Available online at www.sciencedirect.com
SCIENCE DIRECT®
Electronic Notes in Theoretical Computer Science
ELSEVIER Electronic Notes in Theoretical Computer Science 123 (2005) 125–137
www.elsevier.com/locate/entcs

The Implication Problem of Functional Dependencies in Complex-value Databases

Sven Hartmann[1] and Sebastian Link[2]

Information Science Research Centre, Massey University, Palmerston North, New Zealand

Abstract

Modern applications increasingly require the storage of data beyond relational structure. The challenge of providing well-founded data models that can handle complex objects such as lists, sets, multisets, unions and references has not been met yet. The success of such data models will greatly depend on the existence of automated database design techniques that generalise achievements from relational database design. In this paper, a provably-correct and polynomial-time algorithm for deciding implication of functional dependencies in the presence of all combinations of records, lists, sets, and multisets is proposed. The notion of a functional dependency is based on a Brouwerian algebra of subattributes, yielding a complementary expressiveness.

Keywords: Logic in Databases, Implication Problem, Functional Dependency, Brouwerian Algebra, Data Type

1 Introduction

Functional dependencies (FDs) were introduced in the context of the relational data model (RDM) by Codd in 1972 (see [18]). FDs are expressions of the form $X \rightarrow Y$ on a relation schema R with $X, Y \subseteq R$. A relation r over R is said to satisfy the FD $X \rightarrow Y$ if and only if any two tuples of r that coincide on X also coincide on Y. FDs are not independent from one another. That is, an FD $X \rightarrow Y$ is *implied* by a set Σ of FDs, if $X \rightarrow Y$ is satisfied by every relation which already satisfies all dependencies in Σ. A sound and

[1] Email:s.hartmann@massey.ac.nz
[2] Email:s.link@massey.ac.nz

doi:10.1016/j.entcs.2004.04.051

complete set of inference rules for the implication of FDs in the RDM has been discovered by Armstrong in [3]. In the context of the RDM such inference rules are easily available. The set of all attribute sets for some relation schema forms a Boolean algebra with respect to set inclusion, union, intersection and complement. On the basis of Armstrong's axiomatisation, polynomial time algorithms for deciding the implication problem [7], deciding the equivalence of two given sets of FDs [10] and deriving minimal covers for FDs [37] have been developed. A solution to these problems was a big step towards automated database schema design [10] which some researchers see as the ultimate goal in dependency theory [8]. Moreover, normal form proposals such as Boyce-Codd Normal Form and Third Normal Form [8,7,11] have been semantically justified [20,50] by formally proving the equivalence to the absence of redundancies and abnormal update behavior using again Armstrong's axiomatisation.

During the last couple of decades, many new and different data models have been introduced. First, so called semantic data models have been developed [17,33,47], which were originally just meant to be used as design aids, as application semantics was assumed to be easier captured by these models. Later on some of these models, especially the nested relational model [35], object-oriented and object-relational models [6,24,43,44] have become interesting as data models in their own right and some dependency and normalisation theory has been carried over to these advanced data models [25,26,27,39,41,46,52]. Most recently, the major research interest is on the model of semi-structured data and XML [48]. Work on integrity constraints in the context of XML can be found in [2,15,23,22,51]. One key problem is to develop dependency theories (or preferably a unified theory) for these advanced data models. Biskup [13] lists in particular two challenges for database design theory: finding a unifying framework and extending achievements to deal with advanced database features such as complex object types. We propose to classify data models according to the type constructors which are supported by the model. The RDM, for instance, is completely captured by the record type, the nested relational data model by the record and set type. This view allows to study problems in dependency theory for various classes of dependencies in the presence of various combinations of types, as illustrated in Figure 1.

In the present paper we consider all combinations of record, set, multiset and list type that include at least the record type. The need for these various types arises from applications that store ordered relations, time-series data, meteorological and astronomical data streams, runs of experimental data, multidimensional arrays, textual information, voices, sound, images, video, etc. They have been subject to studies in the deductive and temporal database community for some time [42,40], and occur also naturally in object-oriented

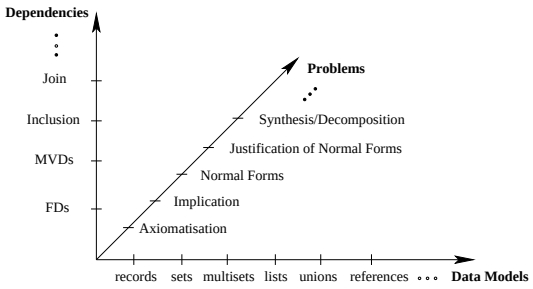

Fig. 1. Research in Dependency Theory

databases [6,43,24] and are in particular important for XML [48]. Recently, bioinformatics has become a very important field of research. Of course, lists and sets occur naturally in genomic sequence databases [36,45,14]. Multisets are the fundamental data structure of a number of computational frameworks, such as Gamma coordination language [4], the Chemical Abstract Machine [12], and P systems modeling membrane computing [19]. For a recent survey on the use of multisets in various areas of logic and computer science see [16], in which [34] specifically focuses on database systems.

2 An Abstract Data Model

The goal of this section is to provide a framework for the study of dependency classes in the context of complex-value databases. In this paper, we will deal with records, lists, sets, and multisets.

Definition 2.1 *A universe is a finite set $\mathcal{U}$ together with domains $dom(A)$ for all $A \in \mathcal{U}$. The elements of $\mathcal{U}$ are called flat attributes.* □

In the following definition we use a set $\mathcal{L}$ of labels, and assume that the symbol λ is neither a flat attribute nor a label, i.e., $\lambda \notin \mathcal{U} \cup \mathcal{L}$. Moreover, flat attributes are not labels and vice versa, i.e., $\mathcal{U} \cap \mathcal{L} = \emptyset$.

Definition 2.2 *Let $\mathcal{U}$ be a universe and $\mathcal{L}$ a set of labels. The set $\mathcal{N}A = \mathcal{N}A(\mathcal{U}, \mathcal{L})$ of nested attributes over $\mathcal{U}$ and $\mathcal{L}$ is the smallest set satisfying the following conditions:*

- $\lambda \in \mathcal{N}A$, *and* $\mathcal{U} \subseteq \mathcal{N}A$,
- *for* $L \in \mathcal{L}$ *and* $N_1, \ldots, N_k \in \mathcal{N}A$ *with* $k \geq 1$ *we have* $L(N_1, \ldots, N_k) \in \mathcal{N}A$,
- *for* $L \in \mathcal{L}$ *and* $N \in \mathcal{N}A$ *we have* $L\{N\}, L\langle N\rangle, L[N] \in \mathcal{N}A$.

We call λ null attribute, $L(N_1, \ldots, N_k)$ record-valued attribute, $L\{N\}$ set-valued attribute, $L\langle N\rangle$ multiset-valued attribute, and $L[N]$ list-valued attribute. □

We extend the mapping *dom* from flat attributes to nested attributes.

Definition 2.3 *For a nested attribute $N \in \mathcal{N}A$ we define the domain $dom(N)$ as follows:*

- $dom(\lambda) = \{ok\}$,
- $dom(L(N_1, \ldots, N_k)) = \{(v_1, \ldots, v_k) \mid v_i \in dom(N_i) \text{ for } i = 1, \ldots, k\}$, *i.e., the set of all k-tuples $(v_1, \ldots, v_k)$ with $v_i \in dom(N_i)$ for all $i = 1, \ldots, k$,*
- $dom(L\{N\}) = \{\{v_1, \ldots, v_n\} \mid v_i \in dom(N) \text{ for } i = 1, \ldots, n\}$, *i.e., $dom(L\{N\})$ is the set of all finite subsets of $dom(N)$,*
- $dom(L\langle N\rangle) = \{\langle v_1, \ldots, v_n\rangle \mid v_i \in dom(N) \text{ for } i = 1, \ldots, n\}$, *i.e., $dom(L\langle N\rangle)$ is the set of all finite multisets with elements in $dom(N)$,*
- $dom(L[N]) = \{[v_1, \ldots, v_n] \mid v_i \in dom(N) \text{ for } i = 1, \ldots, n\}$, *i.e., the set of all finite lists with elements in $dom(N)$.* □

We denote empty set, empty multiset, and empty list by $\emptyset, \langle\ \rangle, [\ ]$, respectively. Nested attributes can be partially ordered according to the information content they represent.

Definition 2.4 The *subattribute relation* $\leq$ on the set of nested attributes $\mathcal{N}$A over $\mathcal{U}$ and $\mathcal{L}$ is defined by the following rules, and the following rules only:

- $N \leq N$ for all $N \in \mathcal{N}$A, and $\lambda \leq A$ for all $A \in \mathcal{U}$,
- $\lambda \leq N$ for all set-, multiset- and list-valued attributes $N \in \mathcal{N}$A,
- $L(N_1, \ldots, N_k) \leq L(M_1, \ldots, M_k)$ whenever $N_i \leq M_i$ for all $i = 1, \ldots, k$,
- $L\{N\} \leq L\{M\}$, $L\langle N\rangle \leq L\langle M\rangle$, and $L[N] \leq L[M]$, if $N \leq M$.

For $N, M \in \mathcal{N}$A we say that M is a *subattribute* of N if and only if $M \leq N$ holds. We write $M \not\leq N$ if and only if M is not a subattribute of N. □

The subattribute relation is indeed a partial order on nested attributes, i.e., reflexive, anti-symmetric and transitive. Informally, $M \leq N$ for $N, M \in \mathcal{N}$A if and only if M comprises at most as much information as N does.

Definition 2.5 *Let $N, M \in \mathcal{N}A$ with $M \leq N$. The projection function $\pi^N_M : Dom(N) \to Dom(M)$ is defined as follows:*

- *if $N = M$, then $\pi^N_M = id_{Dom(N)}$ is the identity on $dom(N)$,*
- *if $M = \lambda$, then $\pi^N_\lambda : Dom(N) \to \{ok\}$ maps every $v \in Dom(N)$ to ok,*
- *if $N = L(N_1, \ldots, N_k)$ and $M = L(M_1, \ldots, M_k)$, then $\pi^N_M = \pi^{N_1}_{M_1} \times \cdots \times \pi^{N_k}_{M_k}$ mapping $(v_1, \ldots, v_k) \in Dom(N)$ to $(\pi^{N_1}_{M_1}(v_1), \ldots, \pi^{N_k}_{M_k}(v_k)) \in Dom(M)$,*
- *if $N = L\{N'\}$ and $M = L\{M'\}$, then $\pi^N_M : Dom(N) \to Dom(M)$ maps every set $S \in Dom(N)$ to the set $\{\pi^{N'}_{M'}(s) : s \in S\} \in Dom(M)$,*

- *if* $N = L\langle N'\rangle$ *and* $M = L\langle M'\rangle$, *then* $\pi^{N'}_{M'} : Dom(N) \to Dom(M)$ *mapping* $S \in Dom(N)$ *to* $\langle \pi^{N'}_{M'}(s) \ : \ s \in S\rangle \in Dom(M)$, *and*
- *if* $N = L[N']$ *and* $M = L[M']$, *then* $\pi^{N}_{M} : Dom(N) \to Dom(M)$ *maps every list* $[v_1, \ldots, v_n] \in Dom(N)$ *to the list* $[\pi^{N'}_{M'}(v_1), \ldots, \pi^{N'}_{M'}(v_n)] \in Dom(M)$. □

It follows, in particular, that $\emptyset, \langle\ \rangle, [\ ]$ are always mapped to themselves, except when projected on the null attribute λ in which each of them is mapped to *ok*. Fix a set $\mathcal{U}$ of attribute names, and a set $\mathcal{L}$ of labels.

Definition 2.6 Let $N \in \mathcal{N}\mathrm{A}$ be a nested attribute. The set $Sub(N)$ of *subattributes* of N is $Sub(N) = \{M \mid M \leq N\}$. The *bottom element* λ_N of $Sub(N)$ is given by $\lambda_N = L(\lambda_{N_1}, \ldots, \lambda_{N_k})$ whenever $N = L(N_1, \ldots, N_k)$, and $\lambda_N = \lambda$ whenever N is not a record-valued attribute. □

We study the algebraic structure of $Sub(N)$. A *Brouwerian algebra* [38] is a lattice $(L, \sqsubseteq, \sqcup, \sqcap, \dot{-}, 1)$ with top element 1 and a binary operation $\dot{-}$ which satisfies $a \dot{-} b \sqsubseteq c$ iff $a \sqsubseteq b \sqcup c$ for all $c \in L$. In this case, the operation $\dot{-}$ is called the *pseudo-difference*. The *Brouwerian complement* $\neg a$ of $a \in L$ is then defined by $\neg a = 1 \dot{-} a$. The system of all closed subsets of a topological space is a well-known Brouwerian algebra [38].

Definition 2.7 *Let* $N \in \mathcal{N}A$ *and* $X, Y \in Sub(N)$. *The join* $X \sqcup_N Y$, *meet* $X \sqcap_N Y$ *and pseudo-difference* $X \dot{-}_N Y$ *of* X *and* Y *in* $Sub(N)$ *are inductively defined as follows:*

- *if* $X \leq Y$, *then* $X \sqcup_N Y = Y, X \sqcap_N Y = X$ *and* $X \dot{-}_N Y = \lambda_N$,
- $X \dot{-}_N \lambda_N = X$,
- *if* $N = L\{M\}$, $X = L\{X'\}, Y = L\{Y'\}$, *then* $X \circ_N Y = L\{X' \circ_M Y'\}$ *for* $\circ \in \{\sqcup, \sqcap\}$ *and if* $X \not\leq Y$, *then* $X \dot{-}_N Y = L\{X' \dot{-}_M Y'\}$,
- *if* $N = L\langle M\rangle$, $X = L\langle X'\rangle, Y = L\langle Y'\rangle$, *then* $X \circ_N Y = L\langle X' \circ_M Y'\rangle$ *for* $\circ \in \{\sqcup, \sqcap\}$ *and if* $X \not\leq Y$, *then* $X \dot{-}_N Y = L\langle X' \dot{-}_M Y'\rangle$,
- *if* $N = L[M]$, $X = L[X'], Y = L[Y']$, *then* $X \circ_N Y = L[X' \circ_M Y']$ *for* $\circ \in \{\sqcup, \sqcap\}$ *and if* $X \not\leq Y$, *then* $X \dot{-}_N Y = L[X' \dot{-}_M Y']$,
- *if* $N = L(N_1, \ldots, N_k), X = L(X_1, \ldots, X_k)$ *and* $Y = L(Y_1, \ldots, Y_k)$, *then* $X \circ_N Y = L(X_1 \circ_{N_1} Y_1, \ldots, X_k \circ_{N_k} Y_k)$ *for* $\circ \in \{\sqcup, \sqcap, \dot{-}\}$. □

In order to simplify notation, occurences of λ in a record-valued attribute are usually omitted if this does not cause any ambiguities. If the context allows, we omit the index N from the operations $\sqcup_N, \sqcap_N, \dot{-}_N$ and from λ_N. Given some nested attribute $N \in \mathcal{N}\mathrm{A}$ and $Y, Z \in Sub(N)$, we use $Y^{\mathcal{C}}_N = N \dot{-} Y$ to denote the *Brouwerian complement* of Y in $Sub(N)$. The pseudo difference $Z \dot{-} Y$ of Z and Y in $Sub(N)$ satisfies $Z \dot{-} Y \leq X$ if and only if $Z \leq Y \sqcup X$ for all $X \in Sub(N)$. Consequently, for all $X \in Sub(N)$ holds $Y^{\mathcal{C}} \leq X$ if and only

if $X \sqcup Y = N$ holds.

Theorem 2.8 *$(Sub(N), \leq, \sqcup_N, \sqcap_N, \dot{-}_N, N)$ forms a Brouwerian algebra for every $N \in \mathcal{NA}$.* □

3 Axiomatising Functional Dependencies

We define FDs on a nested attribute and introduce some sound inference rules for the implication of FDs.

Definition 3.1 *Let $N \in \mathcal{NA}$ be a nested attribute. A functional dependency on N is an expression of the form $\mathcal{X} \to \mathcal{Y}$ where $\mathcal{X}, \mathcal{Y} \subseteq Sub(N)$ are non-empty. A set $r \subseteq Dom(N)$ satisfies an FD $\mathcal{X} \to \mathcal{Y}$ on N, denoted by $\models_r \mathcal{X} \to \mathcal{Y}$, if and only if $\pi_Y^N(t_1) = \pi_Y^N(t_2)$ holds for all $Y \in \mathcal{Y}$ whenever $\pi_X^N(t_1) = \pi_X^N(t_2)$ holds for all $X \in \mathcal{X}$ and any $t_1, t_2 \in r$.* □

The notions of implication ($\models$) and derivability ($\vdash_{\mathfrak{R}}$) with respect to a rule system $\mathfrak{R}$ for FDs on a nested attribute can be defined analogously to the notions in the RDM (see for instance [1, p. 163-168]). Note that in the case of FDs the finite implication problem coincides with the unrestricted implication problem. We are interested in the set of all FDs implied by Σ, i.e., $\Sigma^* = \{\varphi \mid \Sigma \models \varphi\}$. Our aim is finding a set $\mathfrak{R}$ of inference rules which is *sound* ($\Sigma^+ \subseteq \Sigma^*$) and *complete* ($\Sigma^* \subseteq \Sigma^+$), where $\Sigma^+ = \{\varphi \mid \Sigma \vdash_{\mathfrak{R}} \varphi\}$ is the set of FDs derivable from Σ using only inference rules from $\mathfrak{R}$. The following example reveals a fundamental difference between sound inference rules in the RDM and our data model.

Example 3.2 *Let $N = Tennis\{Match(Winner, Loser)\}$ and $r = \{t_1, t_2\} \subseteq Dom(N)$ an instance with $t_1 = \{(Becker, Agassi), (Stich, McEnroe)\}$ and $t_2 = \{(Becker, McEnroe), (Stich, Agassi)\}$. The projections of t_1 and t_2 coincide on $Tennis\{Match(Winner)\}$ and $Tennis\{Match(Loser)\}$.* □

Example 3.2 shows that Definition 3.1 of an FD $\mathcal{X} \to \mathcal{Y}$ on some nested attribute N cannot be simplified to an expression of the form $X \to Y$ with $X, Y \in Sub(N)$. In general, the values on subattributes X and Y do not determine the value on $X \sqcup Y$. The following condition is sufficient and necessary when values on subattributes X and Y determine the value on $X \sqcup Y$.

Definition 3.3 *Let $N \in \mathcal{NA}$. The subattributes $X, Y \in Sub(N)$ are reconcilable if and only if at least one of the following conditions is satisfied*

- *$Y \leq X$ or $X \leq Y$,*
- *$N = L(N_1, \ldots, N_k), X = L(X_1, \ldots, X_k), Y = L(Y_1, \ldots, Y_k)$ where X_i and Y_i are reconcilable for all $i = 1, \ldots, k$,*

• $N = L[N'], X = L[X'], Y = L[Y']$ *where* X' *and* Y' *are reconcilable.* □

The next result has been proven in [31] which extends the work in [27].

Theorem 3.4 *The generalised Armstrong Axioms for FDs, i.e.*

$$\frac{}{\mathcal{X} \to \mathcal{Y}}\ \mathcal{Y} \subseteq \mathcal{X}, \quad \frac{}{\{X\} \to \{Y\}}\ Y \leq X, \quad \frac{\mathcal{X} \to \mathcal{Y}}{\mathcal{X} \to \mathcal{X} \cup \mathcal{Y}},$$

$$\frac{}{\{X, Y\} \to \{X \sqcup_N Y\}}\ X, Y \text{ reconcilable}, \quad \frac{\mathcal{X} \to \mathcal{Y},\ \mathcal{Y} \to \mathcal{Z}}{\mathcal{X} \to \mathcal{Z}},$$

are minimal, sound and complete for the implication of FDs in the presence of records, lists, sets and multisets. □

4 A polynomial-time Membership Algorithm

We will now present a provably-correct membership algorithm that works in polynomial time. Similar to the RDM [7] we introduce the notion of a closure for a set of nested attributes with respect to a given set of FDs.

Definition 4.1 *Let* $N \in \mathcal{NA}$, $\mathcal{X} \subseteq Sub(N)$ *a set of subattributes of* N, *and* Σ *a set of FDs on* N. *The closure* $\mathcal{X}^+ \subseteq Sub(N)$ *of* $\mathcal{X}$ *with respect to* Σ *is* $\mathcal{X}^+ = \{Z : \mathcal{X} \to \{Z\} \in \Sigma^+\}$. □

We then have $\mathcal{Y} \subseteq \mathcal{X}^+$ iff $\Sigma \models \mathcal{X} \to \mathcal{Y}$. It is even sufficient to compute the set $\mathcal{X}^+_{\max}$ of $\leq$-maximal subattributes of $\mathcal{X}^+$ since $\mathcal{Y} \subseteq \mathcal{X}^+$ if and only if $\mathcal{Y} \subseteq_{\text{gen}} \mathcal{X}^+_{\max}$, where $\mathcal{Y} \subseteq_{\text{gen}} \mathcal{X}$ holds iff for all $Y \in \mathcal{Y}$ there is some $X \in \mathcal{X}$ with $Y \leq X$ (Hoare-ordering). In order to solve the implication problem for FDs on some nested attribute N we will split N into mutually reconcilable subattributes N_i.

Definition 4.2 *Let* $N \in \mathcal{NA}$. *A nested attribute* $N_i \in Sub(N)$ *is a unit of* N *if and only if* N_i *is* $\leq$*-maximal with the property that all reconcilable* $X, Y \leq N_i$ *satisfy* $X \leq Y$ *or* $Y \leq X$. *The set of all units of* N *is denoted by* $\mathcal{U}(N)$. □

Given some set $\mathcal{X} \subseteq Sub(N)$ of nested attributes, the function $\max(\mathcal{X})$ returns all maximal elements of $\mathcal{X}$ with respect to $\leq$. Moreover, if $\mathcal{U}(N) = \{N_1, \ldots, N_k\}$, then $\mathcal{X}_i = \{X \sqcap N_i : X \in \mathcal{X}\}$ for $i = 1, \ldots, k$.

Algorithm 1 (Nested Attribute Closure)
Input: $\boldsymbol{N} \in \boldsymbol{\mathcal{NA}}$, $\boldsymbol{\mathcal{X}} \subseteq \boldsymbol{Sub(N)}$, set $\boldsymbol{\Sigma}$ of FDs on $\boldsymbol{N}$
Output: $\boldsymbol{\mathcal{X}^{\text{alg}}_{\max}}$
Method:
VAR $\mathcal{X}_i^{\text{new}}, \mathcal{X}_i^{\text{old}}, \mathcal{U}_i, \mathcal{V}_i \subseteq Sub(N)$ for $i = 1, \ldots, k$, $N_1, \ldots, N_k \in Sub(N)$;

Compute $\mathcal{U}(N) = \{N_1, \ldots, N_k\}$;
FOR $i = 1$ **TO** k **DO** $\mathcal{X}_i^{\text{new}} := \max(\{X \sqcap N_i \ : \ X \in \mathcal{X}\})$;
REPEAT
  FOR $i = 1$ **TO** k **DO** $\mathcal{X}_i^{\text{old}} := \mathcal{X}_i^{\text{new}}$;
  FOR each $\mathcal{U} \rightarrow \mathcal{V} \in \Sigma$ **DO**
    IF $\mathcal{U}_i \subseteq_{\text{gen}} \mathcal{X}_i^{\text{new}}$ for $i = 1, \ldots, k$ **THEN**
      FOR $i = 1$ **TO** k **DO** $\mathcal{X}_i^{\text{new}} := \max(\mathcal{X}_i^{\text{new}} \cup \mathcal{V}_i)$;
    ENDIF;
  ENDDO;
UNTIL $\mathcal{X}_i^{\text{new}} = \mathcal{X}_i^{\text{old}}$ for $i = 1, \ldots, k$;
$\mathcal{X}_{\max}^{\text{alg}} := \{X_1 \sqcup \ldots \sqcup X_k \ : \ X_i \in \mathcal{X}_i^{\text{new}}\}$;
RETURN$(\mathcal{X}_{\max}^{\text{alg}})$; □

The correctness of Algorithm 1, i.e., $\mathcal{X}_{\max}^{\text{alg}} = \mathcal{X}_{\max}^{+}$ is proven in the full version of the paper.

Theorem 4.3 *Algorithm 1 is correct, i.e.,* $\mathcal{X}_{max}^{alg} = \mathcal{X}_{max}^{+}$. □

The input of Algorithm 1 consists of some nested attribute N, some set Σ of FDs on N and a set $\mathcal{X}$ of subattributes of N. We define the size m of N as the number of subattributes of N, i.e., $m = \mid Sub(N) \mid$. This is a reasonable measure since we consider sets of subattributes in general. The size s of Σ is simply defined as the number of its elements, i.e., $s = \mid \Sigma \mid$. The size of $\mathcal{X}$ is defined as $\mid \mathcal{X} \mid$. The following upper complexity bound for deciding implication in the presence of all types is proven in the full version of the paper.

Theorem 4.4 *In the presence of records, lists, sets and multisets, the implication problem* $\Sigma \models \sigma$ *for FDs on a nested attribute* N *can be solved in time* $\mathcal{O}(m^4 \cdot s \cdot \min\{m, s\})$ *where* $m = \mid Sub(N) \mid$ *and* $s = \mid \Sigma \mid$. □

Figure 2 shows upper complexity bounds for the implication problem of FDs in the presence of various types. If one of the input parameters is the nested attribute N, then let n denote the number of join-irreducible subattributes of N.

Consider a retailer which keeps track of its sales on a daily basis. For each day the sequence of incoming orderes is stored. Every order consists of information about the customer who places the order, the multiset of articles ordered, and the total value of the order. A customer is described by its name, address and payment details. Every article in that order has a title, a

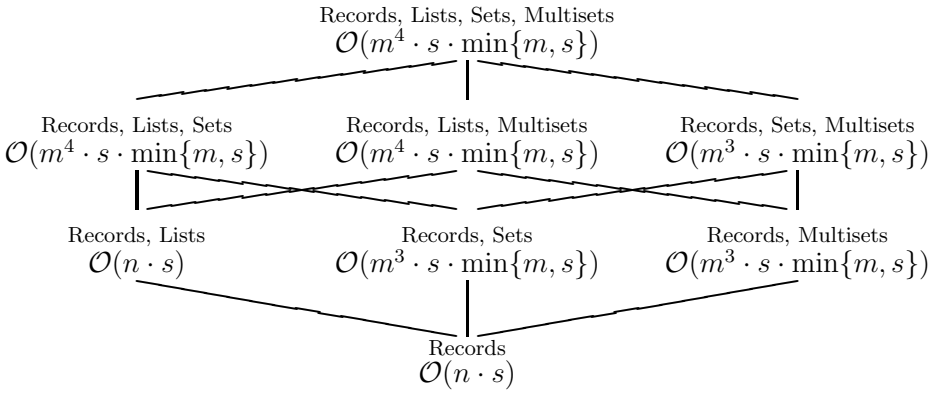

Fig. 2. Upper Complexity Bounds for the Implication Problem

description and a price. Besides the sequence of incoming orders, the retailer stores the set of different products which were sold that day. As a matter of fact, not only the title of the sold product is stored but also the name of the customer who bought it. Moreover, the company keeps information about the total price of sales, the total number of orders, the total number of products sold and the total number of shippings for each day. An order might be desribed by

Order(Cart⟨Article(Title,Type,Price)⟩,Customer(Name,Address,Payment),SubTotal)

in which a cart is used to collect a multiset of articles, and SubTotal is used to denote the total value of the order. In what follows, we will use the label Order to identify the nested attribute above. The final nested attribute N itself may look as follows

Sales(Day,List[Order],Sold{Product(Item,CustName)},Total,NOrd,NProd,NShip).

Product(Item,CustName) denotes an item together with the name of the customer who bought it. Total denotes the total price of sales, NOrd the total number of orders, NProd the total number of products and NShip the total number of shippings. A few reasonable constraints that a database designer may specify for this application are the following.

- Sales(Day) → N,
- Sales(List[Order(Cart⟨Article(Title)⟩)]) → Sales(Sold{Product(Item)}),
- Sales(List[Order(Cart⟨Article(Price)⟩)]) → Sales(List[Order(SubTotal)]),
- Sales(List[Order(SubTotal)]) → Sales(Total),
- Sales(List[Order(Customer(Name))]) → Sales(Sold{Product(CustName)}),
- Sales(List[Order(Cart⟨Article(Title)⟩,Customer(Name))]) → Sales(Sold{Product(Item,CustName)}),
- Sales(List[λ]) → Sales(NOrd), and Sales(NOrd) → Sales(List[λ]),
- Sales(List[Order(Cart⟨λ⟩)]) → Sales(NProd),
- Sales(List[Order(Cart⟨λ⟩,Customer(Address))]) → Sales(NShip).

Suppose we want to find the closure of the subattribute Sales(List[Order(Cart⟨Article(Price)⟩)]) with respect to Σ. Using Algorithm 1 we obtain

Sales(List[Order(Cart⟨Article(Price)⟩,SubTotal)],Total,NOrd,NProd).

This shows that given the list of multisets of individual prices, one can determine the list of total values of the orders, the total price of sales, the total number of orders and the total number of products ordered.

5 Some Applications

Algorithm 1 can be applied to solve several other important problems related to database design. One application is to eliminate redundant FDs. An FD σ is called *redundant* in a set Σ of FDs on some nested attribute N if and only if $(\Sigma - \{\sigma\})^+ = \Sigma^+$. A *non-redundant cover of* Σ is a set Θ of FDs on N where $\Theta^+ = \Sigma^+$ and Θ does not contain any redundant FD. In order to determine if σ is redundant in Σ, one can test whether $\sigma \in (\Sigma - \{\sigma\})^+$ holds. A subset $\Theta \subseteq \Sigma$ that is a non-redundant cover of Σ can be found using the following algorithm:

$\Theta := \Sigma$;
FOR each $\sigma \in \Sigma$ **DO**
  IF $\sigma \in (\Theta - \{\sigma\})^+$ **THEN** $\Theta := \Theta - \{\sigma\}$;
ENDDO;
RETURN(Θ);

Note that Θ will always be a subset of Σ although this is not required by the definition of a non-redundant cover. The result is dependent on the selection order of σ. The running time of the previous algorithm is $\mathcal{O}(m^4 \cdot s^2 \cdot \min\{m, s\})$ in the most general case. A set $\mathcal{X} \subseteq Sub(N)$ of subattributes of some nested attribute N is called a *superkey* for N with respect to a given set Σ of FDs on N if and only if $\Sigma \models \mathcal{X} \to N$ holds. This means that $\mathcal{X}$ is a superkey for N if and only if $N \in \mathcal{X}^+$. The problem of deciding whether $X \in Sub(N)$ is a superkey for N with respect to Σ can therefore be solved in time $\mathcal{O}(m^4 \cdot s \cdot \min\{m, s\})$ in the most general case.

6 Related and Future Work

Our definition of FDs deviates significantly from previous approaches in the nested relational data model, object-oriented data models and XML. Instead of following a path-based notion, our approach is based on a Brouwerian alge-

bra of subattributes, yielding a complementary expressivenss. In a nutshell, we are not concerned with how to represent flat data using complex object types, but with actual dependencies among complex objects. For a detailed comparison to previous works see [31].

Future work is best explained using Figure 1. The class of FDs should be studied in the presence of unions and references which are particularly important for XML [48]. We intend to extend previous work on normal forms, i.e. syntactically describe well-designed nested attributes with respect to a given set of constraints, and to semantically justify this proposal. In [30] the Nested List Normal Form has been proposed and justified. The axiomatisation in [27,31] may help to justify normal form proposals for more sophisticated combinations of types. More classes of relational dependencies are to be studied next, e.g. multi-valued dependencies (MVDs), join and inclusion dependencies. The work in [32,29] provides minimal axiomatisations for the class of MVDs and the class of FDs and MVDs in the presence of records and lists. A provably-correct polynomial time algorithm for the implication of FDs and MVDs in the presence of records and lists can be found in [28]. We intend to address normalisation for FDs and MVDs leading to a normal form proposal which is likely to deviate from a simple extension of the well-known Fourth Normal Form [20,50]. Finally, a more general treatment in which data dependencies are interpreted as formulae in a suitable logic may result in a successful treatment as in the RDM [21,49]. Here, the first-order theories of lists, multisets and sets established in [19] seem promising.

References

[1] Abiteboul, S., R. Hull and V. Vianu, "Foundations of Databases," Addison-Wesley, 1995.

[2] Arenas, M. and L. Libkin, *A normal form for XML documents*, ToDS **29** (2004), pp. 195–232.

[3] Armstrong, W. W., *Dependency structures of database relationships*, Information Processing (1974), pp. 580–583.

[4] Banatre, J. and D. Le Metayer, *Programming by multiset transformation*, Communications of the ACM **36** (1993), pp. 98–111.

[5] Beeri, C., *On the membership problem for functional and multivalued dependencies in relational databases*, ToDS **5** (1980), pp. 241–259.

[6] Beeri, C., *A formal approach to object-oriented databases*, Data and Knowledge Engeneering **5** (1990), pp. 353–382.

[7] Beeri, C. and P. A. Bernstein, *Computational problems related to the design of normal form relational schemata*, ToDS **4** (1979), pp. 30–59.

[8] Beeri, C., P. A. Bernstein and N. Goodman, *A sophisticate's introduction to database normalization theory*, in: *VLDB*, 1978, pp. 113–124.

[9] Beeri, C., R. Fagin and J. H. Howard, *A complete axiomatization for functional and multivalued dependencies in database relations*, in: *SIGMOD*, 1977, pp. 47–61.

[10] Beeri, C., A. Mendelzon, Y. Sagiv and J. Ullman, *Equivalence of relational database schemes*, SIAM Journal on Computation **10** (1981), pp. 647–656.

[11] Bernstein, P., *Synthesizing third normal form relations from functional dependencies*, ToDS **1** (1976), pp. 277–298.

[12] Berry, G. and G. Boudol, *The chemical abstract machine*, Theoretical Computer Science **96** (1992), pp. 217–248.

[13] Biskup, J., *Achievements of relational database schema design theory revisited*, in: *Semantics in databases*, number 1358 in LNCS (1998), pp. 29–54.
[14] Bry, F. and P. Kröger, *A computational biology database digest: data, data analysis, and data management*, Distributed and Parallel Databases **13** (2003), pp. 7–42.
[15] Buneman, P., S. Davidson, W. Fan, C. Hara and W. Tan, *Keys for XML*, in: *Tenth WWW Conference* (2001).
[16] Calude, C., G. Paun, G. Rozenberg and A. Salomaa, editors, "Multiset Processing, Mathematical, Computer Science, and Molecular Computing Points of View," LNCS **2235**, Springer, 2001.
[17] Chen, P. P., *The entity-relationship model: Towards a unified view of data*, ToDS 1 (1976), pp. 9–36.
[18] Codd, E. F., *Further normalization of the database relational model*, in: *Courant Computer Science Symposia 6: Data Base Systems* (1972), pp. 33–64.
[19] Dovier, A., A. Policriti and G. Rossi, *A uniform axiomatic view of lists, multisets, and sets, and the unification algorithm*, Fundamenta Informaticae **36** (1998), pp. 201–234.
[20] Fagin, R., *Multivalued dependencies and a new normal form for relational databases*, ToDS **2** (1977), pp. 262–278.
[21] Fagin, R. and M. Vardi, *The theory of data dependencies: a survey*, in: *Mathematics of Information Processing: Proceedings of Symposia in Applied Mathematics* (1986), pp. 19–71.
[22] Fan, W. and L. Libkin, *On XML integrity constraints in the presence of DTDs*, in: *PODS 2001* (2001).
[23] Fan, W. and J. Siméon, *Integrity constraints for XML*, in: *PODS 2000* (2000).
[24] Gardarin, G., J.-P. Cheiney, G. Kiernan, D. Pastre and H. Stora, *Managing complex objects in an extensible relational DBMS*, in: *VLDB, 1989, Amsterdam* (1989), pp. 55–65.
[25] Hara, C. and S. Davidson, *Reasoning about nested functional dependencies*, in: *PODS 1999* (1999), pp. 91–100.
[26] Hartmann, S., *On the implication problem for cardinality constraints and functional dependencies*, Annals of Mathematics and Artificial Intelligence **33** (2001), pp. 253–307.
[27] Hartmann, S. and S. Link, *On functional dependencies in advanced data models*, ENTCS **84** (2003).
[28] Hartmann, S. and S. Link, *A membership algorithm for functional and multi-valued dependencies in the presence of lists*, ENTCS **91** (2004).
[29] Hartmann, S. and S. Link, *Multi-valued dependencies in the presence of lists*, accepted for PODS (2004).
[30] Hartmann, S. and S. Link, *Normalisation in the presence of lists*, in: *ADC*, Conf. Research and Practice in Inf. Technology **27** (2004), pp. 53–64.
[31] Hartmann, S., S. Link and K.-D. Schewe, *Axiomatising functional dependencies in the presence of records, lists, sets and multisets*, Technical Report 1/2004, Dept of Information Systems, Massey University, New Zealand (2004).
[32] Hartmann, S., S. Link and K.-D. Schewe, *Reasoning about functional and multi-valued dependencies in the presence of lists*, in: *FoIKS*, LNCS **2942** (2004), pp. 134–154.
[33] Hull, R. and R. King, *Semantic database modeling: Survey, applications and research issues*, ACM Computing Surveys **19** (1987).
[34] Lamperti, G., M. Melchiori and M. Zanella, *On multisets in database systems*, in: *WMP 2000*, LNCS **2235** (2000), pp. 147–216.
[35] Levene, M., "The Nested Universal Relation Database Model," Springer, 1992.
[36] Li, J., S. Ng and L. Wong, *Bioinformatics adventures in database research*, in: *Database Theory - ICDT 2003*, LNCS **2572** (2002), pp. 31–46.
[37] Maier, D., *Minimum covers in relational database model*, JACM **27** (1980), pp. 664–674.
[38] McKinsey, J. and A. Tarski, *On closed elements in closure algebras*, Annals of Mathematics **47** (1946), pp. 122–146.
[39] Mok, W. Y., Y. K. Ng and D. W. Embley, *A normal form for precisely charachterizing redundancy in nested relations*, ToDS **21** (1996), pp. 77–106.
[40] Naqvi, S. and S. Tsur, "A logical language for data and knowledge bases," Computer Science Press, 1989.
[41] Özsoyoglu, Z. M. and L. Y. Yuan, *A new normal form for nested relations*, ToDS **12** (1987), pp. 111–136.
[42] Richardson, J., *Supporting lists in a datamodel*, in: *Proceeding of VLDB*, 1992, pp. 127–192.
[43] Schewe, K.-D. and B. Thalheim, *Fundamental concepts of object oriented databases*, Acta Cybernetica **11** (1993), pp. 49–85.
[44] Scholl, M. and H.-J. Schek, *A relational object model*, in: *ICDT* (1990), pp. 89–105.
[45] Seshadri, P., M. Livny and R. Ramakrishnan, *The design and implementation of sequence database system*, in: *VLDB, Mumbai, India*, 1996.

[46] Tari, Z., J. Stokes and S. Spaccapietra, *Object normal forms and dependency constraints for object-oriented schemata*, ToDS **22** (1997), pp. 513–569.
[47] Thalheim, B., "Entity-Relationship Modeling: Foundations of Database Technology," Springer-Verlag, 2000.
[48] Thompson, H. S. D. Beech, M. Maloney, and N. Mendelsohn, *XML Schema, W3C Recommendation, May 2001*, http://www.w3.org/XML/Schema (2001).
[49] Vardi, M. Y., *Fundamentals of dependency theory*, in: E. Börger, editor, *Trends in Theoretical Computer Science* (1987), pp. 171–224.
[50] Vincent, M., "The semantic justification for normal forms in relational database design," Ph.D. thesis, Monash University, Melbourne, Australia (1994).
[51] Vincent, M. W., J. Liu and C. Liu, *A redundancy free 4NF for XML*, in: *XML Database Symposium*, 2003.
[52] Weddell, G., *Reasoning about functional dependencies generalized for semantic data models*, TODS **17** (1992), pp. 32–64.