{i}$ for a given range $[O_{x}, O_{i}]$.

Definition 5. (SM-trigger condition) A range $[Y_{1}, Y_{2}]$ is said to satisfy the SM-trigger condition with respect to attribute $X$ iff there exists an attribute $X$ which is nested in $r_{x}$ such that

(a) $|S_{x}^{a}| = odd > 0$, or
(b) $|S_{x}^{e}| = even > 0$ and $[Last_{x}(O_{x}), Last_{x}(O_{i})]$ is an SM-range of $X$, or
(c) $|S_{x}^{e}| = even > 0$ and $[Last_{x}(O_{x}), Last_{x}(O_{i})]$ is an MS-range of $X$.

Please note that the SM-trigger condition iteratively makes use of SM- and MS-ranges of different attributes.

Example 7. Consider the sequence $SEQ_{1}(R_{0}) = X_{1}Y_{2}X_{3}\bar{Y}_{4}Y_{5}$ and $O_{5} = Y_{5}$, where $R_{0} = \{XY\}$. The instance $r_{0}$ is shown in Figure 3(a). Clearly, $[Y_{2}, Y_{3}]$ satisfies the SM-trigger condition. Also $f_{t1}$ and $f_{t2}$ are split in $r_{2}$ and merged in $r_{3}$, hence $[Last_{y}(\bar{Y}_{5}), Y_{3}]$ is an SM-range of $Y$. Hence $SEQ_{1}(R_{0})$ is irreversible.

Consider another sequence $SEQ_{2}(R_{0}) = X_{1}Y_{2}X_{3}\bar{Y}_{4}X_{5}\bar{Y}_{6}$ and $O_{7} = Y_{7}$, where $R_{0} = \{XYZ\}$. The
characterize the iterative use of SM-ranges (possibly end-

condition for irreversibility. Unfortunately, it merged in range then they must be merged in. Thus, \{Y_a, Y_b\} is an SM-range, and \textit{SEQ}_{3}(R_0) is irreversible.

Consider the third sequence \textit{SEQ}_{3}(R_0) = \{Z_7 X_2 Y_3 \bar{X}_5 X_6 Y_7 \} and \textit{O}_4 = Y_8, where \textit{R}_0 = \{X Y Z\}. The instance \textit{r}_0 is shown in Figure 3(c). The range \{Y_a, Y_b\} satisfies the SM-trigger condition, and \textit{ft}_1 and \textit{ft}_2 are split in \textit{r}_0 and merged in \textit{r}_8. Thus, \{Y_a, Y_b\} is an SM-range, and \textit{SEQ}_{3}(R_0) is irreversible.

\[
Y \rightarrow X
\]

(a) \textit{r}_0 Instance

\[
Y \rightarrow X \rightarrow Z
\]

(b) \textit{r}_0 Instance

\[
Y \rightarrow X \rightarrow Z
\]

(c) \textit{r}_0 Instance

Figure 3. Illustrations for Example 7

Lemma 1. If a range \{\textit{O}_a, \textit{O}_b\} is an SM-range of \textit{Y} then \{\textit{O}_a, \textit{O}_b\} satisfies the SM-trigger condition.

Therefore, the SM-trigger condition is a necessary condition for irreversibility. Unfortunately, it is not a sufficient condition. We give an example.

Example 8. Consider the sequence \{X_1 Y \bar{X}_2 Y_3 \bar{X}_5 \bar{Y}_7 \}. The reader may verify that the range \{X_5, X_4\} satisfies the SM-trigger condition. However, \{X_5, X_4\} is not an SM-range: If there exist two flat tuples \textit{ft}_1 and \textit{ft}_2 that are merged together in \textit{r}_8 then they must be merged in \textit{r}_4, and then \textit{ft}_1 and \textit{ft}_2 must be merged in \textit{r}_5 also (since \textit{Z}_4 splits these two flat tuples then they cannot be merged in \textit{r}_6).

We now present the SM-factor condition, which is also a necessary condition for irreversibility. Let us first characterize the iterative use of SM-ranges (possibly ending with an MS-range) in the SM-trigger condition. If \{\textit{O}_a, \textit{O}_b\} satisfies the SM-trigger condition with respect to \textit{X}, and \{\textit{Last}_x(\textit{O}_a), \textit{Last}_x(\textit{O}_b)\} is an MS-range then we can follow the same process to get the next range \{\textit{Last}_x(\textit{O}_a), \textit{Last}_x(\textit{O}_b)\} which is perhaps an SM-range of \textit{X}, and so on. Thus, we get a sequence \{\textit{O}_1, \textit{O}_2, ..., \textit{O}_k, \textit{O}_k\} such that\[\{\textit{O}_1, \textit{O}_2\} = \{\textit{Last}_x(\textit{O}_1), \textit{Last}_x(\textit{O}_2)\}, i=2, ..., r-1, is an SM-range of \textit{Y}_z; in addition, \{\textit{O}_z, \textit{O}_z\} = \{\textit{Last}_x(\textit{O}_z), \textit{Last}_x(\textit{O}_z)\} is an MS-range or (b) \textit{Last}_x(\textit{O}_z-1) = \textit{UNNEST}_x and \textit{z} = \textit{t} = t^2 = 0.

The example 9 and figure 4 illustrate the k-dimensional arrangement of "almost the same" tuples defined using SM-paths. In figure 4, the two tuples that are incident to the edge that is not part of the three dimensional cube finally demonstrates the irreversibility of the sequence.

Example 9. Consider the sequence \{S_4 Y_3 W_3 Y_4 \} \textit{S}_4 \textit{X}_4 \textit{W}_3 \textit{Y}_4 \textit{Z}_3 \textit{X}_4 \textit{Z}_6 \textit{W}_3 \textit{Y}_4 \textit{Z}_7 \textit{W}_3 \textit{Y}_4 \textit{Z}_8 \textit{W}_3 \textit{Y}_4 \textit{Z}_9 \textit{W}_3 \textit{Y}_4 \textit{Z}_10 \textit{W}_3 \textit{Y}_4 \textit{Z}_11 \textit{W}_3 \textit{Y}_4 \textit{Z}_12 \textit{W}_3 \textit{Y}_4 \textit{Z}_13 \textit{W}_3 \textit{Y}_4 \textit{Z}_14 \} and the relation instances \textit{r}_0 and \textit{r}_0 in Figure 3(a) and 3(b). The reader can verify that \{\textit{Z}_4 \textit{Z}_1\} is an SM-range of \textit{Z} using \textit{r}_0 or \textit{r}_0; \{\textit{X}_4, \textit{Y}_4\} is an SM-range of \textit{Y} (using \textit{r}_0 or \textit{r}_0); \{\textit{X}_2, \textit{X}_1\} is an SM-range of \textit{X} (using \textit{r}_0); and \{\textit{W}_3, \textit{W}_1\} is an SM-range of \textit{W} (using \textit{r}_0). Since the attribute \textit{S} satisfies the SM-trigger condition(a),


(ii) \{\textit{Z}_4, \textit{Z}_1\}, \{\textit{X}_4, \textit{Y}_4\}, \{\textit{W}_3, \textit{W}_1\}, \{\textit{S}_4, \textit{S}_1\} is an SM-path of \{\textit{Z}_4, \textit{Z}_1\} (see Figure 4 (c and d)).

Definition 6. (SM-factor condition) Given a range \{\textit{O}_a, \textit{O}_b\}, \textit{O}_{\textit{a}} = \textit{O}_{\textit{b}} = \textit{NEST}_x, s.t. \{\textit{O}_a, \textit{O}_b\} satisfies the SM-factor condition with respect to \textit{P} if there exists an SM-path \textit{P} of \{\textit{O}_a, \textit{O}_b\}, where \textit{P} = \{\textit{Z}_1, \textit{O}_1, ..., \textit{Z}_r, \textit{O}_r\}, r \geq 2, such that each member \{\textit{Z}_1, \textit{Z}_2, \textit{Z}_3\} \textit{O}_{\textit{a}}, \textit{O}_{\textit{b}}\} in the SM-path satisfies the following conditions:

(a) if \{\textit{Z}_1, \textit{O}_a, \textit{O}_b\} is an SM-range then \{\textit{O}_a, \textit{O}_b\} satisfies the SM-factor condition \textit{wrt} \textit{P'} where \textit{P'} is the subpath \{\textit{Z}_1, \textit{O}_a, \textit{O}_b\}, ..., \{\textit{Z}_r, \textit{O}_r\} of \textit{P}, and

(b) \{\textit{O}_a\} is located in the Unnested Zone of \textit{Y}, or
Example 10. Consider the sequence $S = Z_1 W_2 Z_3 Y_4 X_3 Y_5 Z_1 X_1 Y_2 Z_2 X_4 Z_{16}$ and the path $P = \{<Z, Z_{10}, Z_{16}>\}$.

(a) $Y_8$ and $X_5$ are located in the Unnested Zone of $Z$. Thus, $<Y, Y_8, Y_{13}>$ and $<X, X_5, X_{11}>$ both satisfy the SM-factor condition (b)(i) of $[Z_{10}, Z_{16}]$.

(b) $W_2$ is located in the Unnested Zone of $X$ and $Y$, and the Nested Zone of $Z$. Also, $[Last_t(O_2), O_{16}] = [Z_1, Z_{16}]$ is not an SM-range. However, attribute $X$ is unnested in $r_{t_6}$, and hence, $<W, W_2, \overline{W}>$ satisfies the SM-factor condition (b)(iii) of $[Z_{10}, Z_{16}]$.

(c) Follow the same process for each subpath of $P$, and each subpath satisfies the SM-factor condition (a) and (b) of the associated range $[O_s, O_t]$. For example, $X_5$ is located in the Nested Zone of $Y$. Since $[Y, Y_{13}]$ is an SM-range (e.g., consider the $r_0$ instance $f_1 \cdots f_{t_2} \cdots f_{t_3}$), $<X, X_5, X_{11}>$ satisfies the SM-factor condition (b)(ii) of $[Y, Y_{13}]$.

Hence, $[Z_{10}, Z_{16}]$ satisfies the SM-factor condition.

Lemma 2. If $[O_s, O_t]$ is an SM-range of $Y$ then $[O_s, O_t]$ satisfies the SM-factor condition.

We now give a single name, i.e., SM-condition, to the two necessary conditions for irreversibility.

Definition 7. (SM-condition) Given a range $[O_s, O_t]$, where $O_s, O_t$ nest on attribute $Y$, $[O_s, O_t]$ satisfies the SM-condition of $Y$ iff $[O_s, O_t]$ satisfies the SM-trigger condition and the SM-factor condition.

Lemma 3. If $[O_s, O_t]$ is an SM-range of $Y$ then $[O_s, O_t]$ satisfies the SM-condition of $Y$.

3.3. MS-Range and MS-Condition

Since the child-range of an SM-range is at times an MS-range, we need to develop the conditions to check for an MS-range. The conditions for MS-range are mirror images of those for the SM-range, and defined below.

Definition 8. (MS-trigger condition) A range $[Y_s, Y_t]$ is said to satisfy the MS-trigger condition with respect to attribute $X$ iff there exists an attribute $X$ which is nested in $r_t$ such that
Lemma 4. If a range $[O_s, O_t]$ is an MS-range of $Y$ then $[O_s, O_t]$ satisfies the MS-trigger condition.

Given a range $[O_s, O_t]$, where $O_s = O_t = \text{NEST}_Y$, if $[O_s, O_t]$ satisfies the MS-trigger condition with respect to $X$, and $\text{Last}_X(O_s), \text{Last}_X(O_t)$ is an MS-range then we can follow the same process to get the next range $\text{Last}_X(\text{Last}_X(O_s)), \text{Last}_X(\text{Last}_X(O_t))$ which is perhaps an MS-range of $Y$, and so on. Thus, we get a sequence $[O_s, O_t], \ldots, [O_{i+1}, O_{i+2}]$, such that $[O_{i+1}, O_{i+2}] = \text{Last}_X([O_s, O_t]), \text{Last}_X([O_{i-1}, O_i])$, $i = 2, \ldots, r$, is an MS-range of attribute $Z_i$; in addition, $[O_s, O_t], [O_{i+1}, O_{i+2}]$ is an MS-range of attribute $Z_i$; and either (a) $[O_s, O_t] = \text{Last}_X([O_{i-1}, O_i]), \text{Last}_X([O_{i+1}, O_{i+2}])$ is an SM-range or (b) $\text{Last}_X([O_{i-1}, O_i]), \text{Last}_X([O_{i+1}, O_{i+2}])$ is an SM-range with respect to $P'$.

Lemma 5. If $[O_s, O_t]$ is an MS-range of $Y$ then $[O_s, O_t]$ satisfies the MS-factor condition.

Definition 10. (MS-condition) Given a range $[O_s, O_t]$, where $O_s = O_t = \text{NEST}_Y$, if $[O_s, O_t]$ satisfies the MS-trigger condition with respect to $X$, and $\text{Last}_X(O_s), \text{Last}_X(O_t)$ is an MS-range then we can follow the same process to get the next range $\text{Last}_X(\text{Last}_X(O_s)), \text{Last}_X(\text{Last}_X(O_t))$ which is perhaps an MS-range of $Y$, and so on. Thus, we get a sequence $[O_s, O_t], \ldots, [O_{i+1}, O_{i+2}]$, such that $[O_{i+1}, O_{i+2}] = \text{Last}_X([O_s, O_t]), \text{Last}_X([O_{i-1}, O_i])$, $i = 2, \ldots, r$, is an MS-range of attribute $Z_i$; in addition, $[O_s, O_t], [O_{i+1}, O_{i+2}]$ is an MS-range of attribute $Z_i$; and either (a) $[O_s, O_t] = \text{Last}_X([O_{i-1}, O_i]), \text{Last}_X([O_{i+1}, O_{i+2}])$ is an SM-range or (b) $\text{Last}_X([O_{i-1}, O_i]), \text{Last}_X([O_{i+1}, O_{i+2}])$ is an SM-range with respect to $P'$.

3.4. Sufficiency of SM-MS-Condition

So far, we have only shown the necessity of the SM-condition for the irreversibility problem. In fact, the SM-condition is also a sufficient condition for irreversibility. The proofs of theorems 2 and 3 below are in [Yao, 89].

Theorem 2. $[O_s, O_t]$ is an SM-(MS)-range iff $[O_s, O_t]$ satisfies the SM-(MS)-condition of $Y$.

Theorem 3. Let $\text{SEQ}(R) = \text{NEST}/\text{UNNEST}$ sequence $O_1, O_2, \ldots, O_n$ on $R$, $O_n = \text{NEST}_Y$. $\text{SEQ}(R)$ is irreversible iff $\text{Last}_Y(O_n), \text{Last}_Y(O_{n+1})$ satisfies the SM-condition of $Y$.

4. Algorithm TEST to check Irreversibility

In this section, we use the result of Section 3, and give the algorithm TEST that checks if the sequence SEQ is irreversible. Theorems 4 and 5 prove that TEST correctly checks irreversibility in polynomial time. The algorithm TEST is given in Figure 5.

Theorem 4. The algorithm TEST correctly decides whether a given sequence $O_1, O_2, \ldots, O_n$ is irreversible.

Theorem 5. The algorithm TEST is a polynomial time algorithm in the worst case.

Sketch of Proof: The global data structures GSM and GMS are used to avoid calling the functions $\text{SM}(O_s, O_t)$ or $\text{MS}(O_s, O_t)$ repeatedly. After calling $\text{SM}(O_s, O_t)$ ($\text{MS}(O_s, O_t)$), the result as to whether $[O_s, O_t]$ forms an SM-(MS)-range or not will be stored into GSM$(i_1)$ (GMS$(i_1)$). Therefore, TEST executes in time $O(n^2 \times V)$, where $O$ denotes the order notation[Ahu74] and $V$ is the worst time complexity of the function $\text{SM}()$ or $\text{MS}()$. 

274
In function SM($O_x, O_y$) (or $MS()$), the two outer for-loops take time $O(n^2)$ in the worst case. Inside these two loops, the third for-loop, tracing the member in the set $F$, takes time $O(n^2)$ also; and since the subpath $C$ is a subset of $T$, finding $C$ will take time $O(n^3)$. In addition, in the last for-loop, the algorithm checks the SM-factor condition (a)(iii) in time proportional to the length of $C$ which takes $O(n)$ time. The total time $V$ of function SM is $O(n^2 * (n^2 + n)) = O(n^5)$. Thus the time complexity of TEST is $O(n^5)$ in the worst case. Q.E.D.

5. Extensions

5.1. Irreversibility in the Presence of Functional Dependencies

We now relax the assumption A1, i.e., consider flat relations with functional dependencies.

Corollary 1. Let $F$ be the nontrivial functional dependency set of the flat relation $r_0$. Consider a sequence $SEQ$ for any range $[O_x, O_y]$, where $O_x$ and $O_y$ nest on the attribute $Y$, and $Y$ appears in the righthand side of any functional dependency in $F$, $[O_x, O_y]$ is not an $SM$- or $MS$-range.

Proof: If attribute $Y$ appears in the righthand side of any functional dependency then there are no two flat tuples $f_1(r_1)$ and $f_2(r_2)$ that are almost the same except attribute $Y$. Q.E.D.

5.2. Nesting and Unnesting on Multiple Attributes

The assumption A2 in section 2 was given to simplify the notation for $NEST$ and $UNNEST$, and to give simpler examples throughout the paper. In fact, the result we derive in section 4 is independent of the assumption A2, i.e., nesting only on a single attribute. We now remove this assumption. If an operator nests on multiple attributes then we use the notation $NEST_x$, where $s = Z_1, ... Z_s$ defines the nesting attributes.


In this section, we discuss an application of the irreversibility in defining a sound and complete set of functional dependency (FD) axioms for one-level nested relations.

We first state the extension of FDs into nested relations [FSTG85]. Let $V$ and $W$ denote two sets of atomic or set-valued attributes. A relation $r$ (not necessarily in 1NF) satisfies the functional dependency $V \rightarrow\rightarrow W$ iff, for all tuples $t_1, t_2 \in r$, whenever $t_1[V] = t_2[V]$, then $t_1[W] = t_2[W]$.

In 1NF relations, FDs contain only atomic-valued attributes; and Armstrong axioms given below are complete and sound [Arms74]. (Let $U$ be the set of attributes in FDs.)

A1) If $Y \subseteq X \subseteq U$ then $X \rightarrow\rightarrow Y$
A2) If $X \rightarrow\rightarrow Y$ and $Z \subseteq U$ then $XZ \rightarrow\rightarrow YZ$
A3) If $X \rightarrow\rightarrow Y$ and $Y \rightarrow\rightarrow Z$ then $X \rightarrow\rightarrow Z$

However, in one-level nested relations, when FDs contain set-valued attributes, Armstrong axioms are sound, but not complete. We give an example.

Example 11. Consider a flat relation $r_0$ with scheme $R_0 = (XYZ)$, where $FD_0 = \{ Y \rightarrow\rightarrow Z \}$ is the functional dependency set of $r_0$. Let $r_1 = NEST_y(r_0)$, and the corresponding functional dependency set is $FD_1 = \{ Y' \rightarrow\rightarrow Z \}$. The functional dependency $XZ \rightarrow\rightarrow Y'$ also holds since $UNNEST_y(r_1)$ is not irreversible, but $XZ \rightarrow\rightarrow Y'$ is not logically implied by $FD_1$ using Armstrong axioms.

It turns out that the addition of the rule R1 (or equivalently, the rule R2 which can be checked using the algorithm TEST) given below to Armstrong axioms produces a sound and complete set of axioms for one-level nested relations.

Rule R1) Let $U$ be a set of atomic- and set-valued attributes, and $Y' \in U$. Consider a one-level nested relation scheme $R$ and $Attr(R) = U$. If $UNNEST_y(R)$ is reversible then $U \cdot Y' \rightarrow\rightarrow Y'$ holds in $R$.

Rule R2) Let $U$ be a set of atomic- and set-valued attributes, and $Y' \in U$. Consider a relation scheme $R$, where there is a sequence $SEQ$ for $SEQ(R) = R = R_0$ is a flat relation scheme, and $Attr(R) = U$. If $UNNEST_y(SEQ(R))$ is reversible then $U \cdot Y' \rightarrow\rightarrow Y'$ holds in $R$.

Theorem 6. Armstrong axioms and rule R1 (or R2) form a complete and sound set of axioms for one-level nested relations.

We now extend the result of theorem 6 to strong functional dependencies [FisG85].
Algorithm TEST(SEQ); let $O_n=\bar{Y}_n$.

<global arrays>  
Let $i,j \leq n$ and $i<j$.

1. **GSM[i,j]**  
   - **yes** if $SM(O_i,O_j)=yes$, i.e., $[O_i,O_j]$ is an SM-range,  
   - **no** if $SM(O_i,O_j)=no$, i.e., $[O_i,O_j]$ is not an SM-range,  
   - **null** if it is not known whether $[O_i,O_j]$ is an SM-range;

2. **GMS[i,j]**  
   - **yes** if $MS(O_i,O_j)=yes$, i.e., $[O_i,O_j]$ is an MS-range,  
   - **no** if $MS(O_i,O_j)=no$, i.e., $[O_i,O_j]$ is not an MS-range,  
   - **null** if it is not known whether $[O_i,O_j]$ is an MS-range;

Begin  
set the dimensions of GSM and GMS as $n+1$; and append $Y_{n+1}$ to the end of the sequence SEQ;  
for (all $i,j$) let GSM[i,j] := null, and GMS[i,j] := null  
if (SM($O_i,O_{i+1}$)=yes) then print "SEQ is irreversible"  
else print "SEQ is reversible" endif  
end.

**Boolean Function SM($O_a,O_b$)**  
/* checks if $[O_a,O_b]$ is an SM-range of attr $Y$ */
  
begin  
if (GSM[s,t]#null) then return(GSM[s,t]) endif  
F := {$Y,O_a,O_b$};  
for (b:=t-1 to 3; a:=b-1 to 1)  
begin if ($O_b$ is a NEST operator on attr $Z$; and $O_a$ is a NEST or UNNEST operator on attr $Z$) then begin  
T := $\triangle$;  
for (each member $<W,O_1,O_b>$ of F in forward order) begin if ($O_a$ is in the Unnested Zone of $W$) or ($SM(\text{Last}_a(O_a),O_m)=yes$) then add $<W,O_1,O_m>$ to T endif  
if (there exists such a subpath $C$) then begin  
if ($1S^i_{a-1}$ = odd or $MS(O_a,O_b)=yes$) then GSM[s,t]:=yes, and return(yes) endif  
endfor  
endfor  
end.

**Boolean Function MS($O_a,O_b$)**  
/* checks if $[O_a,O_b]$ is an MS-range of attr $Y$ */  
(MS() is symmetrical to SM(), and we omit it here)

Figure 5. The algorithm TEST and the Functions SM()/MS().
Definition 12. (SFD) Consider a one-level nested relation \( \mathcal{R} \) with scheme \( R \) and \( \text{Attr}(R) = U \), where \( U = (X_1, X_2, \ldots, X_t, Y_1^*, \ldots, Y_t^*, Z, \ldots) \) is a set of atomic- and set-valued attributes (please note that \( Z \) and \( X_i \) could be atomic- or set-valued attributes, and \( Y_i^* \) is a set-valued attribute). The relation \( \mathcal{R} \) satisfies the strong functional dependency (SFD) \( X_1.X_2.\ldots.Y_t^* --> Z \) iff for any two tuples \( t_1 \) and \( t_2 \) in \( \mathcal{R} \) if \( t_1[X_i] = t_2[X_i] \) and \( t_1[Y_j^*] = t_2[Y_j^*] \) \( \forall i \leq t, 1 \leq j \leq s \), then \( t_1[Z] = t_2[Z] \) holds.

Functional dependencies in nested relations form a special class of strong functional dependencies. Lemma 7 states the SFD relationship between two relations in the NL-class. Based on lemma 7, the rule R3 is true.

Lemma 7. Consider a relation \( \mathcal{R} \) with scheme \( R \) and \( \text{Attr}(R) = U \), where \( N(R) \) denotes the set of nested attributes of \( R \). Let \( Y \in A \subseteq U - N(R) \), and \( X^* \in B \subseteq N(R) \). Then the following statements are true.

(a) \( \mathcal{R} \) satisfies \( AY.B --> Z \) iff \( \text{NEST}_Y(\mathcal{R}) \) satisfies \( AY.B --> Z \).

(b) \( \mathcal{R} \) satisfies \( AY.B --> Z \) iff \( \text{UNNEST}_{X^*}(\mathcal{R}) \) satisfies \( AX.B --> Z \).

Rule R3) Let \( U \) be a set of atomic- and set-valued attributes, and \( Y^* \in U \). Consider a one-level NL-class relation scheme \( R \) and \( \text{Attr}(R) = U \). If a SFD \( AY.B --> Z \) holds in \( \text{NEST}_{Y^*}(R) \), where \( A, B^* \), and \( Z \) represent the disjoint subsets of \( U \) and \( B^* \) only contains set-valued attributes, then \( AY.B --> Z \) holds in \( R \), and vice versa.

The rule R3 gives us a method to obtain sound and complete SFD axioms for one-level nested relations.

Theorem 7. Armstrong axioms, rule R1 (or R2) and rule R3 form a complete and sound set of axioms for strong functional dependencies in one-level nested relations.

Please note that, when given a functional dependency set \( FD_0 \) in \( R_0 \), using theorems 6 and 7, we can find all of the (functional and strong functional) dependencies, i.e., \( FD^*_0 \), in \( R_0 \) that are logically implied by \( FD_0 \).

7. Conclusions and Further Work

In this paper, we first give a necessary and sufficient condition that decides if \( \text{SEQ}(R_0) \) is (single-nest-)irreversible, and then we develop the algorithm TEST which checks the SM-/MS-ranges and the reversibility of an unnest operator in a nest/unnest sequence. We also prove that the irreversibility problem is related to the sound and complete functional dependency axioms (and SFD axioms) in one-level nested relations.

Although arbitrarily nested relations have not been discussed in this paper, we believe that our results can be extended to nested relations. Another research direction is to consider the irreversibility problem when relation schemes contain other dependencies, e.g., multivalued dependencies.

8. References


