

Spoilt for Choice: Full First-Order Hierarchical Decompositions

Sebastian Link*

School of Information Management
Centre for Logic, Language and Computation
Victoria University of Wellington, New Zealand
`sebastian.link@vuw.ac.nz`

Abstract. Database design aims to find a database schema that permits the efficient processing of common types of queries and updates on future database instances. Full first-order decompositions constitute a large class of database constraints that can provide assistance to the database designer in identifying a suitable database schema.

We establish a finite axiomatisation of full first-order decompositions that reflects best database design practice: an inference engine derives all potential candidates of a database schema, but the final choice remains with the database designer.

Key words: Database Decomposition, Database Constraint, Axiomatisation, Propositional Logic

1 Introduction

Modern database management systems provide commensurate tools to store, manage and process different kinds of data. The core of these systems still relies on the sound technology that is based on the relational model of data [8]. From the perspective of finite model theory, a relational database is a finite structure over a relational signature [24]. Relations permit the storage of inconsistent data, i.e., data that violate conditions which every legal database instance ought to satisfy. Consequently, semantic constraints are specified by the database designer in order to restrict the databases to those which are considered meaningful to the application at hand. During database normalisation join-related constraints are explored to minimise data redundancy for efficient means of updating. In practice, most normalised schemata are subject to denormalisation in order to facilitate the efficient processing of the most common types of queries. The quality of the target database depends on the ability to reason correctly and appropriately about database constraints [14, 28].

Full first-order hierarchical decompositions (FOHDs) constitute a large class of relational constraints [9]. A relation is a model of an FOHD when it is the

* This research is supported by the Marsden fund council from Government funding, administered by the Royal Society of New Zealand

| Employee | Child | Insurance | Salary | Year |
|----------|-------|-----------|--------|------|
| Al | Bud | AMI | 52k | 2009 |
| Al | Kelly | AMI | 52k | 2009 |
| Al | Kelly | State | 52k | 2009 |
| Al | Bud | State | 52k | 2009 |

| Employee | Child |
|----------|-------|
| Al | Kelly |
| Al | Bud |

| Employee | Insurance | Salary | Year |
|----------|-----------|--------|------|
| Al | AMI | 52k | 2009 |
| Al | State | 52k | 2009 |

Table 1. A relation over WORK and two of its projections

natural join over at least two of its projections that all share the same join attributes. More precisely, a relation r satisfies the FOHD $X : [Y_1 \mid \dots \mid Y_k]$ when r is the natural join over the projections $r[XY_i]$ of r to the attribute sets XY_i for all $i = 1, \dots, k$.

Example 1. As a running example we consider the relation schema WORK comprising the attributes *Employee*, *Salary*, *Year*, *Insurance* and *Child*. Intuitively, any row over the table WORK collects information about an employee, the salary of this employee in a certain year, an insurance that the employee has taken out, and a child of the employee. It appears that the information on the child of any employee is independent of the information on the insurance, salary and year of the same employee. This separation of facts can be modelled by the FOHD $Employee : [Child \mid Insurance, Salary, Year]$. We may also choose to specify the FOHD $Employee : [Salary, Year \mid Child, Insurance]$ indicating that the information on the year's salary depends only on the employee independently of the employee's information on children and insurances. The WORK-relation in Table 1 satisfies the first FOHD since it is the natural join over its two projections on $\{Employee, Child\}$ and $\{Employee, Insurance, Salary, Year\}$. \square

Database constraints interact with one another. In fact, a constraint is implicitly specified if it is satisfied by any database that satisfies all the constraints that have been specified explicitly. A fundamental problem is the determination of such implicit knowledge. An axiomatisation for the implication of database constraints can form the basis of an enumeration algorithm that lists all consequences. In practice, such an enumeration is often desirable to validate explicit knowledge. For instance, the FOHD $Employee : [Child \mid Insurance \mid Salary, Year]$ is implied by the two FOHDs from Example 1.

FOHDs tell the database designer which attribute sets can form independent information units, e.g. $\{Employee, Child\}$ and $\{Employee, Salary, Year, Insurance\}$ based on the FOHD $Employee : [Child \mid Insurance, Salary, Year]$. Moreover, the original notion of an FOHD, as introduced by Delobel [9], also tells us in which order the separation of these information units takes place.

Example 2. The FOHD $Employee : [Child \mid Insurance \mid Salary, Year]$ instructs us to first decompose the relation schema WORK from Example 1 into $\{Employee, Child\}$ and $\{Employee, Insurance, Salary, Year\}$, and subsequently to decompose the latter schema into $\{Employee, Insurance\}$ and $\{Employee, Salary, Year\}$. \square

Intuitively, the decomposition can be processed in an arbitrary order and not just in the order that is indicated by a given FOHD. That is, for any permutation π the FOHD $X : [Y_{\pi(1)} \mid \cdots \mid Y_{\pi(k)}]$ is implied by the FOHD $X : [Y_1 \mid \cdots \mid Y_k]$. Syntactically, this can be represented as the inference rule

$$\frac{X : [Y_1 \mid \cdots \mid Y_k]}{X : [Y_{\pi(1)} \mid \cdots \mid Y_{\pi(k)}]}$$

known as the *permutation rule* [9, 29].

Example 3. Given the FOHD $Employee : [Child \mid Insurance \mid Salary, Year]$ we apply the permutation rule to infer $Employee : [Insurance \mid Salary, Year \mid Child]$. This FOHD instructs us to first decompose the relation schema WORK from Example 1 into $\{Employee, Insurance\}$ and $\{Employee, Salary, Year, Child\}$, and subsequently to decompose the latter schema into $\{Employee, Salary, Year\}$ and $\{Employee, Child\}$. \square

The final database schema of a decomposition is invariant under the order in which attribute sets on the right-hand side of an FOHD appear. In practice, however, it is rarely the case that the final database schema of a decomposition is chosen as the layout of the target database. The reason for this is that normalised database schemata only guarantee the efficient processing of updates [30]. Very commonly, the efficient processing of common queries outweighs the maintenance of database constraints and hence, databases are denormalised.

Example 4. Consider the two database schemata \mathcal{D}_1 :

$$\{Employee, Child\}, \{Employee, Insurance\}, \{Employee, Salary, Year\}$$

and $\mathcal{D}_2: \{Employee, Child, Insurance\}, \{Employee, Salary, Year\}$. The schema \mathcal{D}_1 provides a good choice when the most common types of queries do not require any joins between the three relation schemata and/or updates of either Child-, or Insurance- or Salary- and Year-information based on Employee-values are common. In that case no maintenance of FOHDs on any of the relation schemata is necessary. The second schema \mathcal{D}_2 may become preferable if common queries require a combination of Child- and Insurance-values, e.g. what is the most common insurance of employees that have at least two children? The efficient processing of these queries may outweigh the maintenance of the FOHD $Employee : [Child \mid Insurance]$ on $\{Employee, Child, Insurance\}$. \square

We note that denormalised database schemata occur in intermediate steps of the normalisation process, e.g. $\{Employee, Salary, Year, Child\}$ in Example 3. Such denormalised schemata may only be discovered due to the inference of an implicitly specified FOHD. In particular, the order in which the attribute sets occur on the right-hand side of an FOHD does seem to matter after all.

However, in database practice it is the database designer who chooses the final layout of the database. Consequently, the database designer chooses the order in which the information units are separated from one another. Hence, we

would like to gain complete knowledge about these information units, i.e., the minimal sets of attributes that are independent of one another.

Contributions. In this paper, we will develop a theory that reflects this common practice in database design. First, we develop a theory based on the original notion of an FOHD. We establish an axiomatisation in which it is always possible to delay an application of the permutation rule to the very last step of the inference. Consequently, the decision on the order in which the information units are separated from one another is delayed until the end of the design process (this reflects the choice of the database designer after they become aware of all possibilities for separating the information). If it was not always possible to shift the permutation rule to the very last step of an inference, then one may argue that database designers are limited in their choices for the final layout of the database. That is, the inference engine pre-determines an order in which the information units are to be separated, without any good reason.

Secondly, we introduce order-invariant hierarchical dependencies (OIHDs) which do not take the order of a decomposition into account. Intuitively, an OIHD $X : \{Y_1, \dots, Y_k\}$ represents the set

$$\{X : [Y_{\pi(1)} \mid \dots \mid Y_{\pi(k)}] : \pi \text{ is a permutation on } \{1, \dots, k\}\}$$

of FOHDs. The OIHD $Employee: \{\{Insurance\}, \{Salary, Year\}, \{Child\}\}$ represents the 6 different FOHDs where the left-hand side is *Employee* and the right-hand side is a permutation of $\{Insurance\}$, $\{Salary, Year\}$, and $\{Child\}$, for example. We establish an axiomatisation for the implication of OIHDs which contains exactly the inference rules adapted from our axiomatisation of FOHDs, but without the permutation rule. This is a further explanation of our intuition about the design process in practice: an inference engine mechanically determines those information units that can be separated, and, subsequently, the database designer decides how the actual separation will be implemented.

Finally, we show that correspondences between dependencies and fragments of propositional logic [19, 27] do not carry over to full-first order decompositions.

Organisation. We repeat concepts of the relational model of data in Section 2, in particular FOHDs, their semantic implication and syntactical inference. In Section 3 we establish an axiomatisation of FOHDs that reflects the role of the permutation role. We study order-invariant hierarchical dependencies in Section 4 and comment on their relationships to propositional logic in Section 5. We conclude in Section 6.

2 Full First-Order Decompositions

Let $\mathfrak{A} = \{A_1, A_2, \dots\}$ be a (countably) infinite set of symbols, called *attributes*. A *relation schema* is a finite set $R = \{A_1, \dots, A_n\}$ of attributes from \mathfrak{A} . Each attribute A of a relation schema is associated with a domain $dom(A)$ which represents the set of possible values that can occur in the column named A . If X and Y are sets of attributes, then we may write XY for $X \cup Y$. If $X = \{A_1, \dots, A_m\}$, then we may write $A_1 \dots A_m$ for X . In particular, we may write

simply A to represent the singleton $\{A\}$. A *tuple* over R (R -tuple or simply tuple, if R is understood) is a function $t : R \rightarrow \bigcup_{A \in R} \text{dom}(A)$ with $t(A) \in \text{dom}(A)$ for $i = 1, \dots, n$. For $X \subseteq R$ let $t[X]$ denote the restriction of the tuple t over R on X , and $\text{dom}(X) = \prod_{A \in X} \text{dom}(A)$ the Cartesian product of the domains of attributes in X . A *relation* r over R is a finite set of tuples over R . Let $r[X] = \{t[X] \mid t \in r\}$ denote the *projection* of the relation r over R on $X \subseteq R$. For $X, Y \subseteq R$, $r_1 \subseteq \text{dom}(X)$ and $r_2 \subseteq \text{dom}(Y)$ let $r_1 \bowtie r_2 = \{t \in \text{dom}(XY) \mid \exists t_1 \in r_1, t_2 \in r_2 (t[X] = t_1[X] \wedge t[Y] = t_2[Y])\}$ be the *natural join* of r_1 and r_2 .

Definition 1. A full first-order hierarchical decomposition over the relation schema R is an expression $X : [Y_1 \mid \dots \mid Y_k]$ with a non-negative integer k , $X, Y_1, \dots, Y_k \subseteq R$ such that Y_1, \dots, Y_k forms a partition of $R - X$. A relation r over R is said to satisfy (or said to be a model of) the full first-order hierarchical decomposition $X : [Y_1 \mid \dots \mid Y_k]$ over R , denoted by $\models_r X : [Y_1 \mid \dots \mid Y_k]$, if and only if $r = (\dots (r[XY_k] \bowtie r[XY_{k-1}]) \bowtie \dots) \bowtie r[XY_1]$ holds. \square

The FOHD $\emptyset : [Y_1 \mid \dots \mid Y_k]$ expresses the fact that any relation over R is the Cartesian product over its projections to attribute sets in $\{Y_i\}_{i=1}^k$. For $k = 0$, the FOHD $X : []$ is satisfied trivially, where $[]$ denotes the empty list.

Suppose we allow the sets Y_i to be empty. Then for all positive k we have the property that for all relations r the FOHD $X : [\emptyset, Y_2, \dots, Y_k]$ is satisfied by r if and only if r satisfies the FOHD $X : [Y_2, \dots, Y_k]$. In particular, if $k = 1$, then $X : [\emptyset]$ is equivalent to $X : []$; more specifically, they are satisfied by the same relations. One may now define an equivalence relation over the set of FOHDs defined on some fixed relation schema. Indeed, two such FOHDs are equivalent whenever they are satisfied by the same relations over the schema. Strictly speaking, we will apply inference rules to these equivalence classes of FOHDs. For the sake of simplicity, however, we have limited Definition 1 to those FOHDs where no empty sets are allowed to occur within the sequence of attribute sets. As the property above shows, this is not a real limitation but just a suitable choice of a representative from the equivalence classes.

For the design of a relational database schema semantic constraints are defined on the relations which are intended to be instances of the schema. During the design process one usually needs to determine further constraints which are logically implied by the given ones.

Definition 2. Let $\Sigma \cup \{\varphi\}$ be a set of constraints on the relation schema R . We say that Σ implies φ if and only if every relation r over R that satisfies all constraints in Σ also satisfies φ . \square

In order to determine implied FOHDs one can use the set of inference rules from Table 2 [9, 29]. These *inference rules* have the form

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

and inference rules without a premise are called *axioms*.

| | |
|--|---|
| $\frac{}{\emptyset : [R]}$ (universal, \mathcal{U}) | |
| $\frac{X : [Y_1 \mid \cdots \mid Y_k]}{XZ : [Y_1 - Z \mid \cdots \mid Y_k - Z]}$ (augmentation, \mathcal{A}) | $\frac{X : [X_1 \mid X_2] \quad XX_i : [Y_1 \mid \cdots \mid Y_k]}{X : [Y_1 \mid \cdots \mid Y_k \mid X_i]}$ (transitivity, \mathcal{T}) |
| $\frac{X : [Y_1 \mid \cdots \mid Y_k]}{X : [Y_1 \mid \cdots \mid Y_i Y_j \mid \cdots \mid Y_k]}$ (merging, \mathcal{M}) | $\frac{X : [Y_1 \mid \cdots \mid Y_k]}{X : [Y_{\pi(1)} \mid \cdots \mid Y_{\pi(k)}]}$ (permutation, \mathcal{P}) |

Table 2. Inference Rules for Full First-Order Hierarchical Decompositions

Let $\Sigma \cup \{\varphi\}$ be a set of semantic constraints from the class \mathcal{C} , all defined over the relation schema R . In this paper, the class \mathcal{C} may refer to full first-order hierarchical dependencies or order-invariant hierarchical dependencies. Let $\Sigma \vdash_{\mathfrak{S}} \varphi$ denote the inference of φ from a set Σ of constraints in \mathcal{C} by the set \mathfrak{S} of inference rules. Let $\Sigma_{\mathfrak{S}}^+ = \{\varphi \mid \Sigma \vdash_{\mathfrak{S}} \varphi\}$ denote the *closure* of Σ under inferences by \mathfrak{S} . The set \mathfrak{S} is called *sound* for the implication of constraints in \mathcal{C} if for every relation schema R and for every set Σ of constraints in \mathcal{C} over R we have $\Sigma_{\mathfrak{S}}^+ \subseteq \Sigma^* = \{\varphi \in \mathcal{C} \mid \Sigma \text{ implies } \varphi\}$. The set \mathfrak{S} is called *complete* for the implication of constraints in \mathcal{C} if for every relation schema R and for every set Σ of constraints in \mathcal{C} over R we have $\Sigma^* \subseteq \Sigma_{\mathfrak{S}}^+$. A complete set \mathfrak{S} is called *minimal* for the implication of constraints in \mathcal{C} if the removal of any inference rule from \mathfrak{S} results in a system that is incomplete for the implication of constraints in \mathcal{C} .

Note the following global condition that we enforce on all applications of inference rules that infer FOHDs. Whenever we apply such an inference rule, we remove all empty sets from the exact position in which they occur as elements in the sequence in the conclusion. For instance, we can infer $R : []$ by an application of the *augmentation rule* \mathcal{A} to the FOHD $\emptyset : [R]$. The *split rule*

$$\frac{X : [X_1 \mid X_2] \quad X : [Y_1 \mid \cdots \mid Y_k]}{X : [Y_1 \mid \cdots \mid Y_j \cap X_i \mid Y_j - X_i \mid \cdots \mid Y_k]}$$

is derivable from $\{\mathcal{A}, \mathcal{T}, \mathcal{M}, \mathcal{P}\}$

Theorem 1. *The set \mathfrak{F} of inference rules from Table 2 is sound, complete and minimal for the implication of full first-order decompositions.* \square

The completeness argument in the proof of Theorem 1 is similar to the completeness proof for multivalued dependencies [3, 5]. It relies on the notion of a *dependency basis*. Let $Dep_R(X)$ be the set of all $W \subseteq R - X$ for which some FOHD $X : [Y_1 \mid \cdots \mid Y_k]$ with $W \in \{Y_1, \dots, Y_k\}$ can be inferred from Σ by \mathfrak{F} . Due to our definition of FOHDs, we must explicitly enforce the empty set to be

included in $Dep_R(X)$ as well. Note that $Dep_R(X)$ is finite, and $(Dep_R(X), \subseteq, \cup, \cap, (\cdot)^c, \emptyset, R - X)$ constitutes a Boolean algebra due to the soundness of the merging and split rule. In particular, every finite Boolean algebra is atomic [16]. The set $DepB_R(X)$ of all atoms of $(Dep_R(X), \subseteq, \emptyset)$ is called the *dependency basis* of X with respect to Σ [2].

Even though \mathfrak{F} forms a minimal axiomatisation one may still simplify the inference rules in \mathfrak{F} . For example, the transitivity rule \mathcal{T}' :

$$\frac{X : [X_1 \mid X_2] \quad XX_1 : [Y_1 \mid \cdots \mid Y_k]}{X : [Y_1 \mid \cdots \mid Y_k \mid X_1]}$$

and the permutation rule \mathcal{P} allow us to infer the transitivity rule \mathcal{T} . Moreover, the merging rule \mathcal{M}' :

$$\frac{X : [Y_1 \mid \cdots \mid Y_k]}{X : [Y_1 Y_2 \mid \cdots \mid Y_k]}$$

and the permutation rule \mathcal{P} allow us to derive the merging rule \mathcal{M} .

Corollary 1. *The set $\mathfrak{F}' = \{\mathcal{U}, \mathcal{A}, \mathcal{T}', \mathcal{M}', \mathcal{P}\}$ is an axiomatisation for the implication of full first-order hierarchical decompositions. \square*

3 The Role of the Permutation Rule

We will show now that the axiomatisation \mathfrak{F} has the following property. For all relation schemata R , for all sets Σ of FOHDs on R , and for all inferences γ of an FOHD φ from Σ by \mathfrak{F} there is an inference ξ of φ from Σ by \mathfrak{F} in which the permutation rule is only applied in the very last step of ξ . Consequently, \mathfrak{F} soundly reflects the role of the permutation rule as a means for fixing a decomposition strategy, but not as a means to derive elements of the dependency basis.

Example 5. The FOHD $Employee:[Insurance|Salary, Year, Child]$ can be inferred by \mathfrak{F} from $Employee:[Salary, Year|Child, Insurance]$ and $Employee:[Child| Insurance, Salary, Year]$. Indeed, the inference

$$\frac{\frac{\frac{Employee : [Salary, Year \mid Child, Insurance]}{\mathcal{P} : Employee : [Child, Insurance \mid Salary, Year]}}{Employee : [Child \mid Insurance, Salary, Year]} \quad \mathcal{A} : Employee, Child : [Insurance \mid Salary, Year]}{\mathcal{T} : Employee : [Insurance \mid Salary, Year \mid Child]}}{\mathcal{M} : Employee : [Insurance \mid Salary, Year, Child]}$$

applies the permutation rule \mathcal{P} in an intermediate step. According to our reasoning, this inference is not adequate. In fact,

$$\frac{\frac{\frac{Employee : [Salary, Year \mid Child, Insurance]}{\mathcal{A} : Employee, Child : [Salary, Year \mid Insurance]}}{Employee : [Child \mid Insurance, Salary, Year]} \quad \mathcal{T} : Employee : [Salary, Year \mid Insurance \mid Child]}{\mathcal{M} : Employee : [Salary, Year, Child \mid Insurance]}}{\mathcal{P} : Employee : [Insurance \mid Salary, Year, Child]}$$

shows an inference by \mathfrak{F} which is indeed adequate. \square

The reasoning in Example 5 applies to arbitrary inferences by \mathfrak{F} .

Theorem 2. *Let R be some relation schema, and Σ a set of FOHDs over R . For each inference γ from Σ by \mathfrak{F} there is an inference ξ from Σ by \mathfrak{F} with the following properties:*

- γ and ξ infer the same full first-order hierarchical decomposition,
- in ξ the permutation rule \mathcal{P} is applied at the very last step only. \square

Theorem 2 says that the set consisting of the universal axiom \mathcal{U} , the augmentation rule \mathcal{A} , the transitivity rule \mathcal{T} and the merging rule \mathcal{M} is *almost* complete for the implication of FOHDs in the following sense.

Corollary 2. *Let R be some relation schema and Σ a set of full first-order hierarchical dependencies on R . Then for all $X : [Y_1 \mid \cdots \mid Y_k]$ on R we have: $X : [Y_1 \mid \cdots \mid Y_k] \in \Sigma_{\mathfrak{F}}^+$ if and only if there is some permutation π on $\{1, \dots, k\}$ such that $X : [Y_{\pi(1)} \mid \cdots \mid Y_{\pi(k)}] \in \Sigma_{\{\mathcal{U}, \mathcal{A}, \mathcal{T}, \mathcal{M}\}}^+$. \square*

Note that it is, by no means, self-evident that an axiomatisation of FOHDs has the property that applications of the permutation rule can always be deferred until the very last step of an inference. The axiomatisation \mathcal{F}' , for example, does not have this property.

4 Order-invariant Hierarchical Dependencies

We will now introduce and study order-invariant hierarchical dependencies. A single order-invariant hierarchical dependency $X : \{Y_1, \dots, Y_k\}$ is a compact representation of $k!$ many full first-order hierarchical decompositions.

Definition 3. *An order-invariant hierarchical dependency over relation schema R is an expression $X : \{Y_1, \dots, Y_k\}$ with non-negative integer k , $X, Y_1, \dots, Y_k \subseteq R$ such that Y_1, \dots, Y_k forms a partition of $R - X$. A relation r over R is said to satisfy the order-invariant hierarchical dependency $X : \{Y_1, \dots, Y_k\}$ on R , denoted by $\models_r X : \{Y_1, \dots, Y_k\}$, if and only if r is the natural join over $\{r[X Y_i]_{i=1}^k\}$, i.e., if $r = r[X Y_1] \bowtie \cdots \bowtie r[X Y_k]$ holds. \square*

Intuitively, the set-notation of OIHDs makes any application of the permutation rule unnecessary. The remarks regarding empty attribute subsets in FOHDs also apply to OIHDs. In particular, we apply the same global condition to inferences by the inference rules in Table 2.

Theorem 3. *The set \mathfrak{D} of inference rules from Table 3 is sound, complete and minimal for the implication of OIHDs. \square*

Corollary 2 enables us to reason about OIHDs by reasoning about FOHDs, and vice versa. If φ denotes the OIHD $X : \{Y_1, \dots, Y_k\}$, let $\bar{\varphi}$ denote the corresponding FOHD $X : [Y_1 \mid \cdots \mid Y_k]$, and let $\bar{\Sigma} = \{\bar{\sigma} \mid \sigma \in \Sigma\}$ denote the set of FOHDs that corresponds to the set Σ of OIHDs.

| | |
|---|---|
| $\frac{X : \{Y_1, \dots, Y_k\}}{XZ : \{Y_1 - Z, \dots, Y_k - Z\}}$ (augmentation, \mathcal{A}) | $\frac{X : \{X_1, X_2\} \quad XX_i : \{Y_1, \dots, Y_k\}}{X : \{Y_1, \dots, Y_k, X_i\}}$ (transitivity, \mathcal{T}) |
| $\frac{}{\emptyset : \{R\}}$ (universal axiom, \mathcal{U}) | $\frac{X : \{Y_1, \dots, Y_k\}}{X : \{Y_1, \dots, Y_i Y_j, \dots, Y_k\}}$ (merging, \mathcal{M}) |

Table 3. Inference Rules for Order-Invariant Hierarchical Dependencies

Corollary 3. *Let R be some relation schema, Σ a set of OIHDs and $\bar{\Sigma}$ the corresponding set of FOHDs over R . Then for all OIHDs φ on R we have: $\varphi \in \Sigma_{\mathcal{D}}^+$ if and only if $\bar{\varphi} \in \bar{\Sigma}_{\mathcal{D}}^+$. Moreover, for all FOHDs $\bar{\varphi}$ on R we have: $\bar{\varphi} \in \bar{\Sigma}_{\mathcal{D}}^+$ if and only if $\varphi \in \Sigma_{\mathcal{D}}^+$. \square*

5 Logic and Data Dependencies

Essentially, data dependencies are certain first-order formulae [13]. For instance, the OIHD $A : \{B, C, D\}$ over the relation schema R can be expressed by

$$\forall a, b, c, d, b', c', d', b'', c'', d'' \\ ((R(a, b, c, d) \wedge R(a, b', c', d') \wedge R(a, b'', c'', d'')) \Rightarrow R(a, b, c', d'')).$$

Binary OIHD $X : \{Y_1, Y_2\}$ are known as *multivalued dependencies* [12, 5]. The implication of multivalued dependencies (MVDs) is even in one-to-one correspondence with fragments of propositional logic [10, 11, 27].

For a relation schema R let $\mathcal{V}_R = \{V_A : A \in R\}$ denote its corresponding set of propositional variables. For an OIHD $X : \{Y_1, \dots, Y_k\}$ on R , denoted by φ , let φ' denote the following propositional formulae over \mathcal{V}_R :

$$\left(\bigwedge_{A \in X} V_A \right) \Rightarrow \left(\bigvee_{i=1}^k \left(\bigwedge_{B \in Y_i} V_B \right) \right). \quad (1)$$

For a set Σ of OIHDs over R let Σ' denote the set $\{\sigma' : \sigma \in \Sigma\}$ of propositional formulae over \mathcal{V}_R . The following theorem holds for MVDs [27].

Theorem 4. [27] *Let R be some relation schema, and let $\Sigma \cup \{\varphi\}$ be a set of MVDs over R . Then $\Sigma \models \varphi$ if and only if $\Sigma' \models \varphi'$. \square*

However, for the class of OIHDs an extension of the correspondence to the fragment of formulae defined by Equation (1) fails. In fact, let us consider the OIHD $\sigma = A : \{\{B, C\}, \{D\}\}$ and the OIHD $\varphi = A : \{\{B\}, \{C\}, \{D\}\}$. We observe that σ does not imply φ as the two tuple relation $\{(a, b, c, d), (a, b', c', d)\}$ over $R = ABCD$ shows. However, $\sigma' = V_A \Rightarrow ((V_B \wedge V_C) \vee V_D)$ does logically imply $\varphi' = V_A \Rightarrow (V_B \vee V_C \vee V_D)$. Hence, the correspondence fails.

We will now briefly discuss which dependencies do correspond to the fragment of propositional logic defined by Equation (1). These are *degenerated OIHDs*, a subclass of Boolean dependencies [27]. The syntax of a *degenerated OIHD* is that of an OIHD. A relation r over R is said to *satisfy* the degenerated OIHD $X : \{Y_1, \dots, Y_k\}$ over R if for all tuples $t_1, t_2 \in r$ the following holds: if $t_1[X] = t_2[X]$, then there is some $i \in \{1, \dots, k\}$ such that $t_1[Y_i] = t_2[Y_i]$ holds. This is the same as saying that for all tuples $t_1, t_2 \in r$ it is true that for some $i \in \{1, \dots, k\}$ the two-tuple relation $\{t_1, t_2\}$ satisfies the functional dependency $X \rightarrow Y_i$.

A consequence of Theorem 4 is that the implication of binary OIHDs corresponds exactly to the implication of degenerated OIHDs (i.e. degenerated multivalued dependencies) [27]. In general, however, the implication of OIHDs is not equivalent to the implication of degenerated OIHDs. For instance, if we view the OIHDs σ and φ as degenerated OIHDs, then σ does imply φ . Hence, σ implies φ when viewed as degenerated OIHDs, but σ does not imply φ when viewed as OIHD. Vice versa, φ implies σ when viewed as OIHDs, but φ does not imply σ when viewed as degenerated OIHDs.

6 Conclusion

We have established an axiomatisation of full first-order hierarchical decompositions that reflects the role of the permutation rule as a mere means to fix the order in which a database schema is decomposed. Hence, the permutation rule does not have any impact on the set of minimal units into which a database schema can be separated, i.e., the elements of the dependency basis. Indeed, our axiomatisation allows applications of the permutation rule to be delayed until the very last step of an inference. Removing the permutation rule from the axiomatisation results in a set of inference rules that is sound and complete for the implication of order-invariant hierarchical dependencies.

The results of this paper are complementary to Biskup's results on the role of the complementation rule in the context of multivalued dependencies [4]. Indeed, there are axiomatisations of MVDs in which applications of the complementation rule could be avoided completely or deferred until the very last step of an inference [4, 25]. Consequently, the complementation rule is a mere means to achieve database normalisation. These results were extended to the context of partial database relations [26], full hierarchical dependencies [22], functional and multivalued dependencies [5].

The eXtensible Markup Language (XML) [6] offers a flexible way to store data. As such, it has received considerable interest from the database community. One challenging area of XML research addresses constraints, which provide effective means to capture important semantic information about XML data [15]. So far, the interest in XML constraints has mainly addressed keys [7, 18, 20, 21] and functional dependencies [1, 17, 23, 31, 32]. As far as the author is aware, the decomposition of XML data based on integrity constraints has not been studied.

References

1. M. Arenas and L. Libkin. A normal form for XML documents. *ACM Trans. Database Syst.*, 29(1):195–232, 2004.
2. C. Beeri. On the membership problem for functional and multivalued dependencies in relational databases. *ACM Trans. Database Syst.*, 5(3):241–259, 1980.
3. C. Beeri, R. Fagin, and J. H. Howard. A complete axiomatization for functional and multivalued dependencies in database relations. In *SIGMOD*, pages 47–61. ACM, 1977.
4. J. Biskup. Inferences of multivalued dependencies in fixed and undetermined universes. *Theor. Comput. Sci.*, 10(1):93–106, 1980.
5. J. Biskup and S. Link. Appropriate reasoning about data dependencies in fixed and undetermined universes. In *FoIKS*, number 4932 in Lecture Notes in Computer Science, pages 58–77, 2008.
6. T. Bray, J. Paoli, C. M. Sperberg-McQueen, E. Maler, and F. Yergeau. Extensible markup language (XML) 1.0 (fourth edition) W3C recommendation. <http://www.w3.org/TR/xml>, 2006.
7. P. Buneman, S. Davidson, W. Fan, C. Hara, and W. Tan. Keys for XML. *Computer Networks*, 39(5):473–487, 2002.
8. E. F. Codd. A relational model of data for large shared data banks. *Commun. ACM*, 13(6):377–387, 1970.
9. C. Delobel. Normalisation and hierarchical dependencies in the relational data model. *ACM Trans. Database Syst.*, 3(3):201–222, 1978.
10. J. Demetrovics, L. Rónyai, and H. Son. On the representation of dependencies by propositional logic. In *MFDBS 91*, volume 495 of *Lecture Notes in Computer Science*, pages 230–242. Springer, 1991.
11. R. Fagin. Functional dependencies in a relational data base and propositional logic. *IBM Journal of Research and Development*, 21(6):543–544, 1977.
12. R. Fagin. Multivalued dependencies and a new normal form for relational databases. *ACM Trans. Database Syst.*, 2(3):262–278, 1977.
13. R. Fagin. Horn clauses and database dependencies. *J. ACM*, 29(4):952–985, 1982.
14. R. Fagin and M. Y. Vardi. The theory of data dependencies: a survey. In *Mathematics of Information Processing: Proceedings of Symposia in Applied Mathematics*, pages 19–71. American Mathematical Society, 1986.
15. W. Fan. XML constraints. In *DEXA Workshops 2005: Proceedings of the 16th International Workshop on Database and Expert Systems Applications*, pages 805–809. IEEE Computer Society, 2005.
16. G. Graetzer. *General Lattice Theory*. Birkhauser, 1998.
17. S. Hartmann and S. Link. More functional dependencies for XML. In *AdBIS*, number 2798 in Lecture Notes in Computer Science, pages 355–369. Springer, 2003.
18. S. Hartmann and S. Link. Numerical constraints for XML. In *WoLLIC*, number 4576 in Lecture Notes in Computer Science, pages 203–217. Springer, 2007.
19. S. Hartmann and S. Link. Characterising nested database dependencies by fragments of propositional logic. *Ann. Pure Appl. Logic*, 152(1-3):84–106, 2008.
20. S. Hartmann and S. Link. Efficient reasoning about a robust XML key fragment. *ACM Trans. Database Syst.*, 34(2:8), 2009.
21. S. Hartmann and S. Link. Expressive, yet tractable XML keys. In *EDBT*, number 360 in ACM International Conference Proceeding Series, pages 357–367, 2009.
22. S. Hartmann, S. Link, and H. Köhler. Full hierarchical dependencies in fixed and undetermined universes. *Ann. Math. Artif. Intell.*, 50(1-2):195–226, 2007.

23. S. Hartmann and T. Trinh. Axiomatising functional dependencies for XML with frequencies. In *FoIKS*, number 3861 in Lecture Notes in Computer Science, pages 159–178. Springer, 2006.
24. L. Libkin. *Elements of Finite Model Theory*. Springer, 2006.
25. S. Link. Charting the completeness frontier of inference systems for multivalued dependencies. *Acta Inf.*, 45(7-8):565–591, 2008.
26. S. Link. On the implication of multivalued dependencies in partial database relations. *Int. J. Found. Comput. Sci.*, 19(3):691–715, 2008.
27. Y. Sagiv, C. Delobel, D. S. Parker Jr., and R. Fagin. An equivalence between relational database dependencies and a fragment of propositional logic. *J. ACM*, 28(3):435–453, 1981.
28. B. Thalheim. *Dependencies in Relational Databases*. Teubner-Verlag, 1991.
29. B. Thalheim. *Entity-Relationship Modeling: Foundations of Database Technology*. Springer, 2000.
30. M. Vincent. Semantic foundation of 4NF in relational database design. *Acta Inf.*, 36:1–41, 1999.
31. M. Vincent, J. Liu, and C. Liu. Strong functional dependencies and their application to normal forms in XML. *ACM Trans. Database Syst.*, 29(3):445–462, 2004.
32. C. Yu and H. Jagadish. XML schema refinement through redundancy detection and normalization. *VLDB J.*, 17(2):203–223, 2008.