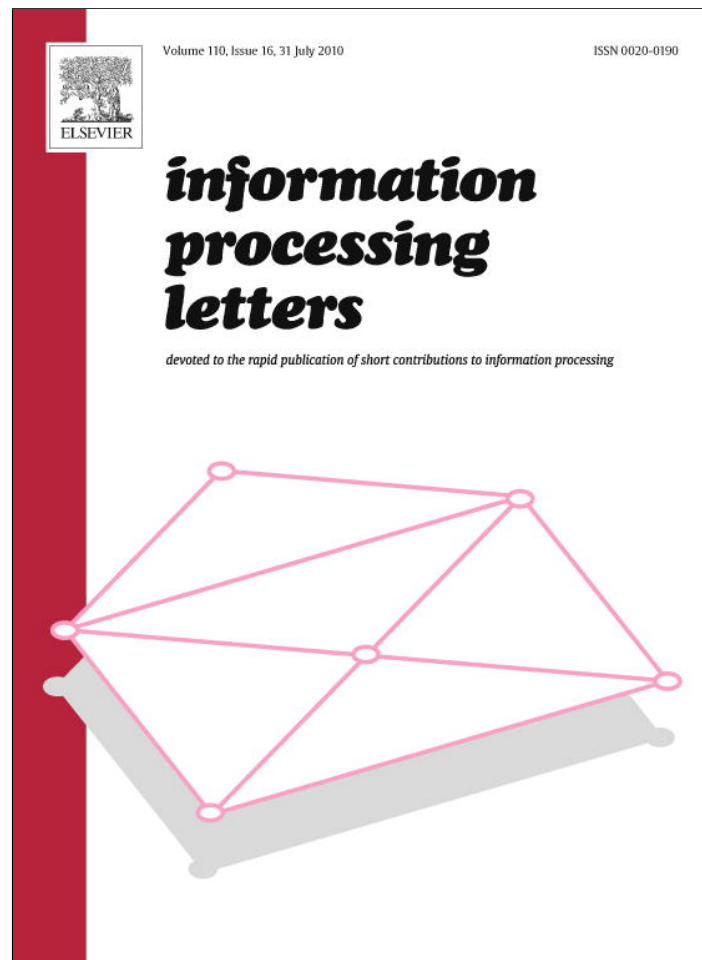


Provided for non-commercial research and education use.
Not for reproduction, distribution or commercial use.



This article appeared in a journal published by Elsevier. The attached copy is furnished to the author for internal non-commercial research and education use, including for instruction at the authors institution and sharing with colleagues.

Other uses, including reproduction and distribution, or selling or licensing copies, or posting to personal, institutional or third party websites are prohibited.

In most cases authors are permitted to post their version of the article (e.g. in Word or Tex form) to their personal website or institutional repository. Authors requiring further information regarding Elsevier's archiving and manuscript policies are encouraged to visit:

<http://www.elsevier.com/copyright>



Contents lists available at ScienceDirect

Information Processing Letters

www.elsevier.com/locate/ipl



Armstrong axioms and Boyce–Codd–Heath Normal Form under bag semantics

Henning Koehler^a, Sebastian Link^{b,*}

^a School of Information Technology & Electrical Engineering, The University of Queensland, Brisbane, Australia

^b School of Information Management, The Victoria University of Wellington, Wellington, New Zealand

ARTICLE INFO

Article history:

Received 21 December 2009

Received in revised form 29 May 2010

Accepted 4 June 2010

Available online 9 June 2010

Communicated by J. Chomicki

Keywords:

Database

Multiset

Key

Functional dependency

Armstrong axioms

Normal form

ABSTRACT

The theory of functional dependencies is based on relations, i.e. sets of tuples. Over relations, the class of functional dependencies subsumes the class of keys. Commercial database systems permit the storage of bags of tuples where duplicate tuples can occur. Over bags, keys and functional dependencies interact differently from how they interact over relations.

We establish finite ground axiomatizations of keys and functional dependencies over bags, and show a strong correspondence to goal and definite clauses in classical propositional logic. We define a syntactic Boyce–Codd–Heath Normal Form condition, and show that the condition characterizes schemata that will never have any redundant data value occurrences in their instances. The results close the gap between the existing set-based theory of data dependencies and database practice where bags are permitted.

© 2010 Elsevier B.V. All rights reserved.

1. Introduction

There is a gap between database theory and practice: all commercial relational database management systems allow duplicate tuples, and so relations in such systems are in effect bags. That is, duplicate tuples usually occur in real world database instances, even though the relational model of data is set-oriented, i.e., does not allow duplicate tuples to occur.

Indeed, databases and bags are tightly coupled. For instance, the cost of duplicate removal, e.g. after projections of relations or the union of several relations, is often prohibitively expensive. Therefore, duplicate removal is only performed if the user explicitly requests so. Furthermore, bag processing has applications in the evaluation of database queries, and in areas such as view maintenance, data warehousing and web information discovery. Consequently, there has been an effort in database research to

establish a foundation for handling bags instead of sets, cf. [1].

An extension of the relational framework of *data dependencies* to bags has not received any attention. This is somewhat surprising since already keys and functional dependencies interact differently over bags than they do over relations. A key over a schema S (i.e. S is a finite attribute set) is an expression $key(X)$ with an attribute subset X of S . An instance over S satisfies the key $key(X)$ if there are not any two distinct tuples in the instance that agree on all the attributes of X . A functional dependency (FD) over S is an expression of the form $X \rightarrow Y$ where X and Y are attribute subsets of S . An instance over S satisfies $X \rightarrow Y$ if all tuples of the instance that agree on all the attributes of X also agree on all the attributes of Y . The class of FDs subsumes the class of keys over instances that are relations. That is, if an instance over S is some relation, then it is not difficult to see that the relation satisfies the key $key(X)$ if and only if the relation satisfies the FD $X \rightarrow S$. This equivalence becomes invalid when bags are permitted as instances. Indeed, a bag may satisfy the functional dependency $X \rightarrow S$, but some distinct tuples of the

* Corresponding author.

E-mail addresses: henning@itee.uq.edu.au (H. Koehler), sebastian.link@vuw.ac.nz (S. Link).

bag may still agree on all the attributes in X , i.e., violate the key $key(X)$.

Example 1. Consider a database where we store customer orders, e.g. for Japanese cuisine. We record the *Item*, and the *Size* and *Price* of each item. An order may look as follows:

<i>Item</i>	<i>Size</i>	<i>Price</i>
takoyaki	large	¥90
takoyaki	large	¥90
okonomiyaki	small	¥300

This bag does not satisfy the key $key(Item, Size)$ but it does satisfy the functional dependency $Item, Size \rightarrow Item, Size, Price$.

Consequently, it is a natural question to ask how exactly keys and FDs interact when bags are permitted as database instances. Further questions arise, for example, about suitable normal form conditions that can be justified from a semantical point of view. These questions are significant since relational database management systems permit bags of tuples, but the underlying relational model of data only permits sets of tuples. Consequently, the current foundation for keys and FDs does not address how they interact in instances that occur in real database systems.

Contributions and organization. We define a relational model under bag semantics in Section 2. Two finite ground axiomatizations for the combined class of keys and FDs are established in Section 3. In Section 4 we show that the implication of keys and FDs corresponds to the implication of goal and definite clauses in Boolean propositional logic. We introduce a syntactic normal form condition on schemata that permit bags as instances in Section 5. It is shown that this condition characterizes those schemata in which all possible instances are free from redundant data value occurrences. We conclude in Section 6.

2. Preliminaries

A *bag schema* is a finite, non-empty set \mathfrak{B} of attributes. Every attribute A of a bag schema \mathfrak{B} is associated with a *domain* $dom(A)$, a countably infinite set of possible values. A *tuple* over \mathfrak{B} is a function $t : \mathfrak{B} \rightarrow \bigcup_{A \in \mathfrak{B}} dom(A)$ such that for all $A \in \mathfrak{B}$ we have that $t(A) \in dom(A)$ holds. For a subset $X \subseteq \mathfrak{B}$ the *projection of a tuple* t over \mathfrak{B} on X is the restriction $t(X)$ of t to X . For subsets X and Y of \mathfrak{B} we write XY for the set union $X \cup Y$. If $X = \{A_1, \dots, A_n\}$, then we may write $A_1 \dots A_n$ for X . In particular, we may write simply A to denote the singleton $\{A\}$. A *bag* over \mathfrak{B} is a finite multiset of tuples over \mathfrak{B} , usually denoted by \mathfrak{b} . We will use $\{\cdot\}$ to enumerate the elements of a bag explicitly, e.g. $\{t, t\}$ consists of two occurrences of the tuple t . A *relation* over \mathfrak{B} , usually denoted by r , is a finite set of tuples over \mathfrak{B} . In the context of a relation, we speak of a *relation schema*, denoted by R , rather than a bag schema. We will speak of a *schema* S if we refer to either a relation or a bag schema. We will speak of an *instance* if we refer to either a relation or a bag. Instances can be illustrated as

tables with each tuple of the instance corresponding to a row of the table. The attributes of the schema may be used as column headers.

Example 2. Consider the bag schema ORDER with attributes *Item*, *Size* and *Price* from Example 1. We may choose the same domain STRING for all the attributes. Quite naturally, duplicate tuples occur in bags over ORDER: some customer's order might consist of 2 large takoyaki for the price of ¥90 each, and a small okonomiyaki for the prize of ¥300, cf. Example 1. For example, the projection of the tuple (takoyaki, large, ¥90) on $\{Item, Price\}$ is (takoyaki, ¥90).

Integrity constraints restrict the set of possible database instances to those considered meaningful to the application at hand. One of the most fundamental classes of integrity constraints are keys and FDs. Keys allow us to uniquely identify tuples within an instance.

Definition 1. Let \mathfrak{B} be a bag schema. A *key* over \mathfrak{B} is an expression $key(X)$ where X is a subset of \mathfrak{B} . A bag \mathfrak{b} over \mathfrak{B} is said to *satisfy* the key $key(X)$ over \mathfrak{B} , denoted by $\models_{\mathfrak{b}} key(X)$, if and only if for every $t_1, t_2 \in \mathfrak{b}$ the following holds: if $t_1(X) = t_2(X)$, then $t_1 = t_2$.

According to Definition 1, a bag violates the key $key(X)$ if there are two distinct tuples in the bag that have equal values on all the attributes in X . In particular, a bag violates the empty key $key(\emptyset)$ if and only if there are at least two tuples in the bag. Since two distinct tuples of a bag can have equal values on all the attributes of the bag schema we require the two notions of tuple identity and value equality, respectively. In fact, while any relation over the relation schema R satisfies the key $key(R)$, proper bags (those that are not sets) do not satisfy any key over the associated bag schema.

Example 3. The bag

<i>Item</i>	<i>Size</i>	<i>Price</i>
okonomiyaki	average	¥450
okonomiyaki	average	¥450

consists of two distinct tuples which have equal values on all the attributes of the bag schema ORDER. Consequently, no key over ORDER is satisfied by this bag.

Interestingly, the semantics of XML keys is defined with respect to value equality and identity of nodes in XML trees [2,3].

In relational databases, i.e. for relations over relation schemata, the concept of a key is subsumed by the concept of a functional dependency.

Definition 2. Let \mathfrak{B} denote a bag schema. A *functional dependency* (FD) over \mathfrak{B} is an expression $X \rightarrow Y$ where $X, Y \subseteq \mathfrak{B}$. A bag \mathfrak{b} over \mathfrak{B} is said to *satisfy* the FD $X \rightarrow Y$ over \mathfrak{B} , denoted by $\models_{\mathfrak{b}} X \rightarrow Y$, if and only if for all $t_1, t_2 \in \mathfrak{b}$ the following holds: if $t_1(X) = t_2(X)$, then $t_1(Y) = t_2(Y)$.

Table 1
Axiomatizations of keys, and of FDs under set semantics.

$\frac{}{\text{key}(R)}$ (set axiom)	$\frac{\text{key}(X)}{\text{key}(XY)}$ (superkey)
Key axiomatization	

$\frac{}{XY \rightarrow X}$ (reflexivity)	$\frac{X \rightarrow Y}{X \rightarrow XY}$ (extension)	$\frac{X \rightarrow Y \quad Y \rightarrow Z}{X \rightarrow Z}$ (transitivity)
System \mathcal{D} : Armstrong axioms for FDs		

A functional dependency $X \rightarrow Y$ is said to be *trivial* if and only if $Y \subseteq X$; and *non-trivial* otherwise.

Example 4. Consider again the bag schema ORDER with attributes *Item*, *Size* and *Price*. Intuitively, the same item of the same size should always have the same price. This “business rule” can be formalized as the functional dependency

$Item, Size \rightarrow Price$.

Note that the bags in Examples 2 and 3 satisfy this functional dependency. Since every instance must satisfy this FD, bags such as

<i>Item</i>	<i>Size</i>	<i>Price</i>
okonomiyaki	average	¥450
okonomiyaki	average	¥500

are prohibited from entering the database. Note that it does not make sense to specify the functional dependency $Item \rightarrow Price$ since the same item may have different prices for different sizes. The challenge for the designer is to identify all keys and FDs that are meaningful to the application domain.

During the database design process one usually needs to determine further constraints which are implied by the given ones. In the following, \mathcal{C} denotes a class of integrity constraints, e.g. the combined class of keys and FDs.

Let S denote a schema, and let $\Sigma \cup \{\varphi\}$ be a set of integrity constraints of class \mathcal{C} over S . We say that Σ *implies* φ , denoted by $\Sigma \models \varphi$, if every instance over S that satisfies all $\sigma \in \Sigma$ also satisfies φ .

In order to determine the logical consequences of a set of keys or a set of FDs one can utilize a syntactic approach by applying inference rules, cf. Table 1. These *inference rules* have the form

$$\frac{\text{premise}}{\text{conclusion}}$$

and rules without any premises are called *axioms*.

Let $\Sigma \cup \{\varphi\}$ be a set of integrity constraints from a class \mathcal{C} , all defined over a schema S . We use \mathfrak{S} to denote a set of inference rules. A finite sequence $\gamma = [\varphi_1, \dots, \varphi_n]$ of constraints from \mathcal{C} is called an *inference from Σ* by \mathfrak{S} if

each φ_i is either an element of Σ or is obtained by applying one of the rules of \mathfrak{S} to appropriate elements of $\{\varphi_1, \dots, \varphi_{i-1}\}$. In this case, we say that γ infers φ_n , i.e. the last element of the sequence γ , and write $\Sigma \vdash_{\mathfrak{S}} \varphi_n$. Let $\Sigma_{\mathfrak{S}}^+ = \{\varphi \mid \Sigma \vdash_{\mathfrak{S}} \varphi\}$ denote the *syntactic closure* of Σ under inferences by \mathfrak{S} . An inference rule is called *sound* if the set of constraints in the premise of the rule implies the constraint in the conclusion. The set \mathfrak{S} is called *sound* for the implication of constraints in \mathcal{C} if for every schema S and for every set Σ of constraints in \mathcal{C} over S we have $\Sigma_{\mathfrak{S}}^+ \subseteq \Sigma^* = \{\varphi \mid \Sigma \models \varphi\}$. The set \mathfrak{S} is called *complete* for the implication of constraints in \mathcal{C} if for every schema S and for every set Σ of constraints in \mathcal{C} over S we have $\Sigma^* \subseteq \Sigma_{\mathfrak{S}}^+$. The (finite) set \mathfrak{S} is called a (finite) *axiomatization* for the implication of constraints in \mathcal{C} if it is both sound and complete for the implication of constraints in \mathcal{C} . The rules of Table 1 form finite axiomatizations for the implication of keys, and of FDs, respectively, over relations.

Let us fix a relation schema R . A key $key(X)$ over R implies the functional dependency $X \rightarrow R$ since in every relation over R that satisfies the key $key(X)$ there can never be two distinct tuples that agree on all attributes in X . Vice versa, the functional dependency $X \rightarrow R$ also implies the key $key(X)$ over R : in every relation over R no two distinct tuples can agree on all attributes in R , i.e., they must not agree on all attributes in X by means of the functional dependency $X \rightarrow R$. Hence, a key $key(X)$ is equivalent to a functional dependency $X \rightarrow R$ over the relation schema R . Consequently, FDs subsume keys over relations.

Let us now fix a bag schema \mathfrak{B} . A key $key(X)$ over \mathfrak{B} still implies the FD $X \rightarrow \mathfrak{B}$ over \mathfrak{B} . However, the reverse direction does not hold: take the bag $b = \{\{t, t\}\}$ where t denotes some tuple over \mathfrak{B} . Then for any $X \subseteq \mathfrak{B}$, b satisfies $X \rightarrow \mathfrak{B}$ yet violates $key(X)$, cf. Example 4.

Since instances that occur in real database systems are bags, and keys and FDs interact differently over bags than they do over relations, there is a practical interest to study the interaction of keys and FDs over bags.

3. Finite ground axiomatizations

In this section we establish an axiomatization for the implication of keys and FDs over bags.

Proposition 5. *Let \mathfrak{B} be a bag schema. For all FD sets Σ over \mathfrak{B} and for all attribute subsets X of \mathfrak{B} the FD set Σ does not imply the key $key(X)$.*

Proof. Let t be a tuple over \mathfrak{B} , and let $b := \{\{t, t\}\}$. For any FD set Σ over \mathfrak{B} and for any attribute subset X of \mathfrak{B} the bag b satisfies Σ and violates $key(X)$. \square

For a set Σ_{key} of keys over a bag schema \mathfrak{B} let

$$\Sigma_{\text{key}}^{\text{FD}} := \{X \rightarrow \mathfrak{B} \mid key(X) \in \Sigma_{\text{key}}\}$$

denote its corresponding set of FDs over \mathfrak{B} .

Lemma 6. *Let Σ_{key} be a set of keys over bag schema \mathfrak{B} , and let Σ_{FD} be a set of FDs over \mathfrak{B} . Then we have:*

Table 2
Finite ground axiomatizations of keys and FDs under bag semantics.

$\frac{\text{key}(X)}{\text{key}(XY)}$ (superkey)	$\frac{\text{key}(X)}{X \rightarrow \mathfrak{B}}$ (implication)	$\frac{\text{key}(\mathfrak{B}) \quad X \rightarrow \mathfrak{B}}{\text{key}(X)}$ (pullback)
$\frac{}{XY \rightarrow X}$ (reflexivity)	$\frac{X \rightarrow Y}{X \rightarrow XY}$ (extension)	$\frac{X \rightarrow Y \quad Y \rightarrow Z}{X \rightarrow Z}$ (transitivity)
\mathfrak{F} : Schema-dependent Armstrong axioms		

$\frac{\text{key}(X)}{X \rightarrow Y}$ (implication')	$\frac{\text{key}(Y) \quad X \rightarrow Y}{\text{key}(X)}$ (pullback')
$\frac{}{XY \rightarrow X}$ (reflexivity)	$\frac{X \rightarrow Y}{X \rightarrow XY}$ (extension)
$\frac{X \rightarrow Y \quad Y \rightarrow Z}{X \rightarrow Z}$ (transitivity)	
\mathfrak{F}' : Schema-independent Armstrong axioms	

1. If $\Sigma_{\text{key}} \cup \Sigma_{\text{FD}}$ implies the key $\text{key}(X)$, then $\Sigma_{\text{key}}^{\text{FD}} \cup \Sigma_{\text{FD}}$ implies the FD $X \rightarrow \mathfrak{B}$.
2. If $\Sigma_{\text{key}} \cup \Sigma_{\text{FD}}$ implies the FD $X \rightarrow Y$, then $\Sigma_{\text{key}}^{\text{FD}} \cup \Sigma_{\text{FD}}$ implies the FD $X \rightarrow Y$.

Proof. For condition 1 let \mathfrak{b} be a bag over \mathfrak{B} such that \mathfrak{b} satisfies $\Sigma_{\text{key}}^{\text{FD}} \cup \Sigma_{\text{FD}}$ and violates $X \rightarrow \mathfrak{B}$. Consequently, there are some tuples $t, t' \in \mathfrak{b}$ such that $t(X) = t'(X)$ and $t(\mathfrak{B}) \neq t'(\mathfrak{B})$. In particular, $t \neq t'$. It follows that $\{t, t'\}$ satisfies $\Sigma_{\text{key}}^{\text{FD}} \cup \Sigma_{\text{FD}}$ and violates $\text{key}(X)$. Suppose there is some key $\text{key}(Y) \in \Sigma_{\text{key}}$ such that $\{t, t'\}$ violates $\text{key}(Y)$. Then $t(Y) = t'(Y)$ and since $\{t, t'\}$ satisfies $Y \rightarrow \mathfrak{B} \in \Sigma_{\text{key}}^{\text{FD}}$ it follows that $t(\mathfrak{B}) = t'(\mathfrak{B})$. This is a contradiction. Consequently, $\{t, t'\}$ satisfies $\Sigma_{\text{key}} \cup \Sigma_{\text{FD}}$ and violates the key $\text{key}(X)$.

For condition 2 let \mathfrak{b} be a bag over \mathfrak{B} such that \mathfrak{b} satisfies $\Sigma_{\text{key}}^{\text{FD}} \cup \Sigma_{\text{FD}}$ and violates the FD $X \rightarrow Y$. Then there are some $t, t' \in \mathfrak{b}$ such that $t(X) = t'(X)$ and $t(Y) \neq t'(Y)$ hold. In particular, $t(\mathfrak{B}) \neq t'(\mathfrak{B})$. Suppose there is some key $\text{key}(Z) \in \Sigma_{\text{key}}$ such that $\{t, t'\}$ violates $\text{key}(Z)$. Then $t(Z) = t'(Z)$ and since $\{t, t'\}$ satisfies $Z \rightarrow \mathfrak{B} \in \Sigma_{\text{key}}^{\text{FD}}$ it follows that $t(\mathfrak{B}) = t'(\mathfrak{B})$. This is a contradiction. Consequently, $\{t, t'\}$ satisfies $\Sigma_{\text{key}} \cup \Sigma_{\text{FD}}$ and violates $X \rightarrow Y$. \square

The completeness proof of the following theorem uses the completeness of Armstrong's axioms \mathfrak{D} for the implication of FDs over relations and the additional inference rules required for the bag semantics.

Theorem 7. *The set \mathfrak{F} in Table 2 is a finite axiomatization for the implication of keys and FDs under bag semantics.*

Proof. For the soundness of \mathfrak{F} we first show the soundness of its inference rules. For the reflexivity axiom, and the extension and transitivity rules this can be done exactly as in the relational model of data [4]. For the soundness of the superkey rule we observe that if two tuples t, t' agree

on all attributes in XY , then they also agree on all the attributes in X . For the soundness of the implication rule we observe that a bag that satisfies the key $\text{key}(X)$ cannot have two tuples that have the same projection on the attribute set X , and consequently, the functional dependency $X \rightarrow \mathfrak{B}$ is satisfied, too. Finally, for the soundness of the pullback rule let \mathfrak{B} be an arbitrary bag schema, and let \mathfrak{b} be an arbitrary bag over \mathfrak{B} that violates the key $\text{key}(X)$. Then there are two distinct tuples $t, t' \in \mathfrak{b}$ that have the same projection on X . Assume that \mathfrak{b} also satisfies the functional dependency $X \rightarrow \mathfrak{B}$. Then t, t' also have the same projection on \mathfrak{B} . Consequently, \mathfrak{b} also violates the key $\text{key}(\mathfrak{B})$. The soundness of \mathfrak{F} follows by a simple induction over the length of an inference, using the soundness of the rules in \mathfrak{F} .

It remains to show the completeness of \mathfrak{F} . Let \mathfrak{B} be an arbitrary bag schema, let Σ_{key} be a set of keys over \mathfrak{B} , and let Σ_{FD} be a set of FDs over \mathfrak{B} . We need to show that whenever φ is a key or FD over \mathfrak{B} such that $\Sigma_{\text{key}} \cup \Sigma_{\text{FD}} \models \varphi$ holds, then $\Sigma_{\text{key}} \cup \Sigma_{\text{FD}} \vdash_{\mathfrak{F}} \varphi$ holds, too.

Suppose that φ denotes an FD $X \rightarrow Y$ over \mathfrak{B} that is implied by $\Sigma_{\text{key}} \cup \Sigma_{\text{FD}}$. Lemma 6 shows that $X \rightarrow Y$ is also implied by the FD set $\Sigma_{\text{key}}^{\text{FD}} \cup \Sigma_{\text{FD}}$. By the completeness of \mathfrak{D} for the implication of FDs it follows that $\Sigma_{\text{key}}^{\text{FD}} \cup \Sigma_{\text{FD}} \vdash_{\mathfrak{D}} X \rightarrow Y$ holds. Since \mathfrak{D} is a subset of \mathfrak{F} it follows that $\Sigma_{\text{key}}^{\text{FD}} \cup \Sigma_{\text{FD}} \vdash_{\mathfrak{F}} X \rightarrow Y$ holds. Moreover, for every $\varphi \in \Sigma_{\text{key}}^{\text{FD}}$ we have $\Sigma_{\text{key}} \vdash_{\mathfrak{F}} \varphi$ by an application of the *implication* rule. Hence, $\Sigma_{\text{key}} \cup \Sigma_{\text{FD}} \vdash_{\mathfrak{F}} X \rightarrow Y$ holds, too.

Suppose φ denotes a key $\text{key}(X)$ over \mathfrak{B} that is implied by $\Sigma_{\text{key}} \cup \Sigma_{\text{FD}}$. Proposition 5 shows that the set Σ_{key} is non-empty. Lemma 6 shows that the FD $X \rightarrow \mathfrak{B}$ is implied by the FD set $\Sigma_{\text{key}}^{\text{FD}} \cup \Sigma_{\text{FD}}$. By the completeness of \mathfrak{D} for the implication of FDs it follows that $\Sigma_{\text{key}}^{\text{FD}} \cup \Sigma_{\text{FD}} \vdash_{\mathfrak{D}} X \rightarrow \mathfrak{B}$ holds. As before, applications of the *implication* rule ensure that $\Sigma_{\text{key}} \cup \Sigma_{\text{FD}} \vdash_{\mathfrak{F}} X \rightarrow \mathfrak{B}$ holds. Moreover, since $\Sigma_{\text{key}} \neq \emptyset$ it follows by an application of the *superkey* rule that $\Sigma_{\text{key}} \vdash_{\mathfrak{F}} \text{key}(\mathfrak{B})$. An application of the *pullback* rule shows that $\Sigma_{\text{key}} \cup \Sigma_{\text{FD}} \vdash_{\mathfrak{F}} \text{key}(X)$. \square

The system \mathfrak{F} contains the *Armstrong axioms* of FDs [4], cf. Table 1, as one would expect from a more general framework. However, \mathfrak{F} does not contain the set axiom since it is not sound over bags. The system \mathfrak{F} is intuitive in the following sense. If any key is specified over \mathfrak{B} , then the superkey rule shows that \mathfrak{B} is a key, too. In this case, the implication and pullback rules show the equivalence between a key $\text{key}(X)$ and a functional dependency $X \rightarrow \mathfrak{B}$. Consequently, the system illustrates the interaction between keys and FDs over bags.

One may criticize \mathfrak{F} because it makes explicit reference to the underlying bag schema \mathfrak{B} , even though the definition of satisfaction of a key and functional dependency is only dependent on the attributes that actually occur in the data dependency, but is independent of the remaining attributes in the bag schema. Hence, one would expect that there is an axiomatization that does not make any reference to the bag schema. This critique of \mathfrak{F} is overcome by the system \mathfrak{F}' which is independent of the underlying bag schema. In this sense, it is a generalization of the Armstrong axioms from relational databases, cf. Table 1.

Theorem 8. *The set \mathfrak{F}' is a finite axiomatization for the implication of keys and FDs under bag semantics.*

Proof. For the soundness of \mathfrak{F}' we show that both the *implication'* rule and *pullback'* rule can be derived from the rules in \mathfrak{F} . The following inference

$$\frac{\frac{\text{key}(X)}{X \rightarrow \mathfrak{B}} \quad \overline{\mathfrak{B} \rightarrow Y}}{X \rightarrow Y}}$$

shows that the *implication'* rule can be derived from the *implication* rule, the *reflexivity* axiom and the *transitivity* rule in \mathfrak{F} . The following inference

$$\frac{\frac{\text{key}(Y)}{\text{key}(\mathfrak{B})} \quad \frac{X \rightarrow Y \quad \frac{\text{key}(Y)}{Y \rightarrow \mathfrak{B}}}{X \rightarrow \mathfrak{B}}}{\text{key}(X)}}$$

shows that the *pullback'* rule can be derived from the *implication* rule, the *transitivity* rule, the *superkey* rule and the *pullback* rule in \mathfrak{F} .

For the completeness of \mathfrak{F}' we first remark that the *implication* and *pullback* rules are special cases of the *implication'* and *pullback'* rules, respectively, where $Y = \mathfrak{B}$. Furthermore, the *superkey* rule can be derived from the *reflexivity* axiom and the *pullback'* rule:

$$\frac{\text{key}(X) \quad \overline{XY \rightarrow X}}{\text{key}(XY)}$$

Consequently, every key and FD that can be derived by \mathfrak{F} can also be derived by \mathfrak{F}' . The completeness of \mathfrak{F}' follows therefore from the completeness of \mathfrak{F} . \square

4. Equivalence to logic

In this section, every FD over \mathfrak{B} is of the form $X \rightarrow B$ where $B \in \mathfrak{B}$. This does not result in a loss of generality since an FD $X \rightarrow Y$ can be replaced by the set of FDs $X \rightarrow B$ for all $B \in Y$. Fagin's original equivalence between the implication of FDs and logical implication of Horn clauses in Boolean propositional logic [5] is also valid under bag semantics. We assume that the reader is familiar with Boolean propositional logic [6].

Under bag semantics the correspondences between FDs and Horn clauses become more refined: i) keys correspond to Horn clauses with no positive literals (*goal clauses*), and FDs correspond to Horn clauses with precisely one positive literal (*definite clauses*); and ii) the truth assignment that assigns true to all propositional variables can indeed be a counter-example for the implication of some Horn clause by a set of Horn clauses (for formulae resulting from FDs over relations this cannot be the case). The reason for ii) is that a bag with two duplicate tuples may also be a counter-example for the implication of a key φ from a set Σ of keys and FDs, e.g. over $\{Item, Size, Price\}$ where $\Sigma = \{Item, Size \rightarrow Price\}$ and $\varphi = key(Item, Size)$.

Let $\phi : \mathfrak{B} \rightarrow \mathcal{V}$ denote a bijection between the attributes of a bag schema \mathfrak{B} and the set \mathcal{V} of propositional variables. We will now extend this mapping ϕ

to a mapping Φ from keys and FDs over \mathfrak{B} to Horn formulae over \mathcal{V} . In fact, for a key $key(X)$ over \mathfrak{B} let $\Phi(key(X)) = \bigvee_{A \in X} \neg \phi(A)$, and for an FD $X \rightarrow B$ over \mathfrak{B} let $\Phi(X \rightarrow B) = (\bigvee_{A \in X} \neg \phi(A)) \vee \phi(B)$. In what follows, for a key or functional dependency σ we write σ' instead of $\Phi(\sigma)$, and for a finite set Σ of keys and FDs we write Σ' instead of $\{\sigma' \mid \sigma \in \Sigma\}$. Counter-example bags to the implication of keys and FDs are in a one-to-one correspondence to counter-example truth assignments to the logical implication of the associated Horn clauses. Indeed, the special truth assignment θ_b assigns *true* to the variable $V_A = \phi(A)$ precisely when the two tuples in a two-tuple bag b have matching values on the attribute A . For instance, the truth assignment θ that assigns *true* to all variables V_{Item}, V_{Size} and V_{Price} is a counter-example for the implication of the Horn clause $\neg V_{Item} \vee \neg V_{Size}$ by the Horn clause $\neg V_{Item} \vee \neg V_{Size} \vee V_{Price}$.

Theorem 9. *Let \mathfrak{B} be a bag schema, and let $\Sigma \cup \{\varphi\}$ denote a set of keys and FDs over \mathfrak{B} . Then Σ implies φ if and only if Σ' logically implies φ' .*

Proof. This is a special case of Theorem 15 for bags in [7]. \square

Remark 3. Fagin showed the correspondence between FDs and Horn clauses for relations [5]. This was extended to cover functional and multivalued dependencies, and also Boolean dependencies [7]. However, to obtain the equivalence between Boolean dependencies and propositional formulae, an additional formula is required in the set Σ' of premises of the implication problem. This formula excludes possible counter-example truth assignments that assign *true* to all variables, and thereby models set semantics [7,8]. It was also noted [7] that the additional formula is not required under bag semantics. Hence, the theorem in [7] established the correspondence between Boolean dependencies and propositional formulae under bag semantics. While Theorem 9 is subsumed by Theorem 15 in [7], a central observation is that unlike general Boolean dependencies, keys and FDs only map to goal and definite clauses. This guarantees that implication can be decided efficiently.

The size $|\varphi|$ of φ is the total number of attributes occurring in φ , and the size $\|\Sigma\|$ of Σ is the sum of $|\sigma|$ over all elements $\sigma \in \Sigma$. Theorem 9 conveniently provides us with an upper bound for the time-complexity of deciding implication for keys and FDs under bag semantics [9].

Theorem 10. *The problem whether a key or FD φ is implied by a set Σ of keys and FDs can be decided in $\mathcal{O}(\|\Sigma \cup \{\varphi\}\|)$ time.*

Boolean dependencies over \mathfrak{B} are generated by recursive applications of the Boolean connectives \neg, \wedge, \vee and \Rightarrow to the attributes of \mathfrak{B} . A Boolean dependency φ over \mathfrak{B} is satisfied by a bag b over \mathfrak{B} if every pair of (distinct) tuples t, t' in b satisfies the following: i) if $\varphi = A$, then $t(A) = t'(A)$, and in case ii) of nested Boolean dependencies we apply the usual semantics of the connectives [7].

As the implication of Boolean dependencies corresponds to the implication of formulae in Boolean propositional logic [7], the implication problem of Boolean dependencies in bags is coNP-complete. Hence, keys and FDs form an important linear-time decidable fragment.

5. Boyce–Codd–Heath Normal Form

Boyce and Codd [10] and Heath [11] introduced a normal form condition on relation schemata that characterizes the absence of certain processing difficulties with any relation over the schema. A schema S is said to be in Boyce–Codd–Heath Normal Form (BCHNF) with respect to a set Σ of keys and FDs over S if for every non-trivial FD $X \rightarrow Y \in \Sigma^*$ it is true that $X \rightarrow S \in \Sigma^*$ [10,11]. When S is a relation schema, i.e. when no duplicates are permitted to occur in any instance over S , then a key $key(X)$ over S is implied by Σ if and only if the FD $X \rightarrow S$ is implied by Σ . Consequently, a relation schema R is in BCHNF with respect to a set Σ of keys and FDs over R if and only if for every non-trivial FD $X \rightarrow Y \in \Sigma^*$ it is true that $key(X) \in \Sigma^*$. The situation is different when bags are permitted as database instances. If \mathfrak{B} is in BCHNF with respect to Σ , then there may still be some non-trivial FD $X \rightarrow Y \in \Sigma^*$ such that $key(X) \notin \Sigma^*$.

Example 5. Adopting the BCHNF condition [10,11] to bag semantics, the schema $ORDER = \{Item, Size, Prize\}$ is in BCHNF with respect to $\Sigma = \{Item, Size \rightarrow Prize\}$, but $key(Item, Size) \notin \Sigma^*$ as the bag

Item	Size	Price
okonomiyaki	average	¥450
okonomiyaki	average	¥450

demonstrates.

Example 5 illustrates that the BCHNF condition is not suitable to characterize the absence of data redundancy in bags of tuples. Intuitively, any of the two occurrences of the value ¥450 in the bag are redundant since every replacement of one of these values results in a bag that violates Σ . Following Vincent [12] we will make this notion of data redundancy explicit.

Definition 4. Let \mathfrak{B} be a bag schema, A an attribute of \mathfrak{B} , and t a tuple over \mathfrak{B} . A *replacement* of $t(A)$ is a tuple t' over \mathfrak{B} that satisfies the following conditions:

- for all $B \in \mathfrak{B} - \{A\}$ we have $t'(B) = t(B)$, and
- $t'(A) \neq t(A)$.

Intuitively, a data value occurrence in some Σ -satisfying instance is redundant if the occurrence cannot be replaced by any other data value without violating some constraint in Σ .

Definition 5. Let \mathfrak{B} be a bag schema, $A \in \mathfrak{B}$ an attribute, Σ a set of keys and FDs over \mathfrak{B} , b a bag over \mathfrak{B} that satisfies Σ , and t a tuple in b . We say that the data value

occurrence $t(A)$ is *redundant* if and only if every replacement t' of $t(A)$ results in a bag $b' := (b - \{t\}) \cup \{t'\}$ that violates Σ .

We say that \mathfrak{B} is in *Redundancy-Free Normal Form* (RFNF) with respect to Σ if and only if there is no bag b over \mathfrak{B} such that i) b satisfies Σ and ii) b contains a tuple t such that for some attribute A of \mathfrak{B} the data value occurrence $t(A)$ is redundant.

We will now give a definition of Boyce–Codd–Heath Normal Form that characterizes bag schemata over which every possible instance is free from redundant data value occurrences. Note that we apply our axiomatization \mathfrak{F} to define the BCHNF condition in purely syntactic terms.

Definition 6. Let \mathfrak{B} be a bag schema and Σ a set of keys and FDs over \mathfrak{B} . \mathfrak{B} is in *Boyce–Codd–Heath Normal Form* (BCHNF) with respect to Σ if and only if for every non-trivial FD $X \rightarrow Y \in \Sigma_{\mathfrak{F}}^+$ it is true that $key(X) \in \Sigma_{\mathfrak{F}}^+$.

We show that the syntactic BCHNF condition of Definition 6 captures the semantic RFNF condition of Definition 5. For the validity of this result we assume that the domains of all attributes contain a sufficient number of values, an assumption already necessary in the relational model of data [12].

Theorem 12. Let \mathfrak{B} be a bag schema and Σ a set of keys and FDs over \mathfrak{B} . Then \mathfrak{B} is in RFNF with respect to Σ if and only if \mathfrak{B} is in BCHNF with respect to Σ .

Proof. Let \mathfrak{B} not be in RFNF with respect to Σ . Then there is some bag b over \mathfrak{B} that satisfies Σ , some tuple $t \in b$ and some attribute $A \in \mathfrak{B}$ such that $t(A)$ is redundant. We need to show that there is some non-trivial FD $X \rightarrow Y \in \Sigma_{\mathfrak{F}}^+$ such that $key(X) \notin \Sigma_{\mathfrak{F}}^+$. Let $b[A] := \{\bar{t}(A) \mid \bar{t} \in b\}$. Define a replacement t' of $t(A)$ such that $t'(A) \in dom(A) - b[A]$. Furthermore, let $b' := (b - \{t\}) \cup \{t'\}$. Since $t(A)$ is redundant it follows that b' violates Σ . Since $\models_b \Sigma$ and b' agrees with b except on $t'(A) \notin b[A]$ it follows that b' cannot violate any key in Σ . Let b' violate the FD $X \rightarrow Y \in \Sigma$. From the definition of t' and the properties of b and b' it follows that $A \in Y - X$. Hence, $X \rightarrow Y$ is non-trivial. Since b' violates $X \rightarrow Y \in \Sigma$ there is some $t'' \in b' - \{t'\}$ such that $t''(X) = t'(X)$ and $t''(A) \neq t'(A)$. Moreover, $t'' \in b$, and $t''(X) = t(X)$ since $A \notin X$. Therefore, b satisfies Σ but b violates the key $key(X)$. Hence, $key(X) \notin \Sigma^*$ and by the soundness of \mathfrak{F} we conclude $key(X) \notin \Sigma_{\mathfrak{F}}^+$. It follows that \mathfrak{B} is not in BCHNF with respect to Σ .

Vice versa, let \mathfrak{B} not be in BCHNF with respect to Σ . Then there is some non-trivial FD $X \rightarrow Y \in \Sigma_{\mathfrak{F}}^+$ such that $key(X) \notin \Sigma_{\mathfrak{F}}^+$. We need to show that there is some bag b over \mathfrak{B} that satisfies Σ , some tuple $t \in b$ and some attribute $A \in \mathfrak{B}$ such that $t(A)$ is redundant. Let $b := \{\{t, t'\}\}$ consist of two tuples t and t' over \mathfrak{B} such that $t(X_{\Sigma}^+) = t'(X_{\Sigma}^+)$ and $t(B) \neq t'(B)$ hold for all $B \in \mathfrak{B} - X_{\Sigma}^+$ where

$$X_{\Sigma}^+ := \{C \in \mathfrak{B} \mid X \rightarrow C \in \Sigma_{\mathfrak{F}}^+\}.$$

We show that b satisfies Σ .

Let $U \rightarrow V \in \Sigma$ and let $t(U) = t'(U)$. It follows that $U \subseteq X_{\Sigma}^+$. From $X \rightarrow X_{\Sigma}^+ \in \Sigma_{\mathfrak{F}}^+$ and $X_{\Sigma}^+ \rightarrow U \in \Sigma_{\mathfrak{F}}^+$ we infer $X \rightarrow U \in \Sigma_{\mathfrak{F}}^+$; and from $X \rightarrow U \in \Sigma_{\mathfrak{F}}^+$ and $U \rightarrow V \in \Sigma$ we infer $X \rightarrow V \in \Sigma_{\mathfrak{F}}^+$, both times by means of the transitivity rule. Consequently, $V \subseteq X_{\Sigma}^+$ and therefore $t(V) = t'(V)$. We conclude that \mathfrak{b} satisfies $U \rightarrow V$.

Let $key(U) \in \Sigma$, and assume that $t(U) = t'(U)$ holds. We conclude that $U \subseteq X_{\Sigma}^+$. From $X \rightarrow X_{\Sigma}^+ \in \Sigma_{\mathfrak{F}}^+$ and $X_{\Sigma}^+ \rightarrow U \in \Sigma_{\mathfrak{F}}^+$ we infer $X \rightarrow U \in \Sigma_{\mathfrak{F}}^+$ by means of the transitivity rule. From $key(U) \in \Sigma$ and $X \rightarrow U \in \Sigma_{\mathfrak{F}}^+$ we infer that $key(X) \in \Sigma_{\mathfrak{F}}^+$ by an application of the pullback rule. This, however, is a contradiction since $key(X) \notin \Sigma_{\mathfrak{F}}^+$. Consequently, $t(U) \neq t'(U)$. This shows that \mathfrak{b} satisfies Σ .

Now let $A \in Y - X$. Since $Y \subseteq X_{\Sigma}^+$ it follows that $t(A)$ is redundant. Therefore, \mathfrak{B} is not in RFNF with respect to Σ . \square

The BCHNF condition entails that redundant data value occurrences in any instances over a schema with some non-trivial FD can only be avoided if some key is specified. If a key is specified, then all instances of the schema will be relations.

Definition 6 refers to the syntactic closure $\Sigma_{\mathfrak{F}}^+$ of Σ under \mathfrak{F} , which can be exponential in the size of Σ . Therefore, the question remains if the problem whether a bag schema is in BCHNF with respect to Σ can be decided efficiently.

Theorem 13. *Let \mathfrak{B} be a bag schema and let Σ be the union of a set Σ_{key} of keys and a set Σ_{FD} of FDs over \mathfrak{B} . Then \mathfrak{B} is in BCHNF with respect to Σ if and only if the following conditions hold:*

1. *if there is some non-trivial FD in Σ_{FD} , then $\Sigma_{key} \neq \emptyset$, and*
2. *for every non-trivial $X \rightarrow Y \in \Sigma_{FD}$ we have $X \rightarrow \mathfrak{B} \in \Sigma_{\mathfrak{F}}^+$.*

Proof. Assume that \mathfrak{B} is in BCHNF with respect to Σ . If every FD in Σ_{FD} is trivial, then conditions 1 and 2 are satisfied. Let $X \rightarrow Y \in \Sigma_{FD}$ be non-trivial. Since \mathfrak{B} is in BCHNF with respect to Σ it follows that $key(X) \in \Sigma_{\mathfrak{F}}^+$. Consequently, condition 1 is satisfied. By an application of the implication rule it follows that $X \rightarrow \mathfrak{B} \in \Sigma_{\mathfrak{F}}^+$. Hence, condition 2 is also satisfied.

Vice versa, assume that \mathfrak{B} is not in BCHNF with respect to Σ . That is, there is some non-trivial FD $X \rightarrow Y \in \Sigma_{\mathfrak{F}}^+$ such that $key(X) \notin \Sigma_{\mathfrak{F}}^+$. We need to show that there is some non-trivial FD $X' \rightarrow Y' \in \Sigma_{FD}$ and that $\Sigma_{key} = \emptyset$ or $X' \rightarrow \mathfrak{B} \notin \Sigma_{\mathfrak{F}}^+$. We first show that there is some non-trivial FD $X' \rightarrow Y' \in \Sigma_{FD}$ such that $key(X') \notin \Sigma_{\mathfrak{F}}^+$. Let

$$\Sigma = \Sigma_0 \subset \Sigma_1 \subset \dots \subset \Sigma_k = \Sigma_{\mathfrak{F}}^+$$

be a proper chain where for all $j = 1, \dots, k$ the set Σ_j results from Σ_{j-1} by an application of some inference rule in \mathfrak{F} . We show that if there is some non-trivial FD $X \rightarrow Y \in \Sigma_j$ such that $key(X) \notin \Sigma_{\mathfrak{F}}^+$, then there is some non-trivial FD $X' \rightarrow Y' \in \Sigma_{j-1}$ such that $key(X') \notin \Sigma_{\mathfrak{F}}^+$. For $j > 0$ let $X \rightarrow Y \in \Sigma_j - \Sigma_{j-1}$ be non-trivial such that $key(X) \notin \Sigma_{\mathfrak{F}}^+$. Then $X \rightarrow Y$ has been inferred either by

means of the extension or transitivity rule. In case of the extension rule we have $Y = XZ$ and $X \rightarrow Z \in \Sigma_{j-1}$ is non-trivial and $key(X) \notin \Sigma_{\mathfrak{F}}^+$. In case of the transitivity rule we know that there are $X \rightarrow Z$ and $Z \rightarrow Y$ in Σ_{j-1} . If $X \rightarrow Z$ is non-trivial, then we are done. If $X \rightarrow Z$ is trivial, then $Z \rightarrow Y$ is non-trivial since otherwise $X \rightarrow Y$ would be trivial, too. If $key(Z) \in \Sigma_{\mathfrak{F}}^+$, then $Z \rightarrow \mathfrak{B} \in \Sigma_{\mathfrak{F}}^+$ by means of the implication rule, and $key(\mathfrak{B}) \in \Sigma_{\mathfrak{F}}^+$ by means of the superkey rule. An application of the transitivity rule shows that $X \rightarrow \mathfrak{B} \in \Sigma_{\mathfrak{F}}^+$ holds as well. A final application of the pullback rule to $key(\mathfrak{B}) \in \Sigma_{\mathfrak{F}}^+$ and $X \rightarrow \mathfrak{B} \in \Sigma_{\mathfrak{F}}^+$ shows that $key(X) \in \Sigma_{\mathfrak{F}}^+$ holds as well. This is a contradiction, i.e., $key(Z) \notin \Sigma_{\mathfrak{F}}^+$. We have just shown that there is some non-trivial FD $X' \rightarrow Y' \in \Sigma_{FD}$ such that $key(X') \notin \Sigma_{\mathfrak{F}}^+$. It remains to show that $\Sigma_{key} = \emptyset$ or $X' \rightarrow \mathfrak{B} \notin \Sigma_{\mathfrak{F}}^+$.

Assume that $\Sigma_{key} \neq \emptyset$ and $X' \rightarrow \mathfrak{B} \in \Sigma_{\mathfrak{F}}^+$. Since $\Sigma_{key} \neq \emptyset$ it follows by an application of the superkey rule that $key(\mathfrak{B}) \in \Sigma_{\mathfrak{F}}^+$. Since $X' \rightarrow \mathfrak{B} \in \Sigma_{\mathfrak{F}}^+$ it follows by an application of the pullback rule that $key(X') \in \Sigma_{\mathfrak{F}}^+$, a contradiction. We conclude that there is some non-trivial FD $X' \rightarrow Y' \in \Sigma_{FD}$ such that $\Sigma_{key} = \emptyset$ or $X' \rightarrow \mathfrak{B} \notin \Sigma_{\mathfrak{F}}^+$. Consequently, condition 1 or condition 2 is violated. \square

The following result follows directly from Theorems 10 and 13.

Theorem 14. *The problem whether a bag schema \mathfrak{B} is in Boyce-Codd-Heath Normal Form with respect to a set $\Sigma = \Sigma_{key} \cup \Sigma_{FD}$ of keys and FDs over \mathfrak{B} can be decided in $\mathcal{O}(|\Sigma_{FD}| \times \|\Sigma\|)$ time.*

6. Conclusion and future work

We observed that keys and FDs interact differently over bags of tuples than they do over sets of tuples. Since commercial database systems permit the storage of bags of tuples, we extended some of the set-based theory of functional dependencies to bags. In particular, we established finite ground axiomatizations of keys and FDs, and showed that their implication is in strong correspondence with that of goal and definite clauses in Boolean propositional logic. Finally, we introduced a normal form condition that characterizes those schemata which do not permit any bags that contain any redundant data value occurrences. The results start to bridge the gap between the set-based theory of data dependencies and the reality of database instances in which duplicate tuples commonly occur.

A characterization of the interaction of XML keys [2,3, 13] and functional dependencies [14–16] is a challenging problem for future research. It would also be interesting to investigate the properties of Armstrong databases for data dependencies under bag semantics. These databases are useful for the acquisition of meaningful integrity constraints [17].

Acknowledgements

We would like to thank David Maier for informing us about the problem of the interaction of keys and FDs under bag semantics.

This research is supported by the Marsden Fund Council from Government funding, administered by the Royal Society of New Zealand.

References

- [1] G. Lamperti, M. Melchiori, M. Zanella, On multisets in database systems, in: Proceedings of WOMP, in: Lecture Notes in Computer Science, vol. 2235, 2000, pp. 147–216.
- [2] P. Buneman, S. Davidson, W. Fan, C. Hara, W. Tan, Keys for XML, *Computer Networks* 39 (5) (2002) 473–487.
- [3] S. Hartmann, S. Link, Efficient reasoning about a robust XML key fragment, *ACM Trans. Database Syst.* 34 (2) (2009).
- [4] W.W. Armstrong, Dependency structures of database relationships, *Information Processing* 74 (1974) 580–583.
- [5] R. Fagin, Functional dependencies in a relational data base and propositional logic, *IBM Journal of Research and Development* 21 (6) (1977) 543–544.
- [6] H. Enderton, *A Mathematical Introduction to Logic*, second ed., Academic Press, 2001.
- [7] Y. Sagiv, C. Delobel, D.S. Parker Jr., R. Fagin, Correction to “An equivalence between relational database dependencies and a fragment of propositional logic”, *J. ACM* 34 (4) (1987) 1016–1018.
- [8] J. Berman, W. Blok, Positive boolean dependencies, *Inform. Process. Lett.* 27 (1988) 147–150.
- [9] W. Dowling, J. Gallier, Linear-time algorithms for testing the satisfiability of propositional Horn formulae, *Journal of Logic Programming* 1 (3) (1984) 267–284.
- [10] E.F. Codd, Recent investigations in relational data base systems, in: IFIP Congress, 1974, pp. 1017–1021.
- [11] I.J. Heath, Unacceptable file operations in a relational data base, in: SIGFIDET Workshop, 1971, pp. 19–33.
- [12] M. Vincent, Semantic foundation of 4NF in relational database design, *Acta Inform.* 36 (1999) 1–41.
- [13] S. Hartmann, S. Link, Numerical constraints on XML data, *Inform. and Comput.* 208 (5) (2010) 521–544.
- [14] M. Arenas, L. Libkin, A normal form for XML documents, *ACM Trans. Database Syst.* 29 (1) (2004) 195–232.
- [15] S. Hartmann, M. Kirchberg, S. Link, A subgraph-based approach towards functional dependencies for XML, in: Proceedings of the 7th World Multiconference on Systemics, Cybernetics and Informatics (SCI), 2003, pp. 200–205.
- [16] L. Kot, W. White, Characterization of the interaction of XML functional dependencies with DTDs, in: Proceedings of ICDT, in: Lecture Notes in Computer Science, vol. 4353, 2007, pp. 119–133.
- [17] W.-D. Langeveldt, S. Link, Empirical evidence for the usefulness of Armstrong relations in the acquisition of meaningful functional dependencies, *Inf. Syst.* 35 (3) (2010) 352–374.