On the Interaction of Functional and Inclusion Dependencies with Independence Atoms

Miika Hannula(✉) and Sebastian Link

Department of Computer Science, The University of Auckland, Auckland, New Zealand
{m.hannula,s.link}@auckland.ac.nz

Abstract. Infamously, the finite and unrestricted implication problems for the classes of (i) functional and inclusion dependencies together, and (ii) embedded multivalued dependencies alone are each undecidable. Famously, the restriction of (i) to functional and unary inclusion dependencies in combination with the restriction of (ii) to multivalued dependencies yield implication problems that are still different in the finite and unrestricted case, but each are finitely axiomatizable and decidable in low-degree polynomial time. An important embedded tractable fragment of embedded multivalued dependencies are independence atoms. These stipulate independence between two attribute sets in the sense that for every two tuples there is a third tuple that agrees with the first tuple on the first attribute set and with the second tuple on the second attribute set. Our main results show that finite and unrestricted implication deviate for the combined class of independence atoms, unary functional and unary inclusion dependencies, but both are axiomatizable and decidable in low-degree polynomial time. This combined class adds arbitrary independence atoms to unary keys and unary foreign keys, which frequently occur in practice as surrogate keys and references to them.

Keywords: Functional dependency · Inclusion dependency
Independence atom · Implication problem

1 Introduction

Databases represent information about some domain of the real world. For this purpose, data dependencies provide the main mechanism for enforcing the semantics of the given application domain within a database system. As such, data dependencies are essential for most data management tasks, including database design, query and update processing, as well as data cleaning, exchange, and integration. The usability of a class $\mathcal{C}$ of data dependencies for these tasks depends critically on the computational properties of its associated implication

The authors were supported by the Marsden Fund grant 3711702.

J. Pei et al. (Eds.): DASFAA 2018, LNCS 10828, pp. 353–369, 2018.
https://doi.org/10.1007/978-3-319-91458-9_21

problem. The implication problem for $\mathcal{C}$ is to decide whether for a given finite set $\Sigma \cup \{\varphi\}$ of data dependencies from $\mathcal{C}$, Σ implies φ, i.e. whether every database that satisfies all the elements of Σ also satisfies φ. If we require databases to be finite, then we speak of the finite implication problem, and otherwise of the unrestricted implication problem. While the importance of data dependencies continues to hold for new data models, the focus of this article is on the implication problems for important classes of data dependencies in the relational model of data. In this context, data dependency theory is deep and rich [37]. Our submission is from the area of database theory, on which DASFAA's call for paper has solicited original contributions.

Functional and inclusion dependencies constitute the most commonly used classes of data dependencies in practice. In particular, functional dependencies (FDs) are more expressive than keys, and inclusion dependencies (INDs) are more expressive than foreign keys, thereby capturing Codd's principles of entity and referential integrity, respectively, on the logical level. An FD $R : X \rightarrow Y$ with attribute subsets X, Y on relation schema R expresses that the values on attributes in Y are uniquely determined by the values on attributes in X. In particular, $R : X \rightarrow R$ expresses that X is a key for R. An inclusion dependency (IND) $R[A_1, \ldots, A_n] \subseteq R'[B_1, \ldots, B_n]$, with attribute sequences $A_1, \ldots, A_n$ on R and $B_1, \ldots, B_n$ on R', expresses that for each tuple t over R there is some tuple t' over R' such that for all $i = 1, \ldots, n$, $t(A_i) = t'(B_i)$ holds. If $n = 1$ we call the IND unary (UIND).

A fundamental result in dependency theory is that the unrestricted and finite implication problems for the combined class of FDs and INDs differ and each is undecidable [10,32,33]. Interestingly, for the expressive sub-class of FDs and UINDs, the unrestricted and finite implication problems still differ but each are axiomatizable and decidable in low-degree polynomial time [12].

Another important expressive class of data dependencies are *embedded multivalued dependencies* (EMVDs). An EMVD $R : X \rightarrow Y \perp Z$ with attribute subsets X, Y, Z of R expresses that the projection $r[XYZ]$ of a relation r over R on the set union XYZ is the join $r[XY] \bowtie r[XZ]$ of its projections on XY and XZ. Another fundamental result in dependency theory is that the unrestricted and finite implication problems for EMVDs differ, each is not finitely axiomatizable [36] and each is undecidable [23,24]. An important fragment of EMVDs are multivalued dependencies (MVDs), which are a class of full dependencies in which XYZ covers the full underlying set R of attributes. In fact, MVDs are the basis for Fagin's fourth normal form [13]. For the combined class of FDs, MVDs, and UINDs, finite implication is axiomatizable and decidable in cubic time, while unrestricted implication is axiomatizable and decidable in almost linear time [12,26].

Independence atoms (IAs) constitute an expressive subclass of EMVDs and FDs. An IA $X \perp Y$ with attribute subsets X, Y of R expresses that $X \cap Y$ is constant (i.e., the FD $\emptyset \rightarrow X \cap Y$ holds) and that the EMVD $\emptyset \rightarrow X \setminus Y \perp Y \setminus X$ holds. The latter expresses that the projection of a relation r on XY equals the cartesian product of its projections on X and Y, i.e., $r[XY] = r[X] \times r[Y]$.

For disjoint X and Y, the independence atoms $X \perp Y$ thus form a subclass of EMVDs. For the class of IAs, the finite and unrestricted implication problems coincide, they are finitely axiomatizable and decidable in low-degree polynomial time [27].

Given the usefulness of EMVDs, FDs, and INDs for data management, given their computational barriers, and given the attractiveness of IAs as a tractable fragment of EMVDs, it is a natural question to ask how IAs, FDs, and INDs interact. Our article helps address the current gap in the existing rich theory of relational data dependencies. Adding further to the challenge it is important to note that IAs still form an embedded fragment of EMVDs, in contrast to MVDs which are a class of full dependencies. Somewhat surprisingly, already the interaction of IAs with just keys is intricate [19,20]. For example, unrestricted implication is finitely axiomatizable but finite implication is not for keys and unary IAs (those with singleton attribute sets), while the finite and unrestricted implication problems coincide and enjoy a finite axiomatization for IAs and unary keys (those with a singleton attribute set). In contrast, the extension of INDs with IAs, although being more expressive than the class of INDs alone, does not add further complexity to the latter. For INDs and IAs taken together both implication problems still coincide and are finitely axiomatizable, PSPACE-complete, and fixed-parameter tractable in their arity [9,21].

Examples. We few examples will illustrate how knowledge about IAs advances data management. FDs and INDs do not require further motivation but the more we know about the interaction of IAs with FDs and INDs, the more we can advance data management.

Our first example is query processing. In particular, we show how the validity of independence atoms is intrinsically linked to the optimization of the famous division operator. The operator $\pi_{XY}(R) \div \pi_Y(R)$ returns all those X-values x such that for every Y-value y there is some tuple t with $t(X) = x$ and $t(Y) = y$ [11]. The ability of the division operator to express universal quantification makes it very powerful for expressing natural queries. The following result establishes the intrinsic link.

Theorem 1. *For all relations r over R, $\pi_{XY}(R)(r) \div \pi_Y(R)(r) = \pi_X(R)(r)$ if and only if r satisfies $X \perp Y$.*

Proof. The division operator is defined as follows:

$$\pi_{XY}(R)(r) \div \pi_Y(R)(r) = \pi_X(R)(r) - \pi_X((\pi_X(R)(r) \times \pi_Y(R)(r)) - \pi_{XY}(R)(r)),$$

and r satisfies $X \perp Y$ if and only if $\pi_X(R)(r) \times \pi_Y(R)(r) = \pi_{XY}(R)(r)$. The result follows directly.

In particular, the validity of an IA reduces the quadratic complexity of the division operator to a linear complexity of a simple projection [28]. The reduction in complexity also applies to the expression complexity of a query. Suppose we would like to return those entities x that occur together with all entities y (for

example, suppliers that supply all products), then we need to express the division operator $\pi_{X,Y}(R) \div \pi_Y(R)$ in SQL by double-negation as in:

```
SELECT R0.X FROM R AS R0
WHERE NOT EXISTS
  SELECT * FROM R AS R1
  WHERE NOT EXISTS
    SELECT * FROM R AS R2
    WHERE R2.X = R1.X AND
          R2.Y = R0.Y ;
```

where $R.X$ is short for $\bigwedge_{A \in X} R.A$, and $R2.X = R1.X$ ($R2.Y = R0.Y$) for $\bigwedge_{A \in X} R2.A = R1.A$ ($\bigwedge_{B \in Y} R2.B = R1.B$). However, if a query optimizer can notice that the IA $X \perp Y$ is implied by the enforced set Σ of constraints, then the query can be rewritten into

```
SELECT X
FROM R;
```

Our second example is database security. More specifically, the aim of inference control is to protect private data under inferences that clever attacks may use to circumvent access limitations [7]. For example, the combination of a particular patient name (say Jack) together with a particular medical examination (say angiogram) may be considered a secret, while access to the patient name and access to the medical examination in isolation may not be a secret. However, in some given context such as a procedure to diagnose some condition, all patients may need to undergo all examinations. That is, the information about the patient is independent of the information about the examination. Now, if the secret (Jack, angiogram) must not be revealed to an unauthorized user that can query the data source, then this user must not learn both: that Jack is a patient undergoing the diagnosis of the condition, and that angiogram is a medical examination that is part of the process for diagnosing the condition. Being able to understand the interaction of independence atoms with other database constraints can therefore help us to protect secrets under clever inference attacks.

Our final example is data profiling. Here we would like to demonstrate that independence atoms do occur in real-world data sets. For that purpose, we have mined some well-known publicly available data sets that have been used for the mining of other classes of data dependencies before [34]. We report the basic characteristics of these data sets in the form of their numbers of rows and columns, and list the number of maximal IAs and the maximum arity of those found. Here, an IA $X \perp Y$ is maximal in a given set of IAs if there is no other IA $V \perp W$ in the set such that $V \subseteq X$ and $W \subseteq Y$ holds. The arity of an IA is defined as the total number of attribute occurrences.

Data set	Number of columns	Number of rows	Number of IAs	Maximum arity
Bridges	13	108	4	3
Echocardiogram	13	132	5	4
Adult	14	48,842	9	3
Hepatitis	20	155	855	6
Horse	27	368	112	3

Table 1. Subclasses of FD+IND+IA. We write "ui" and "fi" for unrestricted and finite implication, respectively.

Class	ui = fi	Complexity: ui/fi	Finite axiomatization: ui/fi
FD	Yes [4]	Linear time [5]	Yes (2-ary) [4]
IND	Yes [9]	PSPACE-complete [9]	Yes (2-ary) [9]
IA	Yes [15,27,35]	Cubic time [15,27]	Yes (2-ary) [15,27,35]
IND+IA	Yes	PSPACE-complete [21]	Yes (3-ary) [21]
FD+IA, FD+UIA	No [20]	?/?	?/no
FD+IND	No [10,32]	Undecidable/undecidable [10,32]	No/no [10,32]
FD+UIND	No [12]	Cubic time/cubic time [12]	Yes/no (infinite) [12]
UFD+UIND	No [12]	Linear time/linear time [12]	Yes /no (infinite) [12]
UFD+UIND+IA	No	Cubic time/cubic time	Yes /no (infinite)

It should be stressed that the usefulness of these IAs is not restricted to those that are semantically meaningful. For example, the optimizations for the division operator also apply to IAs that "accidentally" hold on a given data set.

1.1 Contributions

In this article we make the following contributions.

(1) We illustrate the relevance of independence atoms for data management, such as their intrinsic link to the optimization of the division operator, more precise cardinality estimations for choosing better query plans, and database security. Moreover, we show that they occur in real-world data sets.
(2) For the combined class of FDs and IAs, finite and unrestricted implication differ [19,20]. We show that finite implication is not finitely axiomatizable, already for binary FDs (those with a two-element attribute set on the left-hand side) and unary IAs. For the combined class of IAs and unary FDs, we show that finite and unrestricted implication coincide and establish a finite axiomatization. Hence, the situation for the combined class of FDs and IAs is more intricate than for the combined class of FDs and MVDs, where finite and unrestricted implication coincide, which enjoy an elegant finite axiomatization [6], and for which implication can be decided in almost linear time [14].

(3) For the combined class of IAs, unary FDs, and UINDs, we establish axiomatizations for their finite and unrestricted implication problems, and show that both are decidable in low-degree polynomial time. This is analogous to the results for the combined class of FDs, MVDs, and UINDs. To the best of our knowledge, the class of IAs, unary FDs, and UINDs is only the second known class for which the finite and unrestricted implication differ but both are decidable in low-degree polynomial time. The class is practically relevant as it covers arbitrary IAs on top of unary keys and unary foreign keys, and already unary keys and unary foreign keys occur readily in practice [12]. The significant difference to FDs, MVDs, and UINDs is the more intricate interaction between FDs and IAs in comparison to FDs and MVDs. Note that unary FDs and INDs frequently occur in practice as surrogate keys and foreign keys that reference them. For example, 6 out of 8 keys are unary and 8 out of 9 foreign keys are unary in the TPC-H benchmark, while 20 out of 32 keys are unary and 44 and out 46 foreign keys are unary in the TPC-E benchmark[1]. The ability to reason efficiently about IAs, UFDs, and UINDs is good news for data management. Finally, trading in restrictions of the arity on INDs and FDs for restrictions on the arity of IAs cannot be successful: Finite implication for unary IAs and binary FDs is not finitely axiomatizable, see (2).
(4) For the combined class of IAs and FDs we establish tractable conditions sufficient for non-interaction in both the finite and unrestricted cases. Instances of the finite or unrestricted implication problems that meet the non-interaction conditions can therefore be decided efficiently by using already known algorithms for the sole class of IAs and the sole class of FDs. The decidability of the finite and unrestricted implication problems for IAs and FDs are both still open.

Organization. In Sect. 2 we present all the necessary definitions for the article. Section 3 addresses the combined class of FDs and IAs. In Sect. 4 we focus on the combination of UFDs, UINDs, and IAs, and establish axiomatizations for their finite and unrestricted implication problems. Section 5 identifies polynomial-time criteria for the non-interaction between INDs and IAs, and also between FDs and IAs. Finally, in Sect. 6 we discuss the computational complexity of the implication problems. Due to lack of space we refer the reader to Appendix for any remaining proofs. The appendix can be found in [22].

2 Preliminaries

We denote by $A, B, C, \ldots$ attributes and by $X, Y, Z, \ldots$ either sets or sequences of attributes. For two sets (sequences) X and Y, we write XY for their union (concatenation). Similarly, we may write A instead of the single element set or sequence that consists of A. The size of a set (or length of a sequence) X is written as $|X|$.

[1] http://www.tpc.org.

A *relation schema* consists of attributes A, each equipped with a set of *domain* values denoted by $\mathrm{Dom}(A)$. By *database schema* we denote a pairwise disjoint sequence of relations schemata. Given a relation schema R, a *tuple* over R is a function that maps each attribute A from R to $\mathrm{Dom}(A)$. A *relation* r over R is then a non-empty set of tuples over R, and a database d over $\mathcal{R} = (R_1, \ldots, R_n)$ is a sequence $(r_1, \ldots, r_n)$ where each r_i is a relation over R_i[2]. We sometimes write $r[R]$ to denote that r is a relation over R, and similarly we may write $d[\mathcal{R}]$. A relation is called *finite* if the underlying set of tuples is finite, and a database is finite if it is a sequence of finite relations. For a tuple t and a relation r over R and a subset (or subsequence) X of R, $t(X)$ is the *restriction* of t to X, and $r(X)$ is the set of all restrictions $t(X)$ where $t \in r$.

Next we define the syntax and semantics of functional and inclusion dependencies and independence atoms.

Functional Dependency. Let X and Y be two sets of attributes from a relation schema R. Then $R : X \to Y$ is a *functional dependency* that is satisfied by a database $d = (r[R])$ iff for all $t, t' \in r$, $t(X) = t'(X)$ implies $t(Y) = t'(Y)$.

Inclusion Dependency. Let $A_1, \ldots, A_n$ and $B_1, \ldots, B_n$ be two sequences of distinct attributes from relation schemata R and R', respectively. Then $R[A_1 \ldots A_n] \subseteq R'[B_1 \ldots B_n]$ is an *inclusion dependency* that is satisfied by a database $d = (r[R], r'[R'])$ iff for all $t \in r_i$ there is $t' \in r_j$ such that $t(A_1) = t'(B_1), \ldots, t(A_n) = t'(B_n)$.

Independence Atom. Let X and Y be two (not necessarily disjoint) attribute sets from a shared relation schema R. Then $R : X \perp Y$ is an *independence atom* that is satisfied by a database $d = (r[R])$ iff for all tuples $t, t \in r$ there is a tuple $t'' \in r$ such that $t''(X) = t(X)$ and $t''(Y) = t'(Y)$. A *disjoint* independence atom (DIA) is an IA $X \perp Y$ where $X \cap Y$ is empty.

Regarding all the aforementioned dependencies, if the relation schema R is not needed in the context, we will drop it from the syntax. E.g., we will write $X \perp Y$ instead of $R : X \perp Y$.

We say that an IND is *k-ary* if it is of the form $A_1 \ldots A_k \subseteq B_1 \ldots B_k$. An IA $X \perp Y$ and an FD $X \to Y$ are called k-ary if $\max\{|X|, |Y|\} = k$. A class of dependencies is called k-ary if it contains at most k-ary dependencies. Most of the subclasses that we consider are only unary, so we add "U" to a class name to denote its unary subclass. For instance, UIND denotes the class of all unary INDs. In general, for $k \geq 2$, we add "k" to a class name to denote its k-ary subclass. We use "+" to denote unions of classes, e.g., IND+IA denotes the class of all inclusion dependencies and independence atoms.

[2] We exclude empty relations from our definition. This is a practical assumption with no effect when single relation schemata are considered only. However, on multiple relations it has an effect, e.g., the rule $\mathcal{UI}3$ in Table 2 becomes unsound.

Notice that the semantic condition for IAs $X \perp Y$ holds only if the values of the common attributes of X and Y are constant. In other words, the following holds:

* $d \models R : X \perp X$, if for all $s, s' \in r$ it holds that $s(X) = s'(X)$.

Hence, we also call unary FDs of the form $\emptyset \rightarrow A$ and unary IAs of the form $A \perp A$ *constancy atoms* (CAs).

The *restriction* of a dependency σ to a set of attributes R, written $\sigma \restriction R$, is $X \cap R \rightarrow Y \cap R$ for an FD σ of the form $X \rightarrow Y$, and $X \cap R \perp Y \cap R$ for an IA σ of the form $X \perp Y$. If σ is an IND of the form $A_1 \dots A_n \subseteq B_1 \dots B_n$ and $i_1, \dots, i_k$ lists $\{i = 1, \dots, n : A_i \in R \text{ and } B_i \in R\}$, then $\sigma \restriction R = A_{i_1} \dots A_{i_k} \subseteq B_{i_1} \dots B_{i_k}$. For a set of dependencies Σ, the restriction of Σ to R, written $\Sigma \restriction R$, is the set of all $\sigma \restriction R$ for $\sigma \in \Sigma$. For attributes A and B from R, we denote by $\sigma(R : A \mapsto B)$ the dependencies obtained from σ by replacing any number of occurrences of A with B.

A set of axioms σ and rules of the form $\sigma_1, \dots, \sigma_n \Rightarrow \sigma$ is called an *axiomatization*. A rule is called n-ary if its antecedent part has n conjuncts. An axiomatization consisting of at most n-ary rules is called n-ary. A *deduction* from a set of dependencies Σ by an axiomatization $\mathfrak{R}$ is a sequence of dependencies $(\sigma_1, \dots, \sigma_n)$ where each σ_i is either an element of Σ, an axiom, or follows from $\sigma_1, \dots, \sigma_{i-1}$ by an application of a rule in $\mathfrak{R}$. In such an occasion we write $\Sigma \vdash_{\mathfrak{R}} \sigma$, or simply $\Sigma \vdash \sigma$ if $\mathfrak{R}$ is known.

Given a finite set of database dependencies $\Sigma \cup \{\sigma\}$, the (finite) unrestricted implication problem is to decide whether all (finite) databases that satisfy Σ also satisfy σ, written $\Sigma \models \sigma$ ($\Sigma \models_{\text{fin}} \sigma$). An axiomatization $\mathfrak{R}$ is *sound* for the unrestricted implication problem of a class of dependencies $\mathcal{C}$ if for all finite sets $\Sigma \cup \{\sigma\}$ of dependencies from $\mathcal{C}$, $\Sigma \vdash_{\mathfrak{R}} \sigma \Rightarrow \Sigma \models \sigma$; it is *complete* if $\Sigma \models \sigma \Rightarrow \Sigma \vdash_{\mathfrak{R}} \sigma$. Soundness and completeness for finite implication are defined analogously.

Some of our proofs use the chase algorithm that was invented in the late 70s [3,31]. For a detailed exposition of this technique we refer the reader to [2].

Axiomatizations. Tables 2 and 3 present the axiomatizations considered in this article. In Table 2, the axiomatization $\mathfrak{I} := \{\mathcal{I}1, \dots, \mathcal{I}5\}$ is sound and complete for independence atoms alone [20,27]. The rules $\mathcal{F}1, \mathcal{F}2, \mathcal{F}3$ form the Armstrong axiomatization for functional dependencies [4], and the rules $\mathcal{FI}1$ and $\mathcal{FI}2$ describe simple interaction between independence atoms and functional dependencies. Table 3 depicts the sound and complete axiomatization of inclusion dependencies introduced in [8,9]. Table 2 presents rules describing interaction between inclusion dependencies and independence atoms [21].

We leave it to the reader to check the soundness of the axiom systems in Tables 2 and 3. The proof does not include anything unexpected; we only note that soundness of $\mathcal{UI}3$ follows only if databases are not allowed to contain empty relations.

Theorem 2. *The axiomatization* $\mathfrak{A} \cup \mathfrak{B} \cup \mathfrak{C}$ *is sound for the unrestricted and finite implication problems of FD+IND+IA.*

Lastly, we note that, for notational clarity only, we will restrict attention to the uni-relational case in all our proofs. That is, we will consider only those cases where databases consist of a single relation.

3 IAs+FDs

First we consider the interaction between FDs and IAs. Already keys and IAs combined form a somewhat intricate class: Their finite and unrestricted implication problems differ and the former lacks a finite axiomatization [20]. In Sect. 3.1 we extend these results to the classes FD+IA and 2FD+UIA. However, the interaction between unary FDs and IAs is less involved. In Sect. 3.2 we show that for UFD+IA unrestricted and finite implication coincide and the axiomatization $\mathfrak{A}^*$ given in Table 2 forms a sound and complete axiomatization.

3.1 Implication Problem for FDs and IAs

The following theorem enables us to separate the finite and unrestricted implication problems for FD+IA as well as for FD+UIA.

Theorem 3 [19]. *The unrestricted and finite implication problems for keys and UIAs differ.*

Table 2. Axiomatizations $\mathfrak{A}$ for FDs and IAs and $\mathfrak{C}$ for IAs and INDs. We define $\mathfrak{I} := \{\mathcal{I}1, \ldots, \mathcal{I}5\}$ and $\mathfrak{A}^* := \mathfrak{A} \setminus \{\mathcal{I}5, \mathcal{F}3\}$.

$$\frac{}{\emptyset \perp X}$$
(trivial independence, $\mathcal{I}1$)

$$\frac{X \perp Y}{Y \perp X}$$
(symmetry, $\mathcal{I}2$)

$$\frac{X \perp YZ}{X \perp Y}$$
(decomposition, $\mathcal{I}3$)

$$\frac{X \perp Y \quad XY \perp Z}{X \perp YZ}$$
(exchange, $\mathcal{I}4$)

$$\frac{X \perp Y \quad Z \perp Z}{X \perp YZ}$$
(weak composition, $\mathcal{I}5$)

$$\frac{}{XY \to Y}$$
(reflexivity, $\mathcal{F}1$)

$$\frac{X \to Y \quad Y \to Z}{X \to Z}$$
(transitivity, $\mathcal{F}2$)

$$\frac{X \to Y}{XZ \to YZ}$$
(augmentation, $\mathcal{F}3$)

$$\frac{X \perp Y \quad X \to Y}{\emptyset \to Y}$$
(constancy, $\mathcal{FI}1$)

$$\frac{X \perp YZ \quad Z \to V}{X \perp YZV}$$
(composition, $\mathcal{FI}2$)

Axiomatization $\mathfrak{A}$

$$\frac{R[X] \subseteq R'[Z] \quad R[Y] \subseteq R'[W] \quad R'[Z \perp W]}{R[XY] \subseteq R'[ZW]}$$
(concatenation, $\mathcal{UI}1$)

$$\frac{R[XY] \subseteq R'[ZW] \quad R'[ZW] \subseteq R[XY] \quad R'[Z \perp W]}{R[X \perp Y]}$$
(transfer, $\mathcal{UI}2$)

$$\frac{R[X] \subseteq R'[Y] \quad R' : Y \perp Y}{R'[Y] \subseteq R[X]}$$
(symmetry, $\mathcal{UI}3$)

$$\frac{R[X] \subseteq R'[Y] \quad R' : Y \perp Y}{R : X \perp X}$$
(constancy, $\mathcal{UI}4$)

$$\frac{R[A] \subseteq R'[C] \quad R[B] \subseteq R'[C] \quad R' : C \perp C \quad \sigma}{\sigma(R : A \mapsto B)}$$
(equality, $\mathcal{UI}5$)

Axiomatization $\mathfrak{C}$

Table 3. Axiomatization $\mathfrak{B}$ for INDs

$\frac{}{R[X] \subseteq R[X]}$ (reflexivity, $\mathcal{U}1$)	$\frac{R[X] \subseteq R'[Y] \quad R'[Y] \subseteq R''[Z]}{R[X] \subseteq R''[Z]}$ (transitivity, $\mathcal{U}2$)	$\frac{R[A_1 \dots A_n] \subseteq R'[B_1 \dots B_n]}{R[A_{i_1} \dots A_{i_m}] \subseteq R'[B_{i_1} \dots B_{i_m}]}(^*)$ (projection and permutation, $\mathcal{U}3$)
		$(^*)$ i_j are pairwise distinct and from $\{1, \dots, n\}$

This theorem was proved by showing that $\Sigma \models_{\text{fin}} \sigma$ and $\Sigma \not\models \sigma$, for $\Sigma := \{A \perp B, C \perp D, BC \to AD, AD \to BC\}$ and $\sigma := AB \to CD$. In [19] it was shown that this counterexample can be extended to a non-axiomatizability results for finite implication of keys and IAs. By an analogous line of reasoning this results carries over to the class of FDs and IAs, as well (see Appendix).

Theorem 4. *The finite implication problem for FD+IA (2FD+UIA) is not finitely axiomatizable.*

The implicit assumption in the above theorem is that an axiomatization must be *attribute-bounded*, meaning that it may not introduce new attributes [10]. It is easy to see that with this prerequisite finite axiomatization entails decidability. Contrarily, there are finite axiomatizations for undecidable implication problems that do not adhere to this assumption [16–18,33].

To the best of our knowledge, decidability is open for both FD+IA and FD+UIA with respect to their finite and unrestricted implication problems. It is worth noting here that the unrestricted (finite) implication problem for FD+UIA is as hard as that for FD+IA. For this, we demonstrate a simple reduction from the latter to the former. Let $\Sigma \cup \{\sigma\}$ be a set of FDs and IAs, and let Σ' denote the set of FDs and IAs where each IA of the form $X \perp Y$ is replaced with dependencies from $\{A \perp B, X \to A, A \to X, Y \to B, B \to Y\}$ where A and B are fresh attributes. If σ is an FD, then Σ (finitely) implies σ iff Σ' (finitely) implies σ. Also, if σ is of the form $X \perp Y$, then we have $\Sigma \models \sigma$ iff $\Sigma'' \models \sigma'$, where

$$\Sigma'' := \Sigma' \cup \{X \to A, A \to X, Y \to B, B \to Y\},$$

$\sigma' := A \perp B$, and A and B are fresh attributes.

3.2 Implication for UFDs and IAs

Next we turn to the class UFD+IA. Extending the scope and methods from [20], which presented a finite axiomatization for unary keys and IAs, we show that the axiomatization $\mathfrak{A}^*$ (see Table 2) is sound and complete for UFD+IA in both with respect to finite and unrestricted implication. Hence, compared to UIAs and FDs, the interaction between IAs and UFDs is relatively tame. Combined, however, these two may entail new restrictions to column sizes. For instance, in the finite $A \to B_1$, $A \to B_2$, and $B_1 \perp B_2$ imply $|r(B_1)| \cdot |r(B_2)| \leq |r(A)|$. The proof of the following completeness theorem is obtained by a chase-based model construction (see Appendix).

Theorem 5. *The axiomatization* $\mathfrak{A}^*$ *is sound and complete for the unrestricted and finite implication problems of UFD+IA.*

As the same axiomatization characterizes both finite and unrestricted implication, we obtain the following corollary.

Corollary 1. *The finite and unrestricted implication problems coincide for UFD+IA.*

4 IAs+UFDs+UINDs

Next we turn attention to the combined class of FDs, INDs, and IAs. In the previous section we noticed that the finite implication problem for binary FDs and unary IAs is not finitely axiomatizable. On the other hand, both the finite and unrestricted implication problems for unary FDs and binary INDs are undecidable [32]. Hence, in this section we restrict to unary FDs and unary INDs, a class for which the two implication problems already deviate [12]. It turns out that the combination UFD+UIND+IA can be axiomatized with respect to both problems. However, in the finite case the axiomatization is infinite as one needs to add so-called cycle rules for UFDs and UINDs.

An axiomatization for unrestricted implication follows from results in Sect. 3.2 and [12]. For the proof, see Appendix.

Theorem 6. *The axiomatization* $\mathfrak{A}^* \cup \{\mathcal{U}1, \mathcal{U}2, \mathcal{UI}3, \mathcal{UI}4\}$ *is sound and complete for the unrestricted implication problem of UFD+UIND+IA.*

For finite implication a complete axiomatization of UFD+UIND+IA is found by extending $\mathfrak{A}^* \cup \{\mathcal{U}1, \mathcal{U}2, \mathcal{UI}3, \mathcal{UI}4\}$ with the so-called cycle rules [12] (see Table 2) and by removing $\mathcal{UI}3, \mathcal{UI}4$ which become redundant. However, the completeness proof is now more involved and proved in two steps. We will combine the chase-based approach of the proof of Theorem 5 with the graph-theoretic approach from [12]. The latter method was used to prove a complete axiomatization for the finite implication problem of UIND+FD. For the graph-theoretic approach, we commence by introducing multigraphs with two sorts of edges: red ones which encode UFDs and black ones which encode UINDs.

Table 4. Cycle rules for finite implication

$A_1 \to A_2$	$A_2 \supseteq A_3$	$\ldots$	$A_{2n-1} \to A_{2n}$	$A_{2n} \supseteq A_1$
$A_1 \leftarrow A_2$	$A_2 \subseteq A_3$	$\ldots$	$A_{2n-1} \leftarrow A_{2n}$	$A_{2n} \subseteq A_1$
(cycle rule for n, $\mathcal{C}_n$)				

Definition 1 [12]. *For each set* Σ *of UINDs and UFDs over* R*, let* $G(\Sigma)$ *be the multigraph that consists of nodes* R*, red directed edges* (A, B)*, for* $A \to B \in \Sigma$*, and black directed edges* (A, B)*, for* $B \subseteq A \in \Sigma$*. If* $G(\Sigma)$ *has red (black) directed edges from* A *to* B *and vice versa, then these edges are replaced with an undirected edge between* A *and* B*.*

Given a multigraph $G(\Sigma)$, we first topologically sort its strongly connected components which form a directed acyclic graph [25]. That is, each component is assigned a unique *scc-number*, greater than the scc-numbers of all its descendants. For an attribute A, denote by $\mathrm{scc}(A)$ the scc-number of the component node A belongs to. Note that $\mathrm{scc}(A) \geq \mathrm{scc}(B)$ if (A, B) is an edge in $G(\Sigma)$. Denote also by scc_i the set of attributes A with $\mathrm{scc}(A) = i$, and let $\mathrm{scc}_{\leq i} := \bigcup_{j \leq i} \mathrm{scc}_j$ and define $\mathrm{scc}_{\geq i}$, $\mathrm{scc}_{<i}$, and $\mathrm{scc}_{>i}$ analogously. The following lemma is a simple consequence of the definition.[3]

Lemma 1 [12]. *Let Σ be a set of UFDs and UINDs, closed under $\{\mathcal{F}1, \mathcal{F}2, \mathcal{U}1, \mathcal{U}2\} \cup \{\mathcal{C}_k : k \in \mathbb{N}\}$. Then every node in $G(\Sigma)$ has a red and a black self-loop. The red (black) subgraph of $G(\Sigma)$ is transitively closed. The subgraphs induced by the strongly connected components of $G(\Sigma)$ are undirected. In each strongly connected component, the red (black) subset of undirected edges forms a collection of node-disjoint cliques. Note that the red and black partitions of nodes could be different.*

We now apply this graphical approach to earlier techniques presented in this paper. Theorem 7 shows completeness of the axiomatization $\mathfrak{A}^* \cup \{\mathcal{U}1, \mathcal{U}2\} \cup \{\mathcal{C}_n : n \in \mathbb{N}\}$ for the finite implication problem of UFD+UIND+IA by using the relation generated in Lemma 2. The proof of this lemma describes an incremental modification of the base relation, taken from the proof of Theorem 5, that is shown to reflect a growing number of inclusion dependencies in its composition. This is achieved by an inductive re-organization of the column values according to the underlying scc-numbering while at the same time maintaining the integrity of the UFD and IA dependencies in the base relation. The proof of the theorem and the lemma can be found in Appendix.

Lemma 2. *Let Σ be a set of UFDs, UINDs, and IAs over R, partitioned respectively to Σ_{UFD}, Σ_{UIND}, and Σ_{IA}. Assume that $\Sigma_{\mathrm{UFD}} \cup \Sigma_{\mathrm{UIND}}$ contains all UFDs and UINDs derivable from Σ by $\mathfrak{A}^* \cup \{\mathcal{U}1, \mathcal{U}2\} \cup \{\mathcal{C}_k : k \in \mathbb{N}\}$, and assume that we have assigned an scc-numbering to $G(\Sigma_{\mathrm{UFD}} \cup \Sigma_{\mathrm{UIND}})$. Let E be either the empty set or a single attribute, and let $R' := \{B \in R : E \to B \notin \Sigma\}$. Then there exists a finite relation r and tuples $t_0, t_1 \in r$ such that:*

(i) $\Sigma \vdash X \perp Y$ if $X, Y \subseteq R'$ and for some $t \in r$, $t(X) = t_0(X)$ and $t(Y) = t_1(Y)$;
(ii) $r \models \Sigma_{\mathrm{UFD}} \cup \Sigma_{\mathrm{IA}}$;
(iii) $r(A)$ is (strictly) included in $r(B)$ if $\mathrm{scc}(A)$ is (strictly) less than $\mathrm{scc}(B)$.

Theorem 7. *The axiomatization $\mathfrak{A}^* \cup \{\mathcal{U}1, \mathcal{U}2\} \cup \{\mathcal{C}_n : n \in \mathbb{N}\}$ is sound and complete for the finite implication problem of UFD+UIND+IA.*

[3] Lemma 1 is a reformulation of Lemma 4.2. in [12] where the same claim is proved for a set of FDs and UINDs that is closed under $\{\mathcal{F}1, \mathcal{F}2, \mathcal{F}3, \mathcal{U}1, \mathcal{U}2\} \cup \{\mathcal{C}_k : k \in \mathbb{N}\}$. We may omit $\mathcal{F}3$ here since, when restricting attention to UFDs, $\mathcal{F}3$ is not needed in the proof.

5 Polynomial-Time Conditions for Non-interaction

The interaction-freeness between the class FD+IND has been well-studied in the literature [29,30]. Here, we examine the frontiers for tractable reasoning about the class FD+IA in both the finite and unrestricted cases. For IND+IA these questions have been studied in [21]. The idea is to establish sufficient criteria for the non-interaction between IAs and FDs. There is a trade-off between the simplicity and generality of such criteria. While simple criteria may be easier to apply, more general criteria allow us to establish non-interaction in more cases. Our focus here is on generality, and the criteria are driven by the corresponding inference rules. We define non-interaction between two classes as follows.

Definition 2. *Let Σ_0 and Σ_1 be two sets of dependencies from classes $\mathcal{C}_0$ and $\mathcal{C}_1$, respectively. We say that Σ_0, Σ_1 have* no interaction with respect to unrestricted (finite) implication *if*

- *for σ from $\mathcal{C}_0$, σ is (finitely) implied by Σ_0 iff σ is (finitely) implied by $\Sigma_0 \cup \Sigma_1$.*
- *for σ from $\mathcal{C}_1$, σ is (finitely) implied by Σ_1 iff σ is (finitely) implied by $\Sigma_0 \cup \Sigma_1$.*

Let us now define two syntactic criteria for describing non-interaction. We say that an IA $X \perp Y$ *splits* an FD $U \to V$ if both $(X \setminus Y) \cap U$ and $(Y \setminus X) \cap U$ are non-empty. An IA $X \perp Y$ splits an IND $Z \subseteq W$ if both $X \cap W$ and $Y \cap W$ are non-empty. Furthermore, $X \perp Y$ *intersects* $U \to V$ if $XY \cap U$ is non-empty. Notice that both these concepts give rise to possible interaction between two different classes. We show that lacking splits implies non-interaction for FD+IA in the unrestricted case. Non-interaction for FD+IA in the finite is guaranteed by the stronger condition defined in terms of lacking intersections.

For IND+IA lack of splits entail non-interaction [21].

Theorem 8 [21]. *Let Σ_{IND} and Σ_{IA} be respectively sets of INDs and IAs. If no IA in Σ_{IA} splits any IND in Σ_{IND}, then Σ_{IND} and Σ_{IA} have no interaction with respect to unrestricted (finite) implication.*

We proceed with the non-interaction results for FD+IA. The proofs are located in Appendix. For unrestricted implication the idea is to first apply the below polynomial-time algorithm which transforms an assumption set Σ to an equivalent set Σ^*. The set Σ^* is such that it has no interaction between FDs and IAs provided that none of its FDs split any IAs.

For a set of FDs Σ, let us denote by $\mathrm{Cl}(\Sigma, X)$ the closure set of all attributes A for which $\Sigma \models X \to A$. This set can be computed in linear time by the Beeri-Bernstein algorithm [5]. The non-interaction condition for unrestricted implication is now formulated using $\Sigma^*_{\mathrm{IA}} = \{X_1 \perp Y_1, \ldots, X_n \perp Y_n\}$ and $\Sigma^*_{\mathrm{FD}} = \Sigma_{\mathrm{FD}} \cup \{\emptyset \to Z\}$ where $Z, X_i Y_i$ are computed using the following algorithm that takes an FD set Σ_{FD} and an IA set $\Sigma_{\mathrm{IA}} = \{U_1 \perp V_1, \ldots, U_n \perp V_n\}$ as an input.

Algorithm 1. Algorithm for computing Z, X_i, Y_i

Require: Σ_{FD} and $\Sigma_{\mathrm{IA}} = \{U_i \perp V_i \mid i = 1, \ldots, n\}$
Ensure: Z and $\Sigma^*_{\mathrm{IA}} = \{X_i \perp Y_i \mid i = 1, \ldots, n\}$
1: **Initialize:** $V \leftarrow \emptyset, X_i \leftarrow U_i, Y_i \leftarrow V_i$
2: **repeat**
3: $Z \leftarrow V$
4: **for** $i = 1, \ldots, n$ **do**
5: $X_i \leftarrow \mathrm{Cl}(\Sigma_{\mathrm{FD}}, X_i V)$
6: $Y_i \leftarrow \mathrm{Cl}(\Sigma_{\mathrm{FD}}, Y_i V)$
7: $V \leftarrow V \cup (X_i \cap Y_i)$
8: **until** Z=V

From the construction we obtain that $\Sigma^*_{\mathrm{FD}} \cup \Sigma^*_{\mathrm{IA}}$ is equivalent to $\Sigma_{\mathrm{FD}} \cup \Sigma_{\mathrm{IA}}$ and that

(1) for $Z_1 \perp Z_2 \in \Sigma^*_{\mathrm{IA}}$ and $i = 1, 2$, $\Sigma^*_{\mathrm{FD}} \models Z_i \rightarrow X$ implies $X \subseteq Z_i$;
(2) $\Sigma^*_{\mathrm{FD}} \cup \Sigma^*_{\mathrm{IA}} \models \emptyset \rightarrow A$ iff $A \in Z$.

Recall that the closure set $\mathrm{C}(\Sigma_{\mathrm{FD}}, X)$ can be computed in linear time by the Beeri-Bernstein algorithm. Now, at stage 5 (or stage 6) the computation of the closure set is resumed whenever V introduces attributes that are new to X_i (Y_i). Since the number of the closures considered is $2|\Sigma_{\mathrm{IA}}|$, we obtain a quadratic time bound for the computation of Z, X_i, Y_i.

Theorem 9. *Let Σ_{FD} and Σ_{IA} be respectively sets of FDs and IAs, and let Σ^*_{FD} and Σ^*_{IA} be obtained from Σ_{FD} and Σ_{IA} by Algorithm 1. Then the following holds:*

- *if no IA in Σ^*_{IA} splits any FD in Σ^*_{FD}, then Σ^*_{FD} and Σ^*_{IA} have no interaction with respect to unrestricted implication;*
- *if no IA in Σ_{IA} intersects any FD in Σ_{FD}, then Σ_{FD} and Σ_{IA} have no interaction with respect to finite implication.*

To illustrate the necessity for a stronger condition in the finite case, recall from Sect. 3.1 that $AB \rightarrow CD$ is finitely implied by $\{A \perp B, C \perp D, BC \rightarrow AD, AD \rightarrow BC\}$, and notice that $AB \rightarrow CD$ is not finitely implied by $\{BC \rightarrow AD, AD \rightarrow BC\}$. However, Algorithm 1 does not produce any fresh assumptions, and neither $A \perp B$ nor $C \perp D$ splits any FD assumption. Therefore, lack of splits is not sufficient for non-interaction in the finite case.

6 Complexity Results

Next we examine the computational complexity of the discussed implication problems. We show that both implication problems for UFD+UIND+IA can be solved in low-degree polynomial time, even though the problems differ from one another. The associated decision procedures, found in Appendix, transform the implication problems first to graphs, as earlier in this paper, and subsequently

modify them according to appropriate inference rules. The only difference with finite implication is that an application of the cycle rules is included in the process. The implication problem then reduces, for UFDs and UINDs, to reachability in the graph, and for IAs, to an IA-implication instance which reflects the topology of the graph. Consequently, the stated time bounds follow.

Theorem 10. *Let $\Sigma_{\mathrm{UFD}}, \Sigma_{\mathrm{UIND}}, \Sigma_{\mathrm{IA}}$ be respectively sets of UFDs, UINDs, and IAs over a relation schema R. The unrestricted and finite implication problems for σ by $\Sigma_{\mathrm{UFD}} \cup \Sigma_{\mathrm{UIND}} \cup \Sigma_{\mathrm{IA}}$ can be decided in time:*

- $O(|\Sigma_{\mathrm{IA}}| \cdot |\Sigma_{\mathrm{UFD}}| + |\Sigma_{\mathrm{UIND}}|)$ *if σ is an UFD or UIND;*
- $O(|\Sigma_{\mathrm{IA}}| \cdot (|\Sigma_{\mathrm{UFD}}| + |R|^2) + |\Sigma_{\mathrm{UIND}}|)$ *if σ is a IA.*

7 Conclusion and Outlook

In view of the infeasibility of EMVDs and of FDs and INDs combined, the class of FDs, MVDs and unary INDs is important as it is low-degree PTIME decidable in the finite and unrestricted cases. As independence atoms form an important tractable embedded sub-class of EMVDs, we have delineated axiomatizability and tractability frontiers for sub-classes of FDs, INDs, and IAs. The most interesting class is that of IAs, unary FDs and unary INDs, for which finite and unrestricted implication differ but each is axiomatisable and decidable in low-degree polynomial time. The results form a basis for the advancement of several data processing tasks, including cardinality estimation, database security, and query optimization.

Even though research on dependency theory has been rich and deep, there are many problems that warrant future research. Theoretically, the decidability remains open for both independence atoms and functional dependencies as well as unary independence atoms and functional dependencies, both in the finite and unrestricted case. This line of research should also be investigated in the probabilistic setting of conditional independencies, fundamental to multivariate statistics and machine learning. Practically, implementations and experimental evaluations of the algorithms can complement the findings in the research. Of direct practical use for data profiling would be algorithms that compute the set of IAs that hold on a given relation, as would algorithms to mine notions of approximate IAs [1].

References

1. Abedjan, Z., Golab, L., Naumann, F.: Profiling relational data: a survey. VLDB J. **24**(4), 557–581 (2015)
2. Abiteboul, S., Hull, R., Vianu, V.: Foundations of Databases. Addison-Wesley, Boston (1995)
3. Aho, A.V., Beeri, C., Ullman, J.D.: The theory of joins in relational databases. ACM Trans. Database Syst. **4**(3), 297–314 (1979)

4. Armstrong, W.W.: Dependency structures of data base relationships. In: Proceedings of IFIP World Computer Congress, pp. 580–583 (1974)
5. Beeri, C., Bernstein, P.A.: Computational problems related to the design of normal form relational schemas. ACM Trans. Database Syst. **4**(1), 30–59 (1979)
6. Beeri, C., Fagin, R., Howard, J.H.: A complete axiomatization for functional and multivalued dependenciesin database relations. In: SIGMOD, pp. 47–61 (1977)
7. Biskup, J., Bonatti, P.A.: Controlled query evaluation for enforcing confidentiality in complete information systems. Int. J. Inf. Sec. **3**(1), 14–27 (2004)
8. Casanova, M.A., Fagin, R., Papadimitriou, C.H.: Inclusion dependencies and their interaction with functional dependencies. In: PODS, pp. 171–176 (1982)
9. Casanova, M.A., Fagin, R., Papadimitriou, C.H.: Inclusion dependencies and their interaction with functional dependencies. J. Comput. Syst. Sci. **28**(1), 29–59 (1984)
10. Chandra, A.K., Vardi, M.Y.: The implication problem for functional and inclusion dependencies is undecidable. SIAM J. Comput. **14**(3), 671–677 (1985)
11. Codd, E.F.: Relational completeness of data base sublanguages. In: Rustin, R. (ed.) Database Systems, pp. 65–98. Prentice Hall and IBM Research Report RJ 987, San Jose (1972)
12. Cosmadakis, S.S., Kanellakis, P.C., Vardi, M.Y.: Polynomial-time implication problems for unary inclusion dependencies. J. ACM **37**(1), 15–46 (1990)
13. Fagin, R.: Multivalued dependencies and a new normal form for relational databases. ACM Trans. Database Syst. **2**, 262–278 (1977)
14. Galil, Z.: An almost linear-time algorithm for computing a dependency basis in a relational database. J. ACM **29**(1), 96–102 (1982)
15. Geiger, D., Paz, A., Pearl, J.: Axioms and algorithms for inferences involving probabilistic independence. Inf. Comput. **91**(1), 128–141 (1991)
16. Hannula, M.: Reasoning about embedded dependencies using inclusion dependencies. In: LPAR-20, pp. 16–30 (2015)
17. Hannula, M., Kontinen, J.: A finite axiomatization of conditional independence and inclusion dependencies. In: FoIKS, pp. 211–229 (2014)
18. Hannula, M., Kontinen, J.: A finite axiomatization of conditional independence and inclusion dependencies. Inf. Comput. **249**, 121–137 (2016)
19. Hannula, M., Kontinen, J., Link, S.: On independence atoms and keys. In: CIKM, pp. 1229–1238 (2014)
20. Hannula, M., Kontinen, J., Link, S.: On the finite and general implication problems of independence atoms and keys. J. Comput. Syst. Sci. **82**(5), 856–877 (2016)
21. Hannula, M., Kontinen, J., Link, S.: On the interaction of inclusion dependencies with independence atoms. In: LPAR-21, pp. 212–226 (2017)
22. Hannula, M., Link, S.: On the interaction of functional and inclusion dependencies with independence atoms. Report CDMTCS-518. Centre for Discrete Mathematics and Theoretical Computer Science, University of Auckland, Auckland, New Zealand, February 2018
23. Herrmann, C.: On the undecidability of implications between embedded multivalued database dependencies. Inf. Comput. **122**(2), 221–235 (1995)
24. Herrmann, C.: Corrigendum to on the undecidability of implications between embedded multivalued database dependencies. Inf. Comput. **204**(12), 1847–1851 (2006)
25. Kahn, A.B.: Topological sorting of large networks. Commun. ACM **5**(11), 558–562 (1962)
26. Kanellakis, P.C.: Elements of relational database theory. In: Handbook of Theoretical Computer Science, pp. 1073–1156 (1990)

27. Kontinen, J., Link, S., Väänänen, J.A.: Independence in database relations. In: WoLLIC, pp. 179–193 (2013)
28. Leinders, D., Van den Bussche, J.: On the complexity of division and set joins in the relational algebra. In: PODS, pp. 76–83 (2005)
29. Levene, M., Loizou, G.: How to prevent interaction of functional and inclusion dependencies. Inf. Process. Lett. **71**(3–4), 115–125 (1999)
30. Levene, M., Loizou, G.: Guaranteeing no interaction between functional dependencies and tree-like inclusion dependencies. Theor. Comput. Sci. **254**(1–2), 683–690 (2001)
31. Maier, D., Mendelzon, A.O., Sagiv, Y.: Testing implications of data dependencies. ACM Trans. Database Syst. **4**(4), 455–469 (1979)
32. Mitchell, J.C.: The implication problem for functional and inclusion dependencies. Inf. Control **56**(3), 154–173 (1983)
33. Mitchell, J.C.: Inference rules for functional and inclusion dependencies. In: PODS, pp. 58–69 (1983)
34. Papenbrock, T., Ehrlich, J., Marten, J., Neubert, T., Rudolph, J.-P., Schönberg, M., Zwiener, J., Naumann, F.: Functional dependency discovery: an experimental evaluation of seven algorithms. PVLDB **8**(10), 1082–1093 (2015)
35. Paredaens, J.: The interaction of integrity constraints in an information system. J. Comput. Syst. Sci. **20**(3), 310–329 (1980)
36. Parker Jr., D.S., Parsaye-Ghomi, K.: Inferences involving embedded multivalued dependencies and transitive dependencies. In: SIGMOD, pp. 52–57 (1980)
37. Thalheim, B.: Dependencies in Relational Databases. Teubner, Stuttgart (1991)