Functional Dependencies for XML

Axiomatisation and Normal Form
in the presence of
Frequencies and Identifiers

A thesis presented in partial fulfilment of the requirements for the degree
of

MASTER OF SCIENCES
IN
INFORMATION SYSTEMS

at Massey University, Palmerston North, New Zealand

Diem-Thu Trinh
Revised March 2005

Work supervised by
Dr Sven Hartmann

Acknowledgement

I would like to thank all the people who have supported me throughout this year. Although I do not mention anyone specifically, I hope you all know who you are. For some of you, I am thankful for the guidance and help that you have provided me. For others of you, I am also grateful for all the laughter and friendship that we have shared throughout this time - this year would not have been half as enjoyable without this.

I would also like to extend my gratitude to Mrs Clark for the generous Lovell & Berys Clark Scholarship and to the NZVCC for awarding me the William Georgetti Scholarship. The scholarships have provided me with a huge amount of financial support, allowing me to better focus on my study. The scholarships have also been a great source of motivation and encouragement for me this year.

Thu Trinh
December 17, 2004.

Contents

1 Introduction

XML has gained popularity as a markup language for publishing and exchanging data on the web. Nowadays, there are also ongoing interests in using XML for representing and actually storing data. In particular, much effort has been directed towards turning XML into a real data model by improving the semantics that can be expressed about XML documents. Various works have addressed how to define different classes of integrity constraints and the development of a normalisation theory for XML. One area which received little to no attention from the research community up to five years ago is the study of functional dependencies in the context of XML [37]. Since then, there has been increasingly more research investigating functional dependencies in XML. Nevertheless, a comprehensive dependency theory and normalisation theory for XML have yet to emerge.

Functional dependencies are an integral part of database theory in the relational data model (RDM). In particular, functional dependencies have been vital in the investigation of how to design "good" relational database schemas which avoid or minimise problems relating to data redundancy and data inconsistency. Since the same problems can be shown to exist in poorly designed XML schemas[1], there is a need to investigate how these problems can be eliminated in the context of XML. We believe that the study of an analogy to relational functional dependencies in the context of XML is equally significant towards designing "good" XML schemas.

Researchers have proposed various generalisations of functional dependencies for XML but many do not have direct counterparts in the RDM. One of the main types of storage solutions being proposed for XML is to extend current storage solutions for relational databases to manage XML data (e.g. [9, 13, 24, 25, 27, 39]). A motivation for this approach is that, while technologies for native XML databases are still in the early stages of development, technologies for relational databases are stable and widely available and used. We will limit our study to a definition of functional dependencies which can be readily mapped to the notion of functional dependencies in the RDM.

The main goal of this research is to study functional dependencies for XML in the presence of frequencies and identifiers. Frequencies describe the optionality and cardinality in a DTD which may be associated with elements and attributes, while identifiers refer to attributes whose values are unique within an XML document. Frequencies and identifiers exist naturally in XML, and their existence reflects the much richer (particularly more flexible) structure of XML data. It is therefore surprising that few research into functional

[1]By "XML schemas" we are referring to the structural information of XML data, either implicitly contained in an XML document or explicitly specified using DTDs or other schema specification language.

dependencies for XML has explicitly considered the additional structural information provided by means of frequencies and identifiers.

In this thesis, we continue the study of the subgraph-based approach towards functional dependencies presented in [14, 15]. The thesis covers two issues: axiomatisation for functional dependencies and a normal form which characterises the absence of redundancy. We will present an axiomatisation for functional dependencies on XML schemas with frequencies, and an axiomatisation for functional dependencies on XML schemas with both frequencies and identifiers. In the latter part we propose a normal form which makes use of identifiers. In accordance with other works relating to normal forms in the context of XML, we justify our normal form by proving that it guarantees the absence of redundancy.

1.1 Outline of the Thesis

The rest of the thesis is organised as follows. Section 2 contains some preliminary definitions relating to the XML graph model and a notion of functional dependencies in XML (called XFDs) which we will study. In the next two sections, we discuss inference rules for the derivation of XFDs and present axiomatisations of XFDs which we show to be both sound and complete for the derivation of XFDs. We consider XFDs in the presence of frequencies in Section 3 and XFDs in the presence of frequencies together with identifiers in Section 4. In Section 5, we generalise the notion of redundancy to the context of XML and propose a normal form which is both necessary and sufficient for the absence of redundancy. We also try to highlight the limitation of redundancy as a design quality criteria in the context of XML. Related work is identified in Section 6 and finally we conclude with some possible research directions in Section 7.

2 Preliminary Notations

2.1 XML Graph Model

In this section, we present the XML graph model introduced in [14, 15]. The reader is assumed to be familiar with standard notions from graph theory such as graphs, trees and walks. In accordance with [14, 15], all graphs will be considered to be directed, without parallel arcs and finite unless stated otherwise.

For every graph G, let V_G denote its set of vertices and A_G its set of arcs.

2.1.1 Rooted Graphs and Rooted Trees

A *rooted graph* is a graph with no vertices, or a graph G with one distinguished vertex r_G, called the *root* of G, such that there is a directed path from r_G to every other vertex in V_G. A *rooted tree* is a rooted graph T with no non-directed cycles.

A graph G is called *empty*, written $G = \emptyset$, if A_G is empty and *non-empty* otherwise. Specifically, G is an *empty rooted graph* if it consists of a single vertex r_G or no vertices.

For every vertex v, let $Succ_G(v)$ denote its (possibly empty) set of successors in G. *Leaves* are those vertices without successors. Let L_G denote the set of all leaves in G.

Definition 2.1. Given a vertex $v \in V_G$ and a subset $W \subseteq L_G$ of leaves, a *v-subgraph* of G is the graph union of all directed walks from v to some $w \in W$. A *v-walk* of G is a directed walk from v to a single leaf $w \in L_G$. Every v-subgraph/v-walk of a rooted tree T is again a rooted tree. □

2.1.2 XML Graphs

Let $ENames$ and $ANames$ be fixed sets of element names and attribute names respectively. Also let the symbols E, A and S reflect whether a vertex represents an element, attribute or text data respectively.

Definition 2.2. An *XML graph* is a rooted graph G together with the mappings $name : V_G \to ENames \cup ANames$ and $kind : V_G \to \{E, A, S\}$ assigning every vertex its name and kind respectively. We suppose that every vertex v of kind S is mapped to the same name as the name carried by the predecessor of v. If G is a rooted tree then we can also refer to an XML graph as an *XML tree.* □

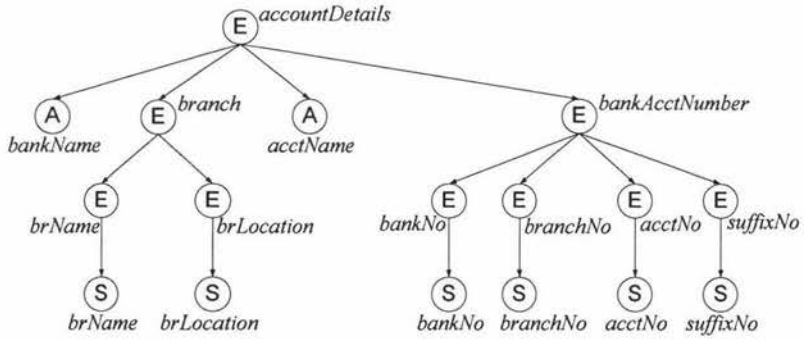


Figure 1: An XML tree showing the names and kinds of vertices

In XML no element has two attributes of the same name, therefore no vertex in an XML graph has two successors of kind A carrying the same name. We do not consider mix-content elements, therefore no vertex of kind E may have both a successor of kind S and a successor of kind E. All XML trees are assumed to be unordered trees.

Let V_G^E, V_G^A and V_G^S consist of all vertices in V_G of kind E, A and S respectively. We suppose that in a non-empty XML graph, vertices of kind A and S are always leaves and conversely all leaves are either of kind A or S, that is, $L_G = V_G^A \cup V_G^S$. Thus we also do not allow empty elements without attributes, except for the root element.

2.1.3 Mappings between XML Graphs

Definition 2.3. Let G' and G be two XML graphs, and consider a mapping $\phi : V_{G'} \to V_G$. ϕ is said to be *kind-preserving* if the image of a vertex is of the same kind as the vertex itself, that is, $kind(v') = kind(\phi(v'))$ for all $v' \in V_{G'}$. Further, ϕ is *name-preserving* if the image of a vertex carries the same name as the vertex itself, that is, $name(v') = name(\phi(v'))$ for all $v' \in V_{G'}$. The mapping ϕ is a *homomorphism* between G' and G if all of the following conditions hold:

(i) the root of G' is mapped to the root of G, that is, $\phi(r_{G'}) = r_G$

(ii) every arc of G' is mapped to an arc of G, that is, $(u', v') \in A_{G'}$ implies $(\phi(u'), \phi(v')) \in A_G$

(iii) ϕ is kind-preserving and name-preserving.

□

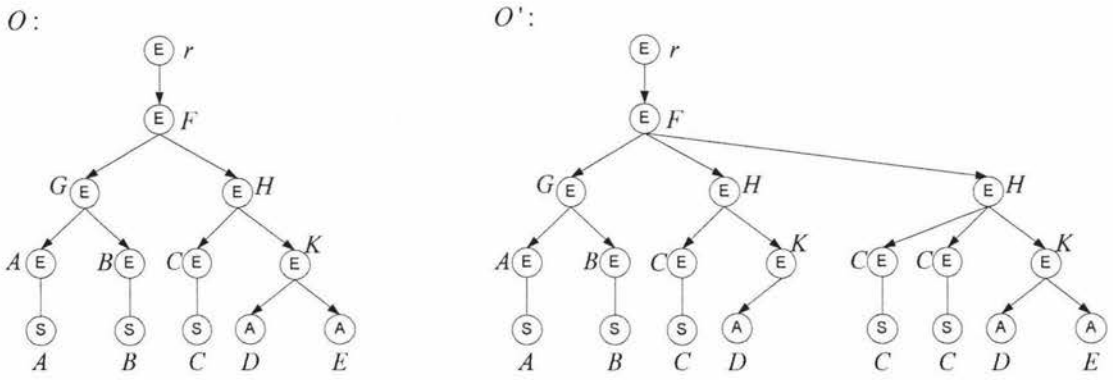


Figure 2: Example of two XML graphs (more specifically XML trees): O and O'.

Definition 2.4. A homomorphism $\phi : V_{G'} \to V_G$ is an *isomorphism* if ϕ is bijective and ϕ^{-1} is a homomorphism. Whenever such an isomorphism exists, G' is said to be *isomorphic* to G, denoted by $G' \cong G$. Alternatively, we may say that G' is a *copy* of G. □

Example 2.1. *Let us consider the two XML trees in Figure 2. There is a homomorphism $\phi : V_{O'} \to V_O$ between the XML tree O' and the XML tree O. This is the name-preserving mapping which maps every vertex from O' to the vertex carrying the same name in O. ϕ^{-1} is clearly root-preserving, name-preserving and kind-preserving. Further, all arcs in O is mapped to some arc in O' by ϕ^{-1}. Therefore ϕ^{-1} is a homomorphism. However, the homomorphism ϕ is not bijective, therefore O' is not isomorphic to O.* □

Definition 2.5. A subgraph H' of G' is a copy of a subgraph H of G if the restriction of $\phi : V_{G'} \to V_G$ to H' and H is an isomorphism between H' and H. An $r_{G'}$-subgraph H' of G' is a *subcopy of* G if it is a copy of some r_G-subgraph H of G. A maximal subcopy of G is a subcopy of G which is not an r_G-subgraph of any other subcopy of G. A maximal subcopy of G is called an *almost copy of* G. □

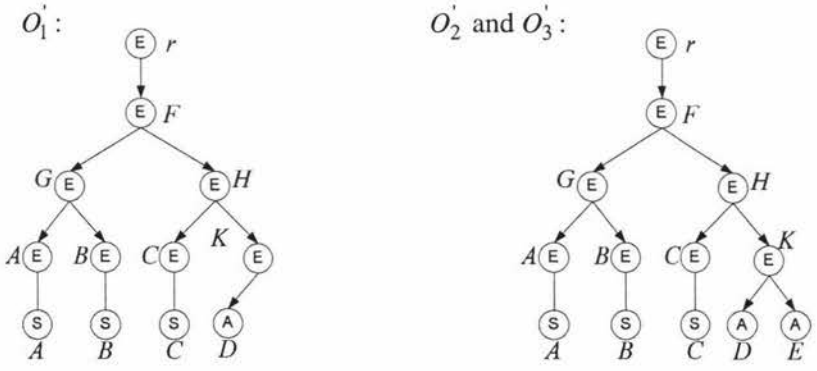


Figure 3: Three almost copies of O in O' . In particular, O'_2 and O'_3 are copies of O.

It should be noted that all copies of G in G' are almost copies of G, but not vice versa. Also we can observe that a homomorphism $\phi : V_{G'} \to V_G$ is not an isomorphism whenever G' contains more than one copy of G or no copy (but possibly many almost copies) of G .

2.1.4 XML Schema Graphs and XML Data Trees

Definition 2.6. An *XML schema graph* is an XML graph G together with a mapping $freq : A_G \to \{?, 1, +, *\}$ assigning every arc its frequency. Every arc $a = (v, w)$ where w is of kind A has frequency $freq(a) =?$ or 1. Every arc $a = (v, w)$ where $kind(v) = E$ and $kind(w) = S$ has frequency $freq(a) = 1$. Further, we assume no vertex in V_G has two successors with the same name and the same kind. If G is more specifically an XML tree, then we can talk about an *XML schema tree.* □

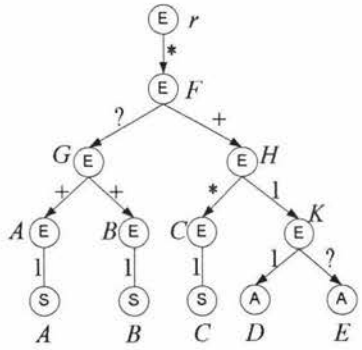


Figure 4: XML tree O (from Figure 2) together with a frequency labels on arcs. From this point on, let O be the XML schema graph shown here.

We re-emphasise, as in [14], that the term "schema graph" is used as an analogy to database schemas in traditional relational database design. There is no intended association between XML schema graphs and the language XML SCHEMA used to describe XML documents.

Unless stated otherwise, whenever the term "schema graph" or "schema tree" is used, we are in fact referring to "XML schema graph" and "XML schema tree" respectively.

An XML schema graph can be generated from a DTD or from an XML document. Refer to [14] for a discussion on how to derive an XML schema graph from a DTD and to [15] for how to generate an XML graph from an XML document.

Several things should be noted in connection with generating XML schema graphs. The first is that, whenever no DTD is available the designer needs to determine a suitable

frequency mapping for the generated XML graph. Secondly, if a DTD is recursive then we only consider a finite number of unfoldings of the recursive expression. Thirdly, we do not consider disjunction. If a DTD contains some disjunctive expression then we simply treat this as a sequential expression with each element in the disjunction being assigned a frequency of * or ?. For example, if $(A|B|C^{+})$ is a regular expression in a given DTD, then this is treated as $(A^{?}, B^{?}, C^{*})$. This is similar to the notion of simplifying DTD [27].

The following are some notations relating to frequencies used throughout the thesis. We use "an *f-arc*" to refer to an arc of frequency f and "an f/g-*arc*" to refer to an arc of frequency f or g. For example, a ?-arc refers to an arc of frequency ?, while a */+-arc refers to an arc of frequency * or +. For an XML schema graph G, let $G_{\leq 1}$ be the graph union of all ?/1-arcs in A_G, and let $G_{\geq 1}$ be the graph union of all 1/+-arcs in A_G. Note that $G_{\leq 1}$ and $G_{\geq 1}$ may not be r_G-subgraphs in G.

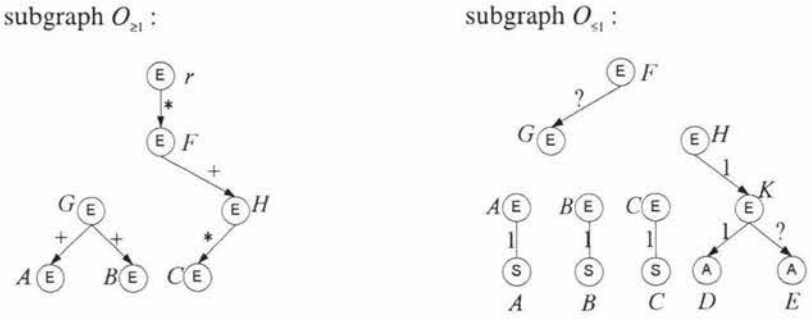


Figure 5: The subgraphs $O_{\geq 1}$ and $O_{\leq 1}$, where O is the XML schema graph from Figure 4.

Definition 2.7. An *XML data tree* is an XML tree T' together with an evaluation $val : L_{T'} \rightarrow STRING$ assigning every leaf v a (possibly empty) string $val(v)$. □

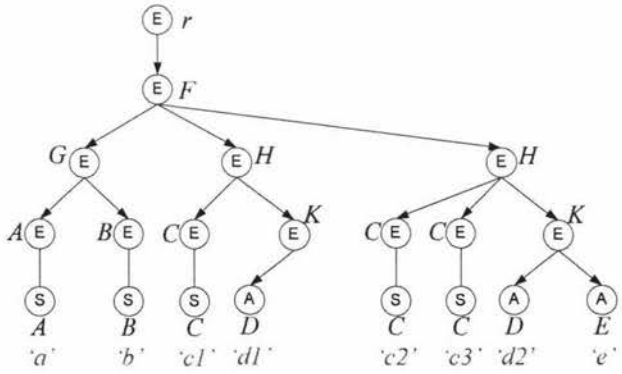


Figure 6: XML tree O' (from Figure 2) together with an evaluation of all leaves. From this point on, let O' be the XML data tree shown here.

Definition 2.8. Let G be an XML schema graph. An XML data tree T' is *compatible* with G, denoted by $T' \rhd G$, if there is a homomorphism $\phi : V_{T'} \to V_G$ between T' and G such that for every vertex v' of T' and every arc $a = (\phi(v'), w)$ of G, the number of arcs $a' = (v', w'_i)$ mapped to a is at most 1 if $freq(a) =?$, exactly 1 if $freq(a) = 1$, at least 1 if $freq(a) = +$ and arbitrarily many if $freq(a) = *$. In particular, due to Definition 2.6, this homomorphism is unique whenever it exists. □

Example 2.2. *Recall the homomorphism between O' and O from Example 2.1. It remains to check the frequencies.*

Let an n-vertex be a vertex carrying the name "n". For the single F-vertex in O' there are two arcs to H-vertices. The frequency of the F-vertex to H-vertex arc in O is $+$, that is, at least one arc is needed. This is true in O'. As another example, consider all H-vertices in O'. For each H-vertex, there is exactly one arc to a K-vertex. This is exactly what is required for the frequency of 1 for the H-vertex to K-vertex arc in O. After examining all frequencies in the schema tree O, we may conclude that $O' \rhd O$. □

Let T'_1 be any almost copy of T in an XML data tree $T' \rhd T$. It is possible that T'_1 does not contain a copy of some r_T-walk which contain an ?-arc or *-arc. Note that this flexibility is one of the desirable features of XML. We say that T'_1 is *missing* a copy of an r_T-walk C in T if T'_1 does not contain a copy of C, otherwise T'_1 is said to be *not missing* a copy of C. Similarly the data tree T' is said to be *missing* a copy of C if it does not contain a copy of C, and *not missing* a copy of C otherwise. In other words, T' is missing a copy of C if every almost copy of T in T' is missing a copy of C, and not missing a copy of C if there is some almost copy of T in T' which is not missing a copy of C. As an example, O'_1 is missing a copy of the r_O-walk to the leaf with name E in O, while O'_2 and O'_3 are not missing a copy of every r_O-walk of O. Altogether, the data tree O' is not missing a copy of every r_O-walk of O.

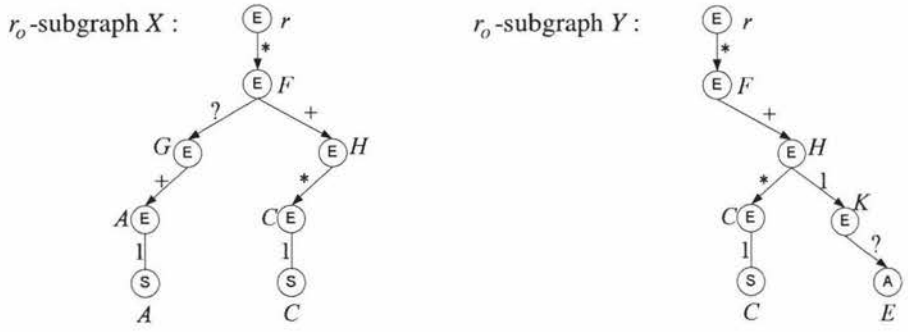
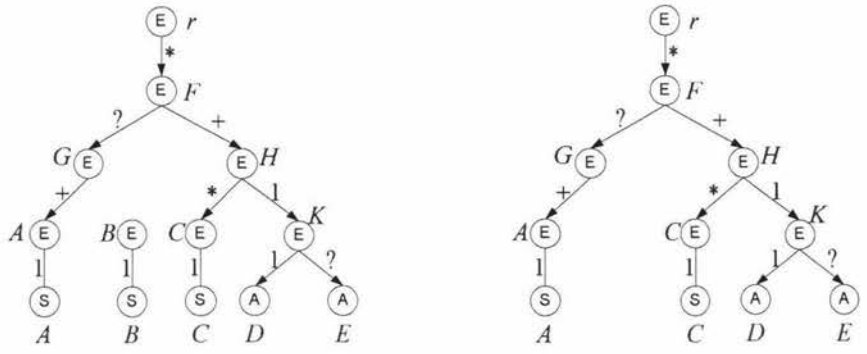


Figure 7: Two further subgraphs of O.

2.1.5 Operators on Subgraphs

Examples illustrating the operators defined in this subsection will also consider the two subgraphs shown in Figure 7.

Definition 2.9. Let G be an XML graph and let X and Y be two subgraphs in G. The *union* of X and Y, denoted by $X \cup Y$, is the restriction of the graph union of X and Y to its maximal r_G-subgraph of G. □



(a) The graph union of X and $O_{\leq 1}$. (b) $X \cup O_{\leq 1}$

Figure 8: An example illustrating the union and graph union of two subgraphs.

Definition 2.10. Let G be an XML graph and let X and Y be two subgraphs in G. The *intersection* of X and Y, denoted by $X \cap Y$, is the union of all r_G-walks that belong to X and to Y. The *difference* between X and Y, denoted by $X - Y$, is the union of all r_G-walks belonging to X but not to Y. In particular, $X \cap Y$ and $X - Y$ are r_G-subgraphs in G. □

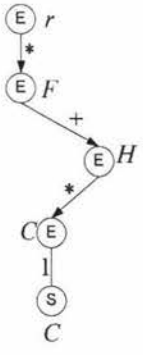


Figure 9: The result of $X \cap Y$

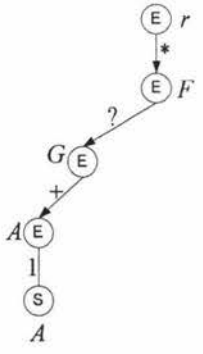
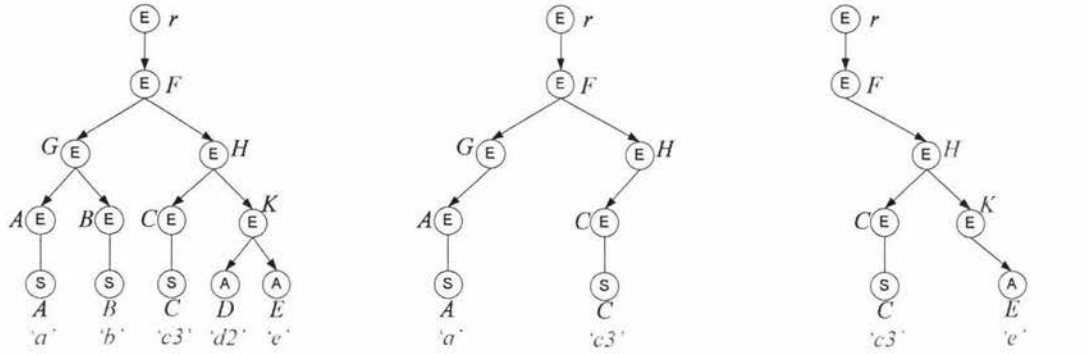


Figure 10: The result of $X - Y$

Remark. The intersection operator is associative but the union and difference operators are not associative. The union and intersection operators are commutative but the difference operator is not.

In the absence of parenthesis, we suppose the union and intersection operator bind tighter than the difference operator. For example, by $X \cup Y - Z$ we mean $(X \cup Y) - Z$.

Definition 2.11. Let G' and G be two XML graphs, and $\phi : V_{G'} \to V_G$ be a homomorphism between them. Given an r_G-subgraph H in G, the *projection* of G' to the subgraph H in G, denoted by $G'|_H$, is the union of all the subcopies of H in G'. The projection $G'|_H$ is an $r_{G'}$-subgraph of G'. □



(a) Almost copy O'_3 of schema graph O.

(b) The result of $O'_3|_X$

(c) The result of $O'_3|_Y$

Figure 11: Two examples for the projections of the almost copies O'_3.

Finally the following notations will be used for convenience. We may sometimes omit the union symbol altogether, for example, we may write XY instead of $X \cup Y$. Within each XML schema graph in the remainder of the thesis, leaves names are unique. Therefore, we may refer to an r_G-walk to some leaf carrying the name "B" simply as $\ll$B$\gg$. Further, we may refer to a non-empty r_G-subgraph X by listing the names of all leaves in X separated by white space. As an example, we may refer to the r_O-subgraph $X \cap Y$ [Figure 9] as $\ll$C$\gg$ and the r_O-subgraph X [Figure 7] as $\ll$A C$\gg$. For two r_G-subgraphs X and Y of some graph G, we may use $X \subseteq Y$ to denote that X is an r_G-subgraph of Y in G, and more specifically $X \in Y$ to denote that X is an r_G-walk of Y in G.

2.2 Running Example

The main running example in this thesis describes information stored by a Human Resource department about employees, their bank accounts, and also details of salaries/wages

payments made to each employee. We will segregate this information into two parts called Bank and Payment.

Both parts will include details about employees such as their name, employee ID and IRD number. A person who is liable to pay tax on his/her earnings may apply to the Inland Revenue Department for a unique IRD number which identifies them as an income tax payer of New Zealand.

In Bank, we will also include details about bank accounts to which salaries/wages payments are made. Each employee provides information on one bank account, including the bank name, account name, bank account number, and possibly information about the branch where the account is held. A bank account number consists of four components: bank number, branch number, account number and suffix number. Suppose we also need to store information about the person to contact about maintaining employees bank information. An XML schema tree for Bank, called BANK, can be found in Figure 12.

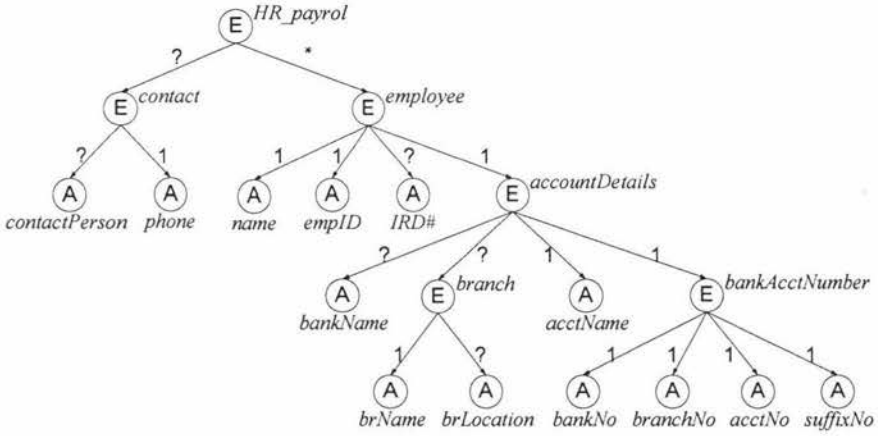


Figure 12: The XML schema tree BANK.

With the Payment part, we will additionally record information about salaries/wages payments made to each employee. Suppose this information includes the time of payment, cumulative number of payments up to that time, and amount paid together with details on deductions and tax. An XML schema tree for Payment will be given later in Section 4 (when we introduce identifiers).

It should be noted that we prefer using attributes wherever possible rather than text elements in the examples purely to end up with more compact XML graphs. Of course, each attribute in an XML graph G with name n may alternatively be modelled by a vertex v of kind E with name n followed by a vertex $w \in Succ_G(v)$ of kind S and name n. As an illustration of this, compare the XML graph in Figure 1 with the corresponding

subgraph of BANK in Figure 12. We will not concern ourselves with the issue of whether some information are better modelled as an attribute or a text element.

2.3 Functional Dependencies in XML

We are ready to present a definition of functional dependencies for XML. We utilise the definition introduced in the first part of [15].

2.3.1 Defining XFDs

Definition 2.12. Two XML data trees T' and T are said to be *equivalent*, denoted by $T' = T$, if the isomorphism $\phi : V_{T'} \to V_T$ between T' and T is evaluation-preserving, that is, $val(\phi(v')) = val(v')$ holds for every $v' \in L_{T'}$. □

Definition 2.13. Given an XML schema graph T, a *functional dependency* (or XFD for short) on T is an expression $X \to Y$ where X and Y are non-empty r_T-subgraphs in T. Let T' be an XML data tree which is compatible with T and let $\phi : V_{T'} \to V_T$ be the unique homomorphism between T' and T. Then T' *satisfies* the XFD $X \to Y$, written as $\models_{T'} X \to Y$, if and only if for any two almost copies T'_1 and T'_2 of T in T' the projections $T'_1|_Y$ and $T'_2|_Y$ are equivalent whenever the projections $T'_1|_X$ and $T'_2|_X$ are equivalent and copies of X, i.e. $T'_1|_Y = T'_2|_Y$ whenever $T'_1|_X = T'_2|_X \cong X$. □

Example 2.3. *From the information provided in Section 2.2 about the Bank part, we can specify the following XFDs:*

$$
\begin{array}{lllll}
(B\_XFD1) & \ll empID \gg & \to & \ll IRD\# \gg & \\
(B\_XFD2) & \ll empID \gg & \to & \ll name \gg & \\
(B\_XFD3) & \ll empID \gg & \to & \ll bankName\ \ brName\ \ brLocation\ \ acctName \gg & \\
(B\_XFD4) & \ll empID \gg & \to & \ll bankNo\ \ branchNo\ \ acctNo\ \ suffixNo \gg & \\
(B\_XFD5) & \ll IRD\# \gg & \to & \ll empID \gg &
\end{array}
$$

The following XFDs are also required to model the banking environment we consider:

$$
\begin{array}{lllll}
(B\_XFD6) & \ll bankName \gg & & \to & \ll bankNo \gg \\
(B\_XFD7) & \ll bankNo \gg & & \to & \ll bankName \gg \\
(B\_XFD8) & \ll bankName & branchNo \gg & \to & \ll brName \gg \\
(B\_XFD9) & \ll bankName & branchNo \gg & \to & \ll brLocation \gg \\
(B\_XFD10) & \ll bankName & brName \gg & \to & \ll branchNo \gg \\
(B\_XFD11) & \ll bankName & brLocation \gg & \to & \ll brName \gg
\end{array}
$$

$$(B\_XFD12) \quad \ll bankName \quad acctNo \gg \quad \rightarrow \quad \ll brName \quad acctName \gg$$
$$(B\_XFD13) \quad \ll bankName \quad acctName \gg \quad \rightarrow \quad \ll acctNo \gg$$

B_XFD6 and B_XFD7 reflects that every bank is associated with a unique name and bank number. With B_XFD8 to B_XFD11, we capture the fact that for each bank, every branch is identifiable by a branch name, a branch location description and a branch number. That is, no two branches of the same bank share their location description, branch name or branch number.

It also happens that the branch at which a customer opens his/her first banking account with a bank is used for all other types of banking accounts he/she may open in future. This is represented by B_XFD12. For each bank, each banking account belonging to a customer can be identified by an account number together with a suffix number. However, B_XFD12 and B_XFD13 further represent that for each bank, every customer can be identified by their account number or account name, regardless of whether the customer has several accounts of various types. A banking account may be shared by various people.

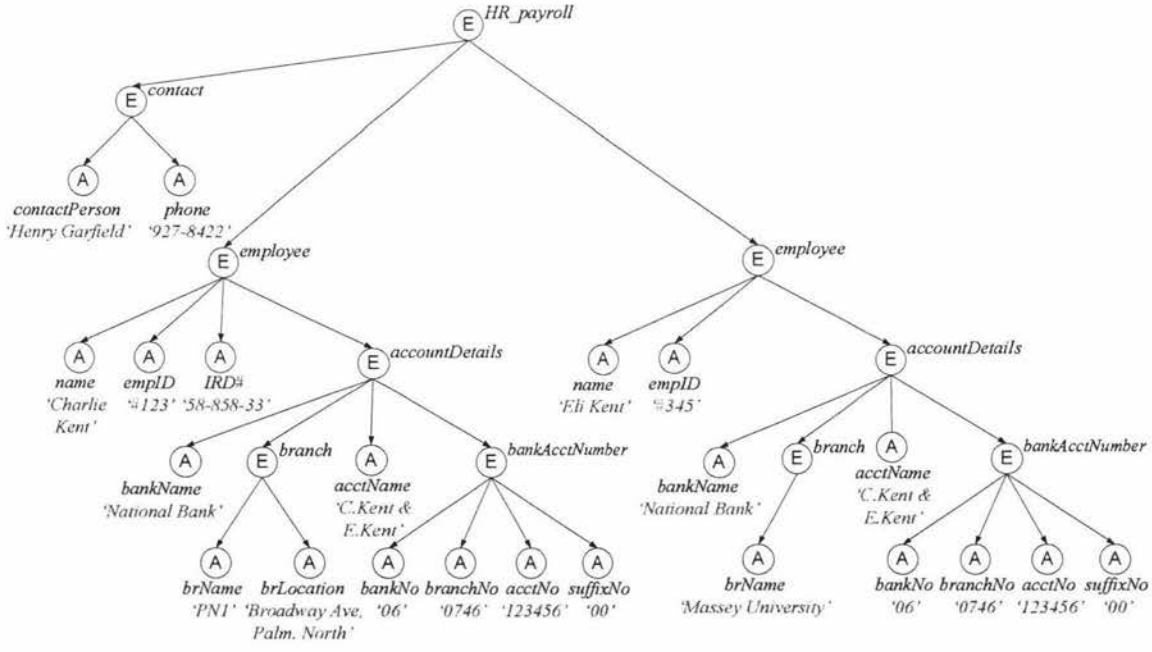


Figure 13: An XML data tree BANK' compatible with BANK.

The XML data tree BANK' *in Figure 13 contains exactly two almost copies of* BANK, *namely B'_1 and B'_2 shown in Figure 14. Both almost copies contain the values "National Bank" and "0746" for $\ll bankName \gg$ and $\ll branchNo \gg$ respectively; in other words, they contain equivalent copies of $\ll bankName \quad branchNo \gg$. However $B'_1|_{\ll brName \gg}$ has*

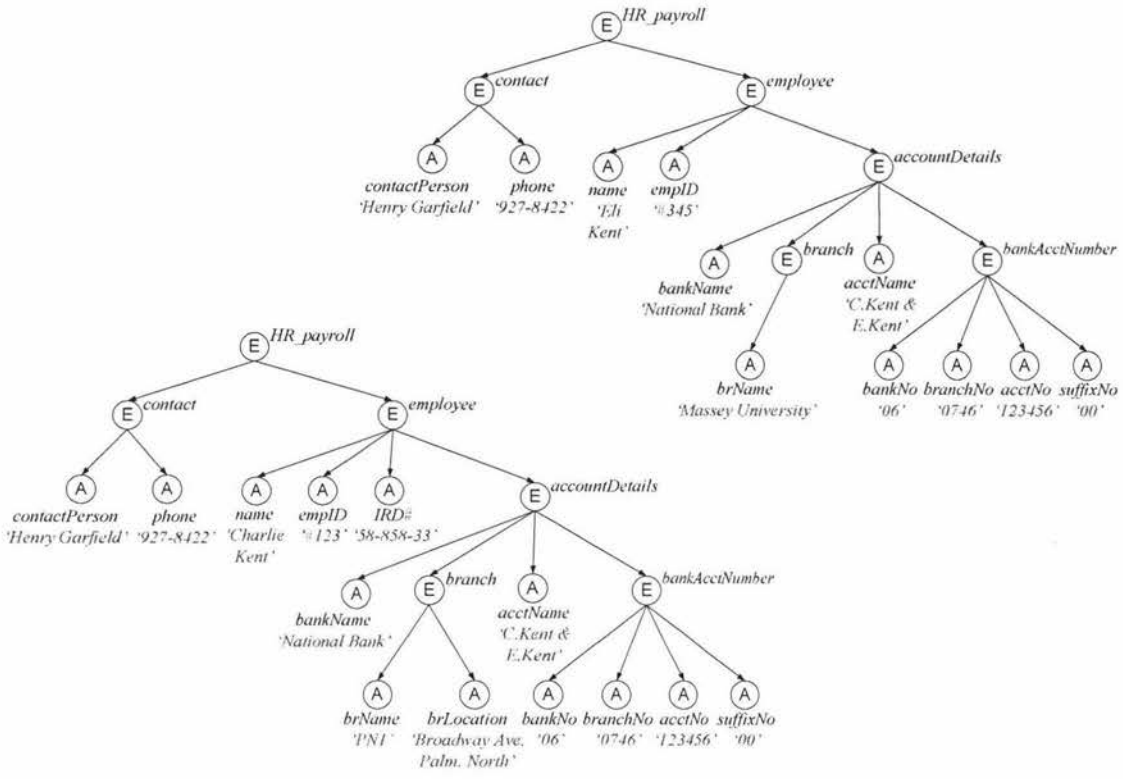


Figure 14: The two almost copies of BANK contained in BANK'. We will denote the bottom almost copy (with information on "Charlie") by B_1' and the other by B_2'.

the value "PN1", while $B_2'|_{\ll brName\gg}$ *has the value "Massey University". Also* B_1' *is not missing a copy of* $\ll brLocation\gg$ *, whereas* B_2' *contains no information about the branch location. Hence,* BANK' *does not satisfy* B_XFD8 *nor* B_XFD9.

On the other hand, B_XFD6 *is satisfied. Both* B_1' *and* B_2' *consist of the value "National Bank" for* $\ll bankName\gg$ *and both carry the value "06" for* $\ll bankNo\gg$ *. That is,* $B_1'|_{\ll bankName\gg} = B_2'|_{\ll bankName\gg} \cong \ll bankName\gg$ *implies* $B_1'|_{\ll bankNo\gg} = B_2'|_{\ll bankNo\gg}$ *and so* BANK' *satisfies* B_XFD6. *Since* $B_2'|_{\ll bankName\ brLocation\gg}$ *is missing a copy of* $\ll brLocation\gg$ *, it is the case that* B_XFD11 *is trivially satisfied.*

One by one, we can verify that all other XFDs are satisfied by BANK'. □

2.3.2 Implication and Derivation of XFDs

Analogous to the RDM, we say T' satisfies a given set Σ of XFDs, denoted by $\models_{T'} \Sigma$, if T' satisfies each XFD in Σ. As was found in the RDM, satisfaction of a given set of

XFDs by an XML data tree usually implies the satisfaction of other XFDs. The notions of implication and derivability (with respect to some rule system $\mathcal{R}$) are defined analogously to similar notions in the RDM.

Let Σ be a set of XFDs and $X \rightarrow Y$ a single XFD. If $X \rightarrow Y$ is satisfied in every XML data tree which satisfies Σ, then Σ *implies* $X \rightarrow Y$, written as $\Sigma \models X \rightarrow Y$. The *semantic closure of* Σ, denoted by Σ^*, is the set of all XFDs which are implied by Σ, that is, $\Sigma^* = \{X \rightarrow Y \mid \Sigma \models X \rightarrow Y\}$.

Given a rule system $\mathcal{R}$, we say an XFD $X \rightarrow Y$ is *derivable* from Σ by $\mathcal{R}$, denoted by $\Sigma \vdash_{\mathcal{R}} X \rightarrow Y$, if there is a finite sequence of XFDs, whose last element is $X \rightarrow Y$, such that each XFD in the sequence is in Σ or can be obtained from Σ by applying one of the inference rules in $\mathcal{R}$ to a finite number of previous XFDs in the sequence. The *syntactic closure of* Σ *with respect to the rule system* $\mathcal{R}$, denoted $\Sigma^+_{\mathcal{R}}$, is the set of all XFDs which are derivable from Σ by means of inference rules in $\mathcal{R}$, that is, $\Sigma^+_{\mathcal{R}} = \{X \rightarrow Y \mid \Sigma \vdash_{\mathcal{R}} X \rightarrow Y\}$. Whenever the rule system is clearly understood, we may omit $\mathcal{R}$.

An inference rule is called *sound* if for any given set Σ of XFDs, every XFD which may be derived from Σ due to that rule is also implied by Σ. A rule system $\mathcal{R}$ is *sound* if all inference rules in $\mathcal{R}$ are sound. In other words, $\mathcal{R}$ is sound if every XFD which is derivable from Σ by $\mathcal{R}$ is also implied by Σ (i.e. $\Sigma^+_{\mathcal{R}} \subseteq \Sigma^*$). A rule system is said to be *complete* if it is possible to derive every XFD which is implied by Σ (i.e. $\Sigma^* \subseteq \Sigma^+_{\mathcal{R}}$).

3 XFDs in the Presence of Frequencies

In this section, we reason about XFDs in the presence of frequencies. Our main result is a sound and complete rule system (called the $\mathcal{F}$-rule system) for the derivation of XFDs in the presence of frequencies.

We begin by introducing some basic sound inference rules which we then show to be complete as well. From the basic inference rules, additional inference rules can be derived. Like in the RDM, there are various sets of inference rules which can be shown to be equivalent, that is, all the XFDs which can be derived from one set of rules can be derived using the other set of rules and vice versa. We will consider a few equivalent sets of inference rules. Lastly, we will generalise the notion of Armstrong relations to XML data trees and investigate whether XFDs in the presence of frequencies enjoy data trees which are Armstrong.

3.1 Sound Inference Rules

The first four inference rules have been taken directly from [15]. Although quite obvious and simple, for the sake of completeness we include soundness proofs for these rules.

Lemma 3.1. *Let T be an XML schema graph and let X, Y, W, Z be r_T-subgraphs in T. The following inference rules for XFDs are sound:*

1. *(*union rule*).* $\dfrac{X \to Y, X \to Z}{X \to Y \cup Z}$
2. *(*reflexivity axiom*).* $\dfrac{}{X \to Y}$ *Y is an r_T-subgraph of X*
3. *(*subtree rule*).* $\dfrac{X \to Y}{X \to Z}$ *Z is an r_T-subgraph of Y*
4. *(*supertree rule*).* $\dfrac{W \to Y}{X \to Y}$ *W is an r_T-subgraph of X*

Proof.

1. Assume that the union rule is not sound, that is, there is an XML data tree $T' \rhd T$ with $\not\models_{T'} X \to Y \cup Z$ and $\models_{T'} \{X \to Y, X \to Z\}$. To violate $X \to Y \cup Z$, T' must includes two almost copies T'_1, T'_2 of T such that $T'_1|_X = T'_2|_X \cong X$ and $T'_1|_{Y \cup Z} \neq T'_2|_{Y \cup Z}$. This means that there is some r_T-walk C with $C \in Y$ or $C \in Z$ such that $T'_1|_C \neq T'_2|_C$. If $C \in Y$, then $T'_1|_Y \neq T'_2|_Y$ hence $\not\models_{T'} X \to Y$ which contradicts our assumption. Similarly if $C \in Z$ then $\not\models_{T'} X \to Z$ which also violates our assumption. Therefore by contradiction $\models_{T'} X \to Y \cup Z$.

2. Assume that we have an XML data tree $T' \rhd T$ in which $X \to Y$ is not satisfied and $Y \subseteq X$. This means there must be two almost copies T'_1, T'_2 of T such that $T'_1|_X = T'_2|_X \cong X$ and $T'_1|_Y \neq T'_2|_Y$. It can be easily seen from Definition 2.12 [equivalence] that $T'_1|_X = T'_2|_X$ if and only if $T'_1|_C = T'_2|_C$ for each r_T-walk C in X. Since $Y \subseteq X$ then using the *union rule* and the previous observation we get $T'_1|_Y = T'_2|_Y \cong Y$. This is a contradiction.

3. Assume that there is an XML data tree $T' \rhd T$ which satisfies $X \to Y$ but violates $X \to Z$ where $Z \subseteq Y$. This means there is some r_T-walk $C \in Z$ such that $T'_1|_C \neq T'_2|_C$. Since $Z \subseteq Y$ it is the case that $C \in Y$. This means $X \to Y$ is not satisfied which contradicts our assumption.

4. Assume $T' \rhd T$ is an XML data tree in which $W \to Y$ is satisfied but $X \to Y$ is not where $W \subseteq X$. There are two almost copies T'_1, T'_2 of T such that $T'_1|_X = T'_2|_X \cong X$, otherwise $X \to Y$ cannot be violated. From $W \subseteq X$ we get $T'_1|_W = T'_2|_W \cong W$. It follows that $T'_1|_Y = T'_2|_Y$, otherwise $W \to Y$ will be violated. But this means that $X \to Y$ is satisfied which contradicts our assumption.

□

We next define a root axiom which is a simplified version of the root axiom introduced in [15]. An explanation for the simplicity of our root axiom is that we do not allow element vertices in any non-empty XML graph to be leaves.

Lemma 3.2. *Let T be an XML schema graph and X be an r_T-subgraph of T. The following* root axiom *is sound for the inference of XFDs*

$$\frac{}{X \to R} \quad \textit{where } R \textit{ is the union over all } r_T\textit{-walks in } T_{\leq 1},$$

Proof. By definition, a data tree T' is compatible with an XML schema graph T if the number of arcs mapped in the homomorphism conforms to the frequencies specified in T. Since $R = \bigcup \{C \mid C \text{ is an } r_T\text{-walk of } T_{\leq 1}\}$, there is at most one copy in T' of each r_T-walk of R. Thus, any almost copy of T in T' will contain the same almost copy of R. It follows that for any two almost copies T'_1 and T'_2 of T we have $T'_1|_R = T'_2|_R$. Therefore $\models_{T'} X \to R$ for any r_T-subgraph X in T. □

It has been observed in [15] that the transitivity rule does not hold for XML in the presence of frequencies. Consider the XFDs $X \to Y$ and $Y \to Z$ defined on some XML schema graph T. For an XML data tree $T' \rhd T$, if any two almost copies of T are missing a copy of some r_T-walk of Y, then T' trivially satisfies $Y \to Z$. Therefore it would be possible

for two almost copies to consist of non-equivalent values for Z while being equivalent and not missing a copy of every r_T-walk of X, that is, $X \to Z$ can be violated.

However we can define a restricted form of the transitivity rule which is sound for the derivation of XFDs. The main idea behind such an inference rule is to use frequencies to ensure that two almost copies are not missing a copy of every r_T-walk of the middle term Y whenever they are not missing a copy of every r_T-walk of X or Z. The notion of Y being X, Z-compliant in the following definition accomplishes this. It should be noted that Definition 3.1 is a re-write of the same concept in [15].

Definition 3.1. Let X, Y and Z be r_T-subgraphs in an XML schema graph T. We say Y *is* X, Z*-compliant* if and only if $Y \subseteq (X \cup C) \cup T_{\geq 1}$ for every r_T-walk C in Z. □

Example 3.1. *According to the schema tree* BANK *in Figure 12, it is the case that* $\ll$*phone name*$\gg \subseteq (\ll$*contactPerson*$\gg \cup \ll$*empID*$\gg) \cup$ BANK$_{\geq 1}$. *Therefore we say* $\ll$*phone name*$\gg$ *is* $\ll$*contactPerson*$\gg$, $\ll$*empID*$\gg$ *-compliant. It can also be verified that* $\ll$*IRD#*$\gg$ *is not* $\ll$*contactPerson*$\gg$, $\ll$*empID*$\gg$ *-compliant.* □

Lemma 3.3. *Let* T *be an XML schema graph and* X, Y, Z *be* r_T*-subgraphs in* T. *The* restricted-transitivity rule *defined as follows is sound:*

$$\frac{X \to Y, Y \to Z}{X \to Z} \; Y \text{ is } X, Z\text{-compliant.}$$

Proof. Assume that there is a data tree $T' \rhd T$ such that $\models_{T'} \{X \to Y, Y \to Z\}$ with Y being X, Z-compliant but $\not\models_{T'} X \to Z$. Suppose the violation of $X \to C$ causes $X \to Z$ to be violated, where C is an r_T-walk of Z. This means T' must have two almost copies T'_1, T'_2 of T such that $T'_1|_X = T'_2|_X \cong X$ and $T'_1|_C \neq T'_2|_C$. By assumption we have $Y \subseteq (X \cup C) \cup T_{\geq 1}$. From $\models_{T'} X \to Y$, we know $T'_1|_Y = T'_2|_Y$. If $T'_1|_Y = T'_2|_Y \cong Y$ then $T'_1|_C = T'_2|_C$ since $\models_{T'} Y \to Z$. Hence, $T'_1|_Y = T'_2|_Y \ncong Y$. That is, T'_1 and T'_2 are missing a copy of some r_T-walk $B \in Y$. However $T'_1|_C \neq T'_2|_C$ tells us that T'_1 or T'_2 is not missing a copy of C. It follows that T'_1 or T'_2 contains every r_T-walk of $(X \cup C) \cup T_{\geq 1}$. Hence $B \notin (X \cup C) \cup T_{\geq 1}$ contradicting $B \in Y \subseteq (X \cup C) \cup T_{\geq 1}$. □

Example 3.2. *Recall the XML schema tree* BANK *and all XFDs specified in Example 2.3.*

There are two r_{BANK}*-walks in* BANK$_{\leq 1}$, *yielding* $R =$ $\ll$*contactPerson phone*$\gg$. *Therefore, with the* root axiom *we may derive XFDs like* $\ll$*name bankNo branchNo acctNo suffixNo*$\gg \to \ll$*contactPerson phone*$\gg$ *and* $\ll$*bankName*$\gg \to \ll$*contactPerson phone*$\gg$.

The supertree rule *enables us to derive from* B_XFD9 *the XFD* $\ll bankName \quad branchNo \quad empID\gg \rightarrow \ll brLocation\gg$. *Applying the* subtree rule *to* B_XFD12 *results in the two XFDs* $\ll bankName \quad acctNo\gg \rightarrow \ll brName\gg$ *and* $\ll bankName \quad acctNo\gg \rightarrow \ll acctName\gg$.

By means of the reflexivity axiom, *we can derive* $\ll bankName \quad brName\gg \rightarrow \ll bankName\gg$. *Then from this new XFD and* B_XFD10 *and an application of the* union rule *we obtain* $\ll bankName \quad brName\gg \rightarrow \ll bankName \quad branchNo\gg$. *It is easy to see that:*

$$\ll bankName \quad branchNo\gg \subseteq \left(\ll bankName \quad brName\gg \cup \ \ll brLocation\gg \right) \cup \text{BANK}_{\geq 1}$$

which means $\ll bankName \quad branchNo\gg$ *is* $\ll bankName \quad brName\gg, \ll brLocation\gg$ *-compliant. Therefore an application of the* restricted-transitivity rule *to* $\ll bankName \quad brName\gg \rightarrow \ll bankName \quad branchNo\gg$ *and* B_XFD9 *yields the XFD* $\ll bankName \quad brName\gg \rightarrow \ll brLocation\gg$. □

We next define the notion of a unit of some r_T-walk which is used in the last basic inference rule presented in this section.

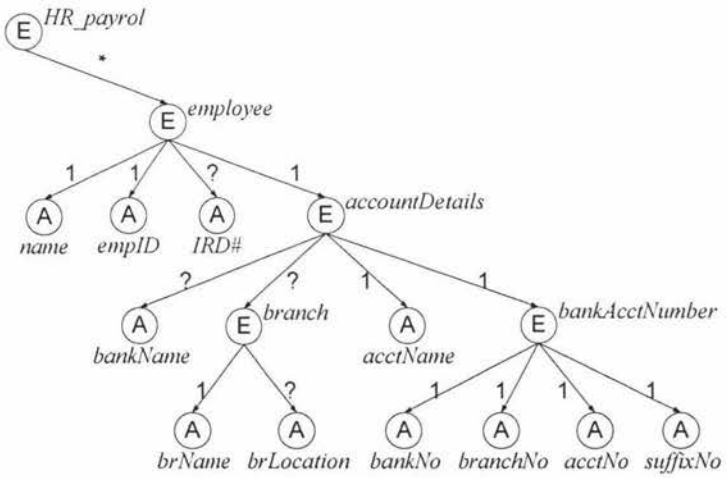


Figure 15: The unit of: ≪name≫ , ≪bankName≫ , ≪bankNo≫ , and every other r_{BANK}-walk in the given r_{BANK}-subgraph. Note that the unit of ≪contactPerson≫ and ≪phone≫ (r_{BANK}-walks not in the given r_{BANK}-subgraph) is the empty rooted graph.

Definition 3.2. Let B be an r_T-walk of some XML schema graph T. The *unit of* B, denoted by U_B, is the union of all r_T-walks sharing some */+-arc with B. □

There are a few observations we can make in relation to the unit of some r_T-walk which are useful for later proofs. For one, it is the case that $U_C = U_B$ for any r_T-walk $C \in U_B$. Furthermore, in any data tree $T' \rhd T$, every almost copy of $T - U_B$ together with any almost copy of U_B form an almost copy of T in T'. In particular, for any two almost copies T'_1, T'_2 of T in T', it is the case that $T'_1|_{T-U_B} \cup T'_2|_{U_B}$ and $T'_2|_{T-U_B} \cup T'_1|_{U_B}$ are also almost copies of T in T'. The mix-and-match approach is only possible because $T'_2|_{U_B}$ shares with $T'_1|_{T-U_B}$ exactly those arcs (and vertices) which $T'_2|_{U_B}$ shares with $T'_2|_{T-U_B}$, and likewise $T'_1|_{U_B}$ shares with $T'_2|_{T-U_B}$ exactly those arcs which $T'_1|_{U_B}$ shares with $T'_1|_{T-U_B}$.

We are now ready to define the last inference rule required in the presence of frequencies. The rule bears a slight resemblance to the notion of "flimsy" XFDs introduced in [22].

Lemma 3.4. *Let T be an XML schema graph, X be an r_T-subgraph and B an r_T-walk of T. The following* noname rule *is sound for the implication of XFDs:*

$$\frac{\big((X \cup B) \cup T_{\geq 1} - U_B\big) \cup X \to B}{X \to B}$$

Proof. Assume there is a data tree $T' \rhd T$ such that $\models_{T'} ((X \cup B) \cup T_{\geq 1} - U_B) \cup X \to B$ but $\not\models_{T'} X \to B$. In particular, T' has two almost copies T'_1, T'_2 of T such that $T'_1|_X = T'_2|_X \cong X$ and $T'_1|_B \neq T'_2|_B$. We have $((X \cup B) \cup T_{\geq 1} - U_B) \not\subseteq X$, since otherwise $((X \cup B) \cup T_{\geq 1} - U_B) \cup X = X$ and it would follow immediately from the assumption that $\models_{T'} X \to B$. This means there is some non-empty r_T-subgraph $W \subseteq (X \cup B) \cup T_{\geq 1} - U_B$ such that $W \cap X = \emptyset$. For convenience, we can now rewrite the noname rule as follows:

$$\frac{\big((X \cup B) \cup T_{\geq 1} - U_B\big) \cup X \to B}{X \to B} = \frac{XW \to B}{X \to B}$$

T'_1 or T'_2 is not missing a copy of B and therefore must not be missing a copy of every r_T-walk of $(X \cup B) \cup T_{\geq 1}$. Without loss of generality assume $T'_1|_B \cong B$ and therefore $T'_1|_W \cong W$ from $W \subseteq (X \cup B) \cup T_{\geq 1}$. There are now two cases to consider: $T'_1|_W = T'_2|_W \cong W$ or $T'_1|_W \neq T'_2|_W$.

Firstly, suppose $T'_1|_W = T'_2|_W \cong W$. Recall that by assumption $T'_1|_X = T'_2|_X \cong X$, therefore we have $T'_1|_{XW} = T'_2|_{XW} \cong XW$. Since $\models_{T'} XW \to B$ it follows that $T'_1|_B = T'_2|_B$ which would contradict our assumption that $\not\models_{T'} X \to B$.

So instead, let us consider the case where $T'_1|_W \neq T'_2|_W$. Since $W \subseteq (X \cup B) \cup T_{\geq 1} - U_B \subseteq T - U_B$, it follows that no r_T-walk of W is in U_B. On the other hand, we have $B \in U_B$ otherwise, $B \in R$ and $\models_{T'} X \to B$ by soundness of the root axiom which would contradict

our assumption. From T'_1 and T'_2 we can form two other almost copies of T, namely: $T'_3 = T'_1|_{T-U_B} \cup T'_2|_{U_B}$ and $T'_4 = T'_2|_{T-U_B} \cup T'_1|_{U_B}$.

Consider the pair of almost copies T'_1, T'_3 of T. By construction of T'_3, we have $T'_1|_W = T'_3|_W$. This means we have established $T'_1|_{XW} = T'_3|_{XW} \cong XW$. Then from $\models_{T'} XW \rightarrow B$, it follows that $T'_1|_B = T'_3|_B$. Since $T'_3\ |_B$ is just $T'_2\ |_B$ then in fact $T'_1|_B = T'_2|_B$ which contradicts our assumption. This concludes the proof. □

Example 3.3. *The only interesting XFDs on* BANK *which we may derive using the noname rule must consist of an r_{BANK}-subgraph of $\ll$contactPerson phone$\gg$ on the left-hand-side and the right-hand-side must be an r_{BANK}-walk (let us call this RHS) other than $\ll$contactPerson$\gg$ and $\ll$phone$\gg$. Altogether, this would mean that in order to apply the noname rule, the XFD $\ll$contactPerson phone$\gg \rightarrow RHS$ must be derivable. It is easy to see that there are no XFDs in Example 2.3 of this form, nor any which can be derived. Hence, the noname rule is not applicable for deriving any XFDs on* BANK *which the other inference rules cannot.* □

Example 3.4. *To illustrate how the noname rule may be applied, let us look at a simple scenario:*

> *There is a small product development company specialising in manufacturing plastic prototypes. The company operates various departments and has multiple branches. It happens that each department is located at exactly one branch.*

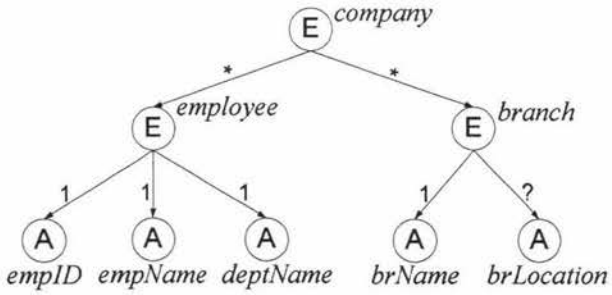


Figure 16: An XML schema tree PD for the product development scenario.

We may model this with the XML schema tree PD *given in Figure 16 and the set of XFDs* $\{ \ll brName \gg \rightarrow \ll brLocation \gg$, $\ll brLocation \gg \rightarrow \ll brName \gg$, $\ll deptName \gg \rightarrow \ll brName \gg \}$. *Using the noname rule we may derive the XFD* $\ll empName \gg \rightarrow \ll brName \gg$. *Let us verify this.*

We have $\ll$*empID empName deptName brName*$\gg$ *for the* r_{PD}*-subgraph corresponding to* $(\ll$*empName*$\gg \cup \ll$*brName*$\gg) \cup \mathrm{PD}_{\geq 1}$*, while* $U_{\ll brName \gg}$ *is the* r_{PD}*-subgraph* $\ll$*brName brLocation*$\gg$*. This amounts to the premise of the noname rule being:* $\ll$*empID empName deptName*$\gg \rightarrow \ll$*brName*$\gg$*. This XFD is derivable from* $\ll$*deptName*$\gg \rightarrow \ll$*brName*$\gg$ *by means of the supertree rule. Hence we may use the noname rule to derive* $\ll$*empName*$\gg \rightarrow \ll$*brName*$\gg$*.* □

3.2 A Sound & Complete Rule System

In this section, we prove that the inference rules from Section 3.1 form a complete rule system for XFDs in the presence of frequencies. Let the *$\mathcal{F}$-rule system* consist of the following inference rules: *reflexivity axiom, root axiom, subtree rule, supertree rule, union rule, restricted-transitivity rule* and *noname rule.*

We take the usual approach to proving completeness. Consider an XML schema graph T, and a given set Σ of XFDs on T. If $X \rightarrow Y$ cannot be derived from Σ by means of the inference rules, then we show that there is an XML data tree $T' \rhd T$ such that $\models_{T'} \Sigma$ but $\not\models_{T'} X \rightarrow Y$. Because of the union rule, this means there is some r_T-walk $B \in Y$ such that $X \rightarrow B$ is not derivable from Σ and T' does not satisfy $X \rightarrow B$. Therefore T' must contain two almost copies T'_1, T'_2 of T such that $T'_1|_X = T'_2|_X \cong X$ and $T'_1|_B \neq T'_2|_B$. We need a general construction for such a counterexample data tree T'.

Without frequencies, we can construct a counterexample data tree from the disjoint union of exactly two copies T'_a, T'_b of $X \cup B$ in which $T'_a|_C = T'_b|_C$ if and only if $C \in X$. Particularly, T'_a, T'_b are each missing a copy of every r_T-walks not in $X \cup B$. However, in the presence of frequencies, we face the additional complication that at least one almost copy of T must contain every r_T-walk of $(X \cup B) \cup T_{\geq 1}$. This means, in addition to $X \cup B$, we also need to determine whether or not T'_1 and T'_2 should be equivalent for each of the remaining r_T-walks in $(X \cup B) \cup T_{\geq 1}$, keeping in mind that T' must still satisfy Σ.

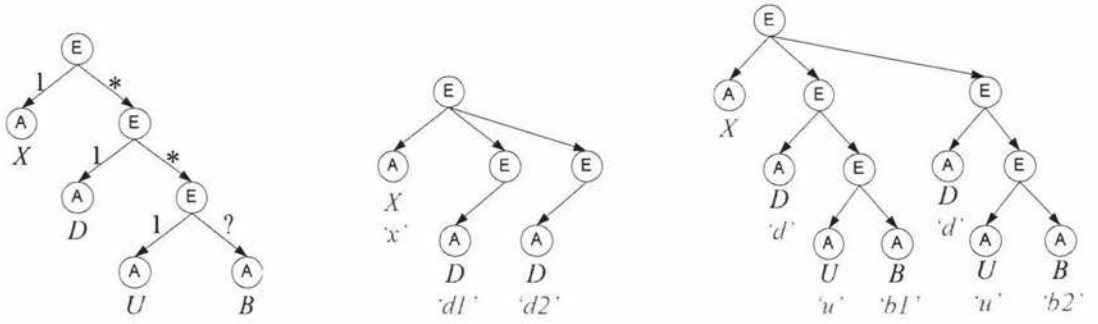
We first define the analogous of the closure of a set of attributes in the RDM.

Definition 3.3. Let T be an XML schema graph, X an r_T-subgraph and Σ a set of XFDs on T. Further let $\mathcal{R}$ be a rule system. The *pre-closure* $X^+_{\mathcal{R}}$ of X with respect to Σ and the rule system $\mathcal{R}$ is the r_T-subgraph defined as follows:

$$X^+_{\mathcal{R}} = \bigcup \{Y \mid X \rightarrow Y \in \Sigma^+_{\mathcal{R}}\}$$

□

Since $T_1'|_X = T_2'|_X \cong X$, to ensure $\models_{T'} \Sigma$ we must ensure that $T_1'|_{X_{\mathcal{F}}^+} = T_2'|_{X_{\mathcal{F}}^+}$. This means that the two almost copies of T must not be missing equivalent copies of every r_T-walk of $X_{\mathcal{F}}^+ \cap ((X \cup B) \cup T_{\geq 1})$. A peculiar situation is encountered: there may be XFDs not implied (hence not derivable by the $\mathcal{F}$-rule system), which need to be non-trivially satisfied in this case because T_1' or T_2' is not missing a copy of every r_T-walk of $(X \cup B) \cup T_{\geq 1}$. The restricted-transitivity rule means $(X_{\mathcal{F}}^+)_{\mathcal{F}}^+ \supseteq X_{\mathcal{F}}^+$, and in particular $X_{\mathcal{F}}^+$ is not a closure like its counterpart attribute closure in the RDM.



(a) Schema tree S. Let $\Sigma = \{ \ll X \gg \rightarrow \ll U \gg , \ll U \gg \rightarrow \ll D \gg \}$ be a set of XFDs on S.

(b) Data tree $S' \rhd S$ showing $\ll X \gg \rightarrow \ll D \gg$ can be violated.

(c) Data tree $S'' \rhd S$ with $\models_{S''} \Sigma$ and S'' is not missing a copy of $\ll B \gg$. It is not possible that $\not\models_{S''} \ll X \gg \rightarrow \ll D \gg$.

Figure 17: XML schema tree and data tree illustrating that there can be an XFD which is not derivable but which is satisfied when there are copies of certain r_T-walk(s).

This is illustrated by the simple XML schema tree S and set Σ of XFDs on S in Figure 17(a). In the schema tree S, $\ll U \gg \not\subseteq (\ll X \gg \cup \ll D \gg) \cup S_{\geq 1}$, that is $\ll U \gg$ is not $\ll X \gg , \ll D \gg$-compliant, therefore we cannot use restricted-transitivity to derive $\ll X \gg \rightarrow \ll D \gg$. Moreover, no other rule in the system allows us to derive $\ll X \gg \rightarrow \ll D \gg$. In order for a data tree to violate $\ll X \gg \rightarrow \ll D \gg$ it must simply be missing a copy of $\ll U \gg$, see Figure 17(b).

However, whenever a data tree compatible with S is not missing a copy of $\ll B \gg$, it will not be missing a copy of $\ll U \gg$. Consider the data tree S'' shown in Figure 17(c). S'' contains exactly two almost copies of S, let us call these S_1' and S_2'. Recall $\models_{S''} \Sigma$ means that $\models_{S''} \ll X \gg \rightarrow \ll U \gg$ and $\models_{S''} \ll U \gg \rightarrow \ll D \gg$. It follows from $\models_{S''} \ll X \gg \rightarrow \ll U \gg$ and $S_1'|_{\ll X \gg} = S_2'|_{\ll X \gg} \cong \ll X \gg$ that $S_1'|_{\ll U \gg} = S_2'|_{\ll U \gg}$. Since $\ll U \gg \in (\ll X \gg \cup \ll B \gg) \cup S''_{\geq 1}$ it is the case that S_1' and S_2' are actually not missing a copy of $\ll U \gg$. This means we have $S_1'|_{\ll D \gg} = S_2'|_{\ll D \gg}$ because $\models_{S''} \ll U \gg \rightarrow \ll D \gg$.

Returning to our discussion, this implies that it is insufficient to stop after having considered only $X_{\mathcal{F}}^{+}$. Because T_1' and T_2' are equivalent and not missing a copy of every r_T-walk of $X_{\mathcal{F}}^{+} \cap ((X \cup B) \cup T_{\geq 1})$, it follows from $\models_{T'} \Sigma$ that T_1' and T_2' must be equivalent and not missing a copy of every r_T-walk of $(X_{\mathcal{F}}^{+} \cap ((X \cup B) \cup T_{\geq 1}))_{\mathcal{F}}^{+} \cap ((X \cup B) \cup T_{\geq 1})$. This then means T_1' and T_2' must be equivalent and not missing a copy of every r_T-walk of $((X_{\mathcal{F}}^{+} \cap ((X \cup B) \cup T_{\geq 1}))_{\mathcal{F}}^{+} \cap ((X \cup B) \cup T_{\geq 1}))_{\mathcal{F}}^{+} \cap ((X \cup B) \cup T_{\geq 1})$ and so on. In the process, we obtain a sequence of pre-closures restricted to $((X \cup B) \cup T_{\geq 1})$. Eventually there is some fix-point X_n because there are only finitely many r_T-walks in $(X \cup B) \cup T_{\geq 1}$ in a finite schema graph. In summary, T_1' and T_2' must be equivalent and not missing precisely a copy of those r_T-walks in X_n in order for $\models_{T'} \Sigma$.

This covers our general approach for constructing a counterexample, although there are a couple of small differences for the approach actually used in the proof of completeness. Recall that $T_1'|_{T-U_B} \cup T_2'|_{U_B}$ and $T_2'|_{T-U_B} \cup T_1'|_{U_B}$ are also possible almost copies of T in T'. In particular, if $T_1'|_{T-U_B} \neq T_2'|_{T-U_B}$ and $T_1'|_{U_B} \neq T_2'|_{U_B}$ then there are at least four almost copies of T in T'. In order to ensure the constructed data tree will have exactly two almost copies of T, we will force T' to contain only one copy of $T|_{T-U_B} \cap ((X \cup B) \cup T_{\geq 1})$. Actually $T|_{T-U_B} \cap ((X \cup B) \cup T_{\geq 1})$ equals $(X \cup B) \cup T_{\geq 1} - U_B$. Therefore, we want to have $T_1'|_{(X \cup B) \cup T_{\geq 1} - U_B} = T_2'|_{(X \cup B) \cup T_{\geq 1} - U_B}$.

In this modified approach, there may be some r_T-walk C in $(X \cup B) \cup T_{\geq 1} - U_B$ which is not in the X_n described previously. This is rectified by actually computing the sequence of restricted pre-closures with $X_0 = ((X \cup B) \cup T_{\geq 1} - U_B) \cup X$ rather than just X. The noname rule means that each restricted pre-closure in the sequence is unaffected except for the additional r_T-subgraph $X_0 - X$.

Example 3.5. *We demonstrate the approach described thus far with an example. Consider the XML schema tree Q shown in Figure 18 together with the set $\Sigma = \{ \ll D\ B \gg \rightarrow \ll C \gg , \ll X \gg \rightarrow \ll B \gg \}$ of XFDs on Q. Obviously we have $\ll X\ F \gg \rightarrow \ll W \gg \notin \Sigma_{\mathcal{F}}^{+}$.*

Suppose we want to construct a counterexample data tree $Q' \rhd Q$ such that $\models_{Q'} \Sigma$ but $\not\models_{Q'} \ll X\ F \gg \rightarrow \ll W \gg$. $(\ll X\ F \gg \cup \ll W \gg) \cup Q_{\geq 1}$ is just the schema tree Q without the r_Q-walk $\ll E \gg$. Therefore each restricted pre-closure must not contain $\ll E \gg$.

The sequence of restricted pre-closures is as follows:

$$\begin{aligned} \ll X\ F \gg_0 &= ((\ll X\ F \gg \cup \ll W \gg) \cup Q_{\geq 1} - U_{\ll W \gg}) \cup \ll X\ F \gg \\ &= (\ll X\ F\ A\ B\ C\ D\ W \gg - \ll A\ B\ C\ F\ W \gg) \cup \ll X\ F \gg = \\ &= \ll X\ D \gg \cup \ll X\ F \gg = \ll X\ F\ D \gg \end{aligned}$$

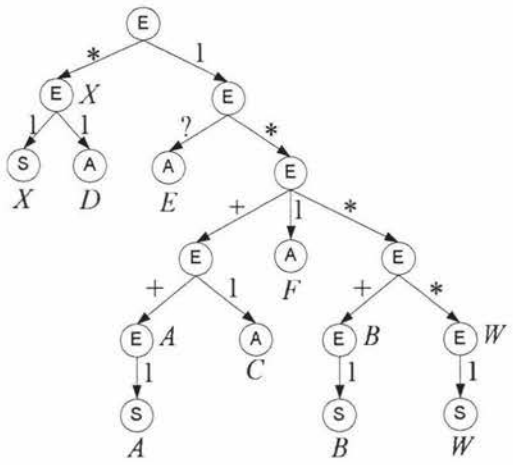


Figure 18: An XML schema tree Q.

$$\begin{aligned} \ll X\ F \gg_1 &= (\ll X\ F\ D \gg)^{+}_{\mathcal{F}} \cap \big((\ll X\ F \gg \cup \ll W \gg) \cup Q_{\geq 1}\big) = \ll X\ F\ D\ B \gg \\ \ll X\ F \gg_2 &= (\ll X\ F\ D\ B \gg)^{+}_{\mathcal{F}} \cap \big((\ll X\ F \gg \cup \ll W \gg) \cup Q_{\geq 1}\big) = \ll X\ F\ D\ B\ C \gg \\ &= \ll X\ F \gg_3 = \ll X\ F \gg_4 = \ldots \end{aligned}$$

Therefore, Q' must contain two almost copies Q'_1 and Q'_2 of Q which are equivalent and not missing precisely a copy of the r_Q-subgraph $\ll X\ F\ D\ B\ C \gg$. Two such almost copies of Q are depicted in Figure 19. □

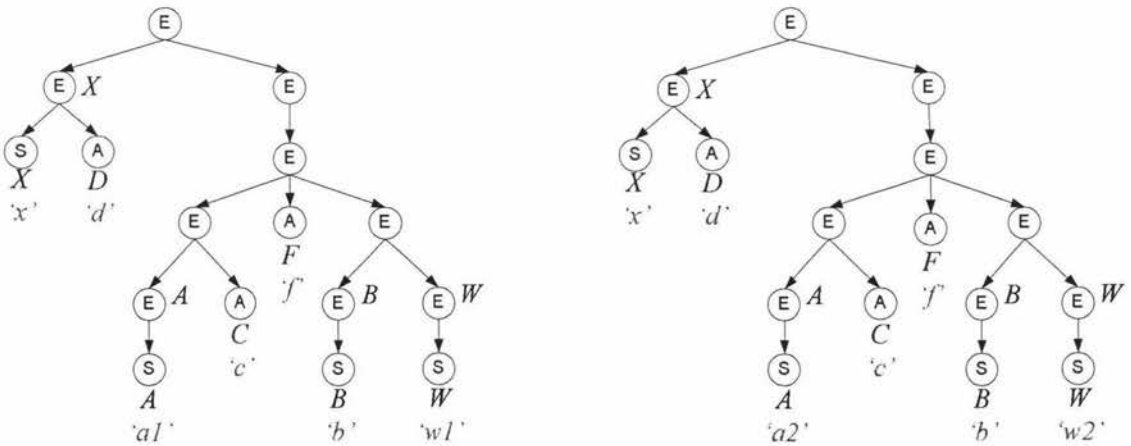


Figure 19: Two almost copies Q'_1, Q'_2 of the XML schema tree Q .

In accordance with the discussion above, we will construct T' by taking the union of two copies of $(X \cup B) \cup T_{\geq 1}$ which are equivalent only on X_n. Recall that T' must conform to the frequencies specified in T in order for it to be compatible with T. Consequently, in

T' the two copies may need to share certain arcs and/or even copies of entire r_T-walk(s) of T.

In the next definition, we provide a term to describe that two almost copies of some schema graph T in a compatible data tree T' share some $r_{T'}$-walk(s). This means that two almost copies share the same copy of some r_T-walk(s) of T. This concept will also be used in Section 5 to define a new notion of redundancy.

Definition 3.4. Let T be an XML schema graph. Two almost copies T'_1, T'_2 of T in an XML data tree $T' \rhd T$ are said to *coincide on an r_T-walk B* of T if and only if $T'_1|_B$ and $T'_2|_B$ are graph unions of exactly the same set of arcs in T'. Two almost copies of T *coincide on an r_T-subgraph Y* of T if and only if they coincide on every r_T-walk of Y. □

Example 3.6. *Recall the XML data tree* BANK' *from Figure 13 and the extracted almost copies B'_1, B'_2 shown in Figure 14. B'_1 and B'_2 coincide on the r_{BANK}-subgraph $\ll contactPerson \quad phone \gg$ but do not coincide on any other r_{BANK}-walk of* BANK. □

With the following amalgamation operator we are able to specify how two almost copies of some schema graph should be combined to construct an XML data tree.

Definition 3.5. Let T be an XML schema graph, T'_a, T'_b be two almost copies of T and X be an r_T-subgraph of T. The *amalgamation of T'_a and T'_b on X*, denoted $T'_a \amalg_{[X]} T'_b$, is the XML data tree obtained by taking the graph union of T'_a and T'_b in such a way that T'_a and T'_b coincide on X, and T'_a and T'_b only share all arcs in their projections to $X \cup T_{\leq 1}$. □

Recall that $T'_a|_{(X \cup W) \cup T_{\geq 1} - U_W} \cup T'_b|_{U_W}$ and $T'_b|_{(X \cup W) \cup T_{\geq 1} - U_W} \cup T'_a|_{U_W}$ are also almost copies of T in an XML data tree containing T'_a and T'_b. Further, from $X_0 = ((X \cup W) \cup T_{\geq 1} - U_W) \cup X \subseteq X_n$ we know that $T'_a|_{(X \cup W) \cup T_{\geq 1} - U_W} = T'_b|_{(X \cup W) \cup T_{\geq 1} - U_W}$. In order for the constructed data tree to contain exactly two almost copies of T, we require that T'_a and T'_b are amalgamated on $(X \cup W) \cup T_{\geq 1} - U_W$.

Example 3.7. *Let us continue with constructing the counterexample from Example 3.5. We need to construct Q' from two copies Q'_a and Q'_b of $(\ll X\ F \gg \cup \ll W \gg) \cup Q_{\geq 1} = \ll X\ F\ A\ B\ C\ D\ W \gg$ which are equivalent on the r_Q-subgraph $\ll X\ F\ D\ B\ C \gg$. Incidentally, these can simply be the two almost copies of Q from Figure 19, that is, $Q'_a = Q'_1$ and $Q'_b = Q'_2$.*

From Example 3.5, we have that $(\ll X\ F \gg \cup \ll W \gg) \cup Q_{\geq 1} - U_{\ll W \gg} = \ll X\ D \gg$. Therefore we amalgamate Q'_a and Q'_b on $\ll X\ D \gg$.

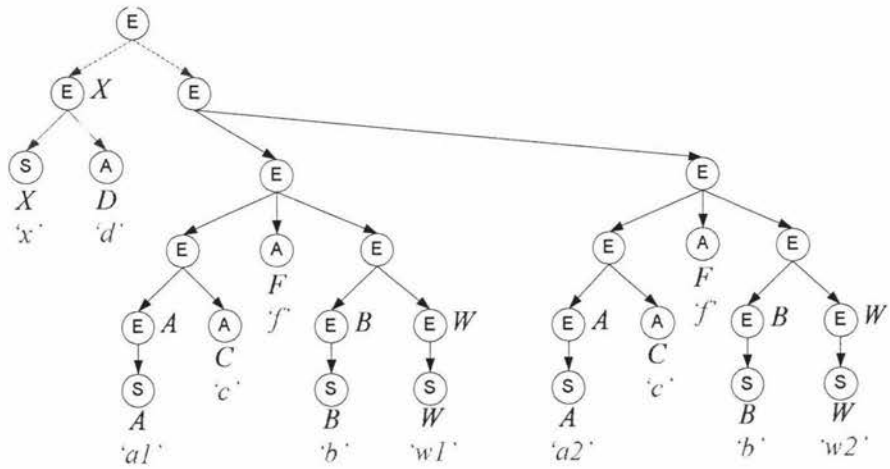


Figure 20: The XML data tree $Q' = Q'_a \amalg_{[XD]} Q'_b$ where $Q' \rhd Q$. Here Q'_a and Q'_b correspond to the almost copies Q'_1 and Q'_2 from Figure 19.

The result of $Q'_a \amalg_{[\ \ll X\ D \gg\ ]} Q'_b$ is the XML data tree Q' in Figure 20. Note that, Q' precisely contains the two almost copies Q'_1 and Q'_2. Though we have left out some vertices' names for convenience, the homomorphism between Q' and Q is the name-preserving mapping which maps every vertex from Q' to the vertex carrying the same name in Q. By also examining the frequencies, it can easily be verified that $Q' \rhd Q$. □

Theorem 3.5. *The $\mathcal{F}$-rule system is sound and complete for the derivation of XFDs in the presence of frequencies.*

Proof. Soundness of the inference rules has already been proven in Lemma 3.1, Lemma 3.2, Lemma 3.3 and Lemma 3.4. It remains to show that the inference rules form a complete rule system. Let T be an XML schema graph and Σ a set of XFDs on T. Let $X \to Y \notin \Sigma^+_{\mathcal{F}}$. That is, there must be an r_T-walk $B \in Y$ such that $X \to B$ cannot be derived, otherwise by means of the union rule $X \to Y$ would be derivable. We construct a counterexample data tree $T' \rhd T$ such that $\models_{T'} \Sigma$ and $\not\models_{T'} X \to B$.

Consider a sequence of restricted pre-closures $X_0, X_1, X_2 \ldots, X_{n-1}, X_n, X_{n+1}, X_{n+2}, \ldots$ where $X_n = X_{n+1} = X_{n+2} \ldots$ for $n \geq 0$, defined as follows:

$$\begin{aligned} X_0 &= \big((X \cup B) \cup T_{\geq 1} - U_B\big) \cup X \\ X_1 &= (X_0)^+_{\mathcal{F}} \cap \big((X \cup B) \cup T_{\geq 1}\big) \\ X_2 &= (X_1)^+_{\mathcal{F}} \cap \big((X \cup B) \cup T_{\geq 1}\big) \\ &\vdots \end{aligned}$$

$$\begin{aligned} X_{n-1} &= (X_{n-2})^+_{\mathcal{F}} \cap \big((X \cup B) \cup T_{\geq 1}\big) \\ X_n &= (X_{n-1})^+_{\mathcal{F}} \cap \big((X \cup B) \cup T_{\geq 1}\big) \\ &\vdots \end{aligned}$$

Let $X^B_{\mathcal{F}} = X_n$. Consider the XML data tree $T' = T'_a \amalg_{[\,(X \cup B) \cup T_{\geq 1} - U_B\,]} T'_b$ where T'_a, T'_b are two copies of $(X \cup B) \cup T_{\geq 1}$ with

$$T'_a|_W = T'_b|_W \text{ if and only if } W \subseteq X^B_{\mathcal{F}}$$

This is always possible since $T'_a|_{(X \cup B) \cup T_{\geq 1} - U_B} = T'_b|_{(X \cup B) \cup T_{\geq 1} - U_B}$. Further, U_B must contain at least the r_T-walk B, otherwise there is no $*/+$-arc in B which means $B \in R$ and $X \to B$ is derivable by means of the *root axiom* which would contradict our assumption. Moreover the resulting data tree T' contains exactly two almost copies of T: $T'_1 = T'_a$ and $T'_2 = T'_b$.

Firstly, we need to show that T' satisfies every XFD in Σ. Assume to the contrary that some $U \to V \in \Sigma$ is not satisfied by T'. There must be some r_T-walk $C \in V$ such that $U \to C$ is not satisfied by T'. From the construction, we have $U \subseteq X^B_{\mathcal{F}} = X_n$, otherwise $U \to C$ cannot be violated. From $U \to C \in \Sigma$ we get $C \in U^+_{\mathcal{F}} \subseteq (X_n)^+_{\mathcal{F}}$. Furthermore $C \in ((X \cup B) \cup T_{\geq 1})$, otherwise T' would be missing a copy of C and $U \to C$ would not be violated. Altogether $C \in (X_n)^+_{\mathcal{F}} \cap ((X \cup B) \cup T_{\geq 1}) = X_{n+1} = X_n = X^B_{\mathcal{F}}$. It follows that $T'_1|_C = T'_2|_C$ which contradicts our assumption that $U \to C$ is not satisfied by T'. Hence by contradiction $\models_{T'} U \to V$.

Lastly, we need to show that $X \to Y$ is violated by T'. If $B \in X^B_{\mathcal{F}} = X_n$, then $X_{n-1} \to B \in \Sigma^+_{\mathcal{F}}$ from the definition of X_n. Also, we have $X_{n-1} \subseteq (X_{n-2})^+_{\mathcal{F}}$ by definition and it follows that $X_{n-2} \to X_{n-1}$. Further $X_{n-1} \subseteq (X_{n-2} \cup B) \cup T_{\geq 1}$ since $X_{n-1} \subseteq (X \cup B) \cup T_{\geq 1}$ and $X \subseteq X_{n-2}$. This means X_{n-1} is $(X_{n-2}), B$-compliant. Thus application of the *restricted-transitivity rule* yields $X_{n-2} \to B \in \Sigma^+_{\mathcal{F}}$. Following the same reasoning, we can obtain $X_{n-3} \to B \in \Sigma^+_{\mathcal{F}}$, then $X_{n-4} \to B \in \Sigma^+_{\mathcal{F}}$, and so on until eventually we end up with $X_0 \to B \in \Sigma^+_{\mathcal{F}}$. Recall $X_0 = ((X \cup B) \cup T_{\geq 1} - U_B) \cup X$ and so we can use the *noname rule* to obtain $X \to B \in \Sigma^+_{\mathcal{F}}$. This would contradict our assumption, hence $B \notin X^B_{\mathcal{F}}$. According to the construction, T' contains two almost copies of T which are non-equivalent on B but equivalent and not missing a copy of every r_T-walk of X. This means $\not\models_{T'} X \to B$ hence $\not\models_{T'} X \to Y$. □

3.3 Additional Inference Rules for XFDs

In this section we identify some additional inference rules derivable from the basic rules given in Section 3.1.

Lemma 3.6. *Let T be an XML schema graph and X, Y, W, Z be r_T-subgraphs in T. The additional inference rules for XFDs are sound:*

1. *(*extension rule*).* $\dfrac{X \to Y}{X \to X \cup Y}$
2. *(*augmentation rule*).* $\dfrac{X \to Y}{X \cup W \to Y \cup W}$
3. *(*intersection rule*).* $\dfrac{X \to Y, X \to Z}{X \to Y \cap Z}$
4. *(*restricted-pseudo-transitivity*).* $\dfrac{X \to Y,\ Y \cup W \to Z}{X \cup W \to Z}$ *Y is $(X \cup W), Z$-compliant.*

Proof.

1. From the *reflexivity axiom* and $X \subseteq X$ we obtain $X \to X$. Then by means of the *union rule* and $X \to Y$, we get $X \to X \cup Y$.
2. By means of the *supertree rule* and $X \to Y$, we get $X \cup W \to Y$. From the *reflexivity axiom*, $X \cup W \to W$ since $W \subseteq X \cup W$. Finally using the *union rule* we obtain $X \cup W \to Y \cup W$.
3. Using the *union rule*, we get $X \to Y \cup Z$. Since $Y \cap Z \subseteq Y \cup Z$, by means of the *subtree rule* it is the case that $X \to Y \cap Z$.
4. By means of the *augmentation rule*, $X \to Y$ implies $X \cup W \to Y \cup W$. W is always $(X \cup W), Z$-compliant since $W \subseteq X \cup W$. Moreover, it follows from Y being $(X \cup W), Z$-compliant that $Y \cup W$ is $(X \cup W), Z$-compliant. Thus by *restricted-transitivity rule* we can infer $X \cup W \to Z$.

□

We now identify sets of inference rules which are equivalent to each other in the derivation of XFDs. As a consequence of Theorem 3.5, these results immediately allow us to identify alternative sound and complete rule systems for the derivation of XFDs in the presence of frequencies.

Proposition 3.1. *Let $\mathcal{S}$ denote the set of inference rules for XFDs consisting of the reflexivity axiom, union rule and subtree rule.*

1. *Every XFD derivable from* $\mathcal{S} \cup \{$*augmentation rule*$\}$ *is derivable from* $\mathcal{S} \cup \{$*supertree rule*$\}$ *and vice versa.*
2. *Every XFD derivable from* $\mathcal{S} \cup \{$*augmentation rule, restricted-pseudo-transitivity rule*$\}$ *is derivable from* $\mathcal{S} \cup \{$*augmentation rule, restricted-transitivity*$\}$ *and vice versa.*
3. *Every XFD derivable from* $\mathcal{S} \cup \{$*supertree rule, restricted-pseudo-transitivity rule*$\}$ *is derivable from* $\mathcal{S} \cup \{$*supertree rule, restricted-transitivity*$\}$ *and vice versa.*

Proof.

1. Lemma 3.6 has shown that every XFD derivable by augmentation rule is derivable by means of the supertree, reflexivity and union inference rules which are all in $\mathcal{S} \cup \{$supertree rule$\}$.

   It remains to show that every XFD derivable using the supertree rule is also derivable from $\mathcal{S} \cup \{$augmentation rule$\}$. By application of the augmentation rule to $X \rightarrow Y$ we obtain $X \cup W \rightarrow Y \cup W$. Then by means of the subtree rule we obtain $X \cup W \rightarrow Y$. The augmentation rule and subtree rule are both in $\mathcal{S} \cup \{$augmentation rule$\}$.

2. Lemma 3.6 also shows that the restricted-pseudo-transitivity rule is derivable by means of the restricted-transitivity rule and augmentation rule.

   Conversely, the restricted-transitivity rule is derivable from $\mathcal{S} \cup \{$augmentation rule, restricted-pseudo-transitivity rule$\}$. Suppose we have $X \rightarrow Y, Y \rightarrow Z$ and Y is X, Z-compliant. Let $W = \emptyset$. Using the sequence of derivation from the above proof of 1, $Y \cup W \rightarrow Z$ is derivable by means of the augmentation rule and subtree rule. Since Y is X, Z-compliant, it is $(X \cup W), Z$-compliant. Therefore using the restricted-pseudo-transitivity rule we obtain $X \cup W \rightarrow Z$. This means $X \rightarrow Z$ since $X \cup W = X$, which is the conclusion we need for the restricted-transitivity rule.

3. This is immediate from the proof of 1 and 2 above.

□

Corollary 3.1. *The sets of inference rules consisting of:*

- *reflexivity axiom, subtree rule, union rule, root axiom, noname rule, and*
- *augmentation rule or supertree rule, and*
- *restricted-pseudo-transitivity rule or restricted-transitivity rule*

are sound and complete. □

3.4 Armstrong XML Data Trees

For the relational data model, it is possible to construct Armstrong relations for certain classes of integrity constraints. A relation r over a relational schema R is said to be Armstrong with respect to a set of functional dependencies Σ over R if r satisfies precisely those functional dependencies implied by Σ and violates all other functional dependencies. We can define a similar notion for XML data trees.

Definition 3.6. Let T be an XML schema graph and Σ a set of XFDs on T. An XML data tree $T' \rhd T$ is said to be *Armstrong* with respect to Σ if and only if for every XFD $X \rightarrow Y$ over T, $\models_{T'} X \rightarrow Y$ if and only if $\Sigma \models X \rightarrow Y$. □

The immediate consequence of Definition 3.6 and the sound and complete $\mathcal{F}$-axiom system is that, in the presence of frequencies, T' is Armstrong with respect to Σ if and only if for every XFD $X \rightarrow Y$ over T, $\models_{T'} X \rightarrow Y$ if and only if $X \rightarrow Y \in \Sigma^+_{\mathcal{F}}$.

XFDs are said to enjoy Armstrong XML data trees if for any XML schema graph T and any set Σ of XFDs on T, it is possible to construct an XML data tree $T' \rhd T$ such that T' satisfies precisely Σ and violates all XFDs not implied by Σ. It is straightforward to show that XFDs do not enjoy Armstrong XML data trees in the presence of frequencies.

Theorem 3.7. *XFDs in the presence of frequencies do not enjoy Armstrong XML data trees.*

Proof. It is sufficient to give an example XML schema graph T and a set of XFDs on T for which no Armstrong data tree exists. Recall the schema graph from Figure 17(a) and the corresponding set of XFDs. It is easy to see that $\ll$X$\gg \rightarrow \ll$D$\gg$, $\ll$X$\gg \rightarrow \ll$B$\gg \notin \Sigma^+_{\mathcal{F}}$. We proceed to show that it is not possible to construct a data tree $T' \rhd T$ which satisfies Σ but violates both $\ll$X$\gg \rightarrow \ll$D$\gg$ and $\ll$X$\gg \rightarrow \ll$B$\gg$.

Assume there is an XML data tree $T' \rhd S$ which violates $\ll$X$\gg \rightarrow \ll$D$\gg$ and $\ll$X$\gg \rightarrow \ll$B$\gg$. Since $\not\models_{T'} \ll$X$\gg \rightarrow \ll$D$\gg$, there are two almost copies T'_1, T'_2 of S such that $T'_1|_{\ll X\gg} = T'_2|_{\ll X\gg} \cong \ll$X$\gg$ and $T'_1|_{\ll D\gg} \neq T'_2|_{\ll D\gg}$. From $\models_{T'} \ll$X$\gg \rightarrow \ll$U$\gg$ it follows that $T'_1|_{\ll U\gg} = T'_2|_{\ll U\gg}$. If the two almost copies are not missing a copy of $\ll$U$\gg$ then $T'_1|_{\ll D\gg} = T'_2|_{\ll D\gg}$ because $\models_{T'} \ll$U$\gg \rightarrow \ll$D$\gg$. Hence T'_1 and T'_2 must be missing a copy of $\ll$U$\gg$. However, T'_1 and T'_2 can only be missing a copy of $\ll$U$\gg$ if they are missing a copy of $\ll$B$\gg$. In this case we have $T'_1|_{\ll B\gg} = T'_2|_{\ll B\gg}$ which then yields $\models_{T'} \ll$X$\gg \rightarrow \ll$B$\gg$. We have obtained a contradiction. This completes our proof. □

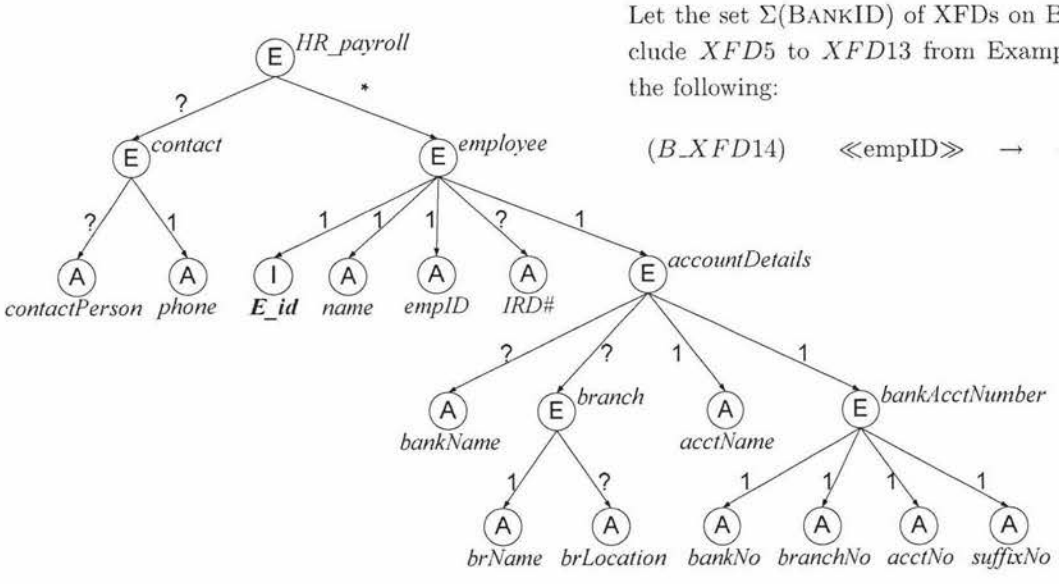
4 XFDs in the Presence of Frequencies and Identifiers

In addition to frequencies, we now consider the situation where certain attributes are only assigned values which are unique in an XML data tree. We refer to such attributes as *identifier-attributes*. In particular, an attribute of type ID in a DTD is an identifier-attribute in the corresponding XML schema graph. If no DTD is available then it is again up to the designer to add suitable identifiers to the generated XML graph. We first revise some definitions from Section 2.1 to account for the presence of identifier-attributes.

4.1 Revised XML Graph Model

We will use "I" to differentiate an identifier-attribute from an ordinary attribute. The kind assignment in Definition 2.2 [XML graph] is thus extended to: $kind : V_G \to \{E, A, I, S\}$. The possible frequencies for an arc $a = (v, w)$ where $kind(w) = I$ is the same as for $kind(w) = A$, that is, $freq(a) =?$ or 1. We add this to Definition 2.6 [XML schema graph].

We call an r_T-walk to some identifier-attribute an *identifier*.

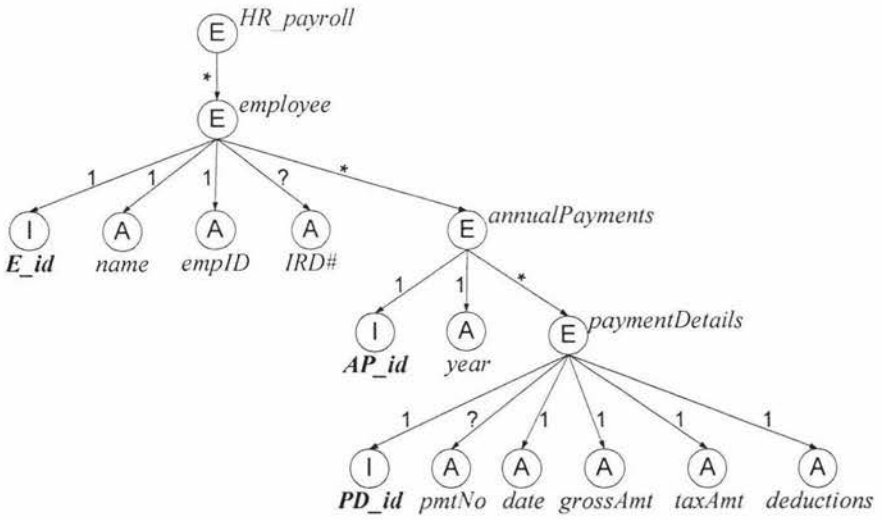


Let the set Σ(BANKID) of XFDs on BANKID include $XFD5$ to $XFD13$ from Example 2.3 and the following:

(B_XFD14) $\ll$empID$\gg$ $\to$ $\ll$E_id$\gg$

Figure 21: An XML schema tree BANKID which extends BANK by an identifier.

Let V_G^I consists of all vertices in V_G of kind I. Leaves are now vertices of kind A, I or S, that is $L_G = V_G^A \cup V_G^S \cup V_G^I$. It is not necessary to modify Definition 2.7 [XML data tree] and Definition 2.12 [equivalent].



Let Σ(PAYMENTID) be the set of XFDs on PAYMENTID consisting of:

$$
\begin{array}{lll}
(P\_XFD1) & \ll \text{empID} \gg & \rightarrow \ \ll \text{E\_id} \gg \\
(P\_XFD2) & \ll \text{IRD\#} \gg & \rightarrow \ \ll \text{empID} \gg \\
(P\_XFD3) & \ll \text{E\_id year} \gg & \rightarrow \ \ll \text{AP\_id} \gg \\
(P\_XFD4) & \ll \text{E\_id pmtNo} \gg & \rightarrow \ \ll \text{PD\_id} \gg \\
(P\_XFD5) & \ll \text{E\_id date} \gg & \rightarrow \ \ll \text{PD\_id} \gg
\end{array}
$$

Figure 22: An XML schema tree PAYMENTID (with identifiers).

Values assigned to identifier-attributes need to be unique in an XML data tree T', that is, for every string value $s \in STRING$ if there exists a vertex $v \in V_{T'}^{I}$ such that $val(v) = s$ then for every other leaf u we have $val(u) \neq s$. We refer to this as the *unique identifier value constraint.* The revision of Definition 2.8 [compatible] simply requires that T' also satisfies the unique identifier value constraint.

Example 4.1. *In the remainder of the section, we mainly consider the second part of the running example, that is the payments records. An XML schema tree* PAYMENTID *modelling the Payment part is shown in Figure 22. Consider the XML data tree* PAYMENTID' *depicted in Figure 23.*

The name-preserving mapping which maps every vertex from PAYMETNID' *to the vertex carrying the same name in* PAYMENTID *is a homomorphism between* PAYMENTID' *and* PAYMENTID. *It is easy to verify that this homomorphism complies with the frequencies specified in* PAYMENTID.

Now look at all vertices of kind I in PAYMENTID'. *The value of each vertex of kind I does not occur anywhere else in* PAYMENTID'. *This means* PAYMENTID' *satisfies the unique identifier value constraint. Therefore* PAYMENTID' *is compatible with* PAYMENTID.

Some examples in which the unique identifier value constraint is violated are:

- *if the value "e_1" was changed to "2003", or*
- *if the values "e_1" and "ap_1" were changed to "id_1", or*
- *if the value "ap_3" was changed to "ap_1".*

□

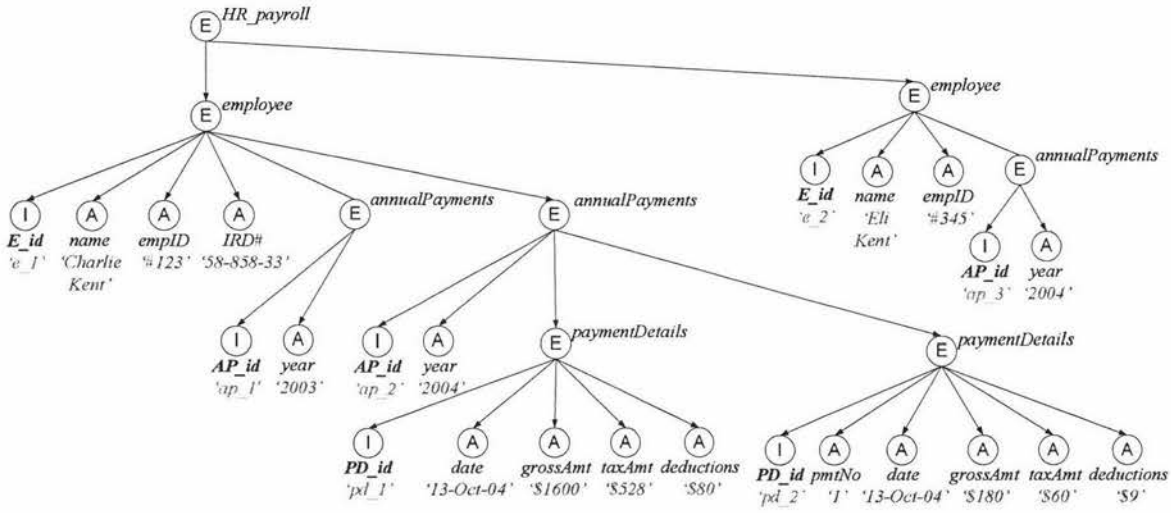


Figure 23: An XML data tree PAYMENTID' compatible with PAYMENTID.

4.2 Sound Inference Rules

It is easy to see that all inference rules which are sound in the presence of frequencies are also sound in the presence of frequencies and identifiers. However, a few other inference rules are required to obtain a sound and complete rule system in the presence of frequencies and identifiers.

The first rule stems from the uniqueness of values taken by identifier-attribute. For an r_T-subgraph X, let X_{ID} be the union of all identifiers in X, that is, $X_{ID} = \bigcup \{I \mid I \in X \text{ and } I \text{ is an identifier in } T\}$.

Lemma 4.1. *Let T be an XML schema graph and X an r_T-subgraph of T with X_{ID} being non-empty. The* noname2 axiom, *as defined below, is sound:*

$$\overline{X_{ID} \rightarrow X_{ID} \cup T_{\leq 1}}$$

Proof. Suppose there is a data tree $T' \rhd T$ such that there are two almost copies of T with $T'_1|_{X_{ID}} = T'_2|_{X_{ID}} \cong X_{ID}$. Since X_{ID} consists only of identifiers, T'_1 and T'_2 must coincide on X_{ID} due to the unique identifier value constraint. Further, there is only one almost copy S' of $X_{ID} \cup T_{\leq 1}$ in T' such that $S'|_{X_{ID}} = T'_1|_{X_{ID}} = T'_2|_{X_{ID}}$. Therefore T'_1 and T'_2 must in fact coincide on $X_{ID} \cup T_{\leq 1}$. This implies that $T'_1|_{X_{ID} \cup T_{\leq 1}} = T'_2|_{X_{ID} \cup T_{\leq 1}}$. □

Example 4.2. *In* PAYMENTID *there are three identifiers: $\ll$E_id$\gg$, $\ll$AP_id$\gg$ and $\ll$PD_id$\gg$. The noname2 axiom allows us to derive the following XFDs:*

$$\begin{array}{lcl} \ll E\_id \gg & \rightarrow & \ll E\_id \ \ name \ \ empID \ \ IRD\# \gg \\ \ll AP\_id \gg & \rightarrow & \ll AP\_id \ \ year \ \ E\_ID \ \ name \ \ empID \ \ IRD\# \gg \\ \ll PD\_id \gg & \rightarrow & \text{PAYMENTID} \end{array}$$

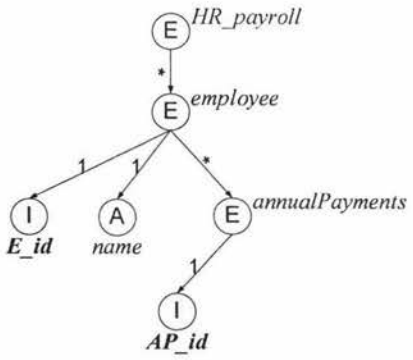
□

Remark. If we allow an XFD to be an expression $X \rightarrow Y$ where X is a possibly empty r_T-subgraph, then we can rewrite the root axiom from Lemma 3.2 such that the noname2 axiom is a generalisation of the rewritten root axiom.

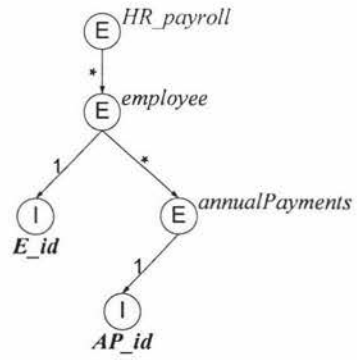
We next extend the notion of a unit of some r_T-walk to account for the presence of identifiers. This then allows us to derive a simple generalisation of the noname rule given in Lemma 3.4.

Definition 4.1. Let X be an r_T-subgraph and B be an r_T-walk of some XML schema graph T. The *unit of B relative to X*, denoted by U_B^X, is the union of all r_T-walks sharing some */+-arc A with B where A is not in X_{ID}. □

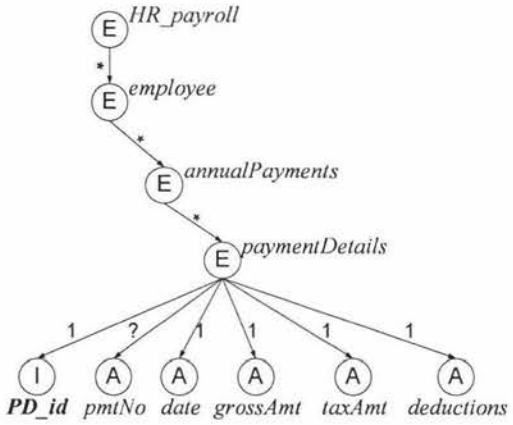
Example 4.3. *Consider the $r_{\text{PAYMENTID}}$-subgraph $\ll$E_id name AP_id$\gg$ in* PAYMENTID. *For convenience, let us denote this simply by X [Figure 24(a)]. Then we have $X_{ID} = \ll$E_id AP_id$\gg$ [Figure 24(b)]. The arc from the vertex with name "annualPayments" to the vertex carrying the name "paymentDetails" is the only */+-arc which is in $\ll$pmtNo$\gg$ but not in X_{ID}. Therefore $U^X_{\ll pmtNo \gg}$ is the $r_{\text{PAYMENTID}}$-subgraph $\ll$PD_id pmtNo date grossAmt taxAmt deductions$\gg$ [Figure 24(c)].* □



(a) $X = \ll$E_id name AP_id$\gg$



(b) $X_{ID} = \ll$E_id AP_id$\gg$



(c) $U^X_{\ll \text{pmtNo} \gg}$

Figure 24: XML trees relating to the computation of the unit of $\ll$pmtNo$\gg$ relative to $\ll$E_id name AP_id$\gg$.

Analogous to U_B, for any r_T-walk $C \in U_B^X$, it is the case that $U_C^X = U_B^X$. It remains to check whether for any two almost copies T_1', T_2' of T in some data tree $T' \rhd T$, it is still true that $T_1'|_{T-U_B^X} \cup T_2'|_{U_B^X}$ and $T_2'|_{T-U_B^X} \cup T_1'|_{U_B^X}$ are almost copies of T in T'.

If X_{ID} is empty, that is, there is no identifier in X, then $U_B^X = U_B$. Obviously in this case, $T_1'|_{T-U_B^X} \cup T_2'|_{U_B^X}$ and $T_2'|_{T-U_B^X} \cup T_1'|_{U_B^X}$ are almost copies of T from the observation we made before. Suppose instead that X_{ID} is non-empty. The above observation is not true unless an additional condition is satisfied. Let $I \in X_{ID}$ be an identifier which shares with each r_T-walk of $U_B - U_B^X$ the last */+-arc which that r_T-walk shares with U_B^X. For any two almost copies of T such that $T_1'|_I = T_2'|_I \cong I$, we can observe that $T_1'|_{T-U_B^X} \cup T_2'|_{U_B^X}$ and $T_2'|_{T-U_B^X} \cup T_1'|_{U_B^X}$ are also almost copies of T. Similar to previously, this mix-and-match is possible because the unique identifier value constraint guarantees that $T_2'|_{U_B^X}$

shares with $T_1'|_{T-U_B^X}$ exactly those arcs (and vertices) which $T_2'|_{U_B^X}$ shares with $T_2'|_{T-U_B^X}$ and, $T_1'|_{U_B^X}$ shares with $T_2'|_{T-U_B^X}$ exactly those arcs which $T_1'|_{U_B^X}$ shares with $T_1'|_{T-U_B^X}$

Example 4.4. *Let us reconsider* $U^X_{\ll pmtNo\gg}$ *from Figure 24(c). It is the case that* $U_{\ll pmtNo\gg}$ = PAYMENTID, *and hence* $U_{\ll pmtNo\gg} - U^X_{\ll pmtNo\gg}$ *is* $\ll E\_id$ *name empID IRD# AP_id year*$\gg$. *For convenience, let us call this subgraph* Y. *The only identifier in* X_{ID} *which shares with each* $r_{\text{PAYMENTID}}$*-walk in* Y *the last */+-arc which that* $r_{\text{PAYMENTID}}$*-walk shares with* $U^X_{\ll pmtNo\gg}$ *is* $\ll AP\_id\gg$. *Therefore, for any two almost copies of* PAYMENTID *which are equivalent and not missing a copy of* $\ll AP\_id\gg$, *it is possible to form two almost copies by exchanging their almost copies of* $U^X_{\ll pmtNo\gg}$ *and* PAYMENTID $- U^X_{\ll pmtNo\gg} = Y$.

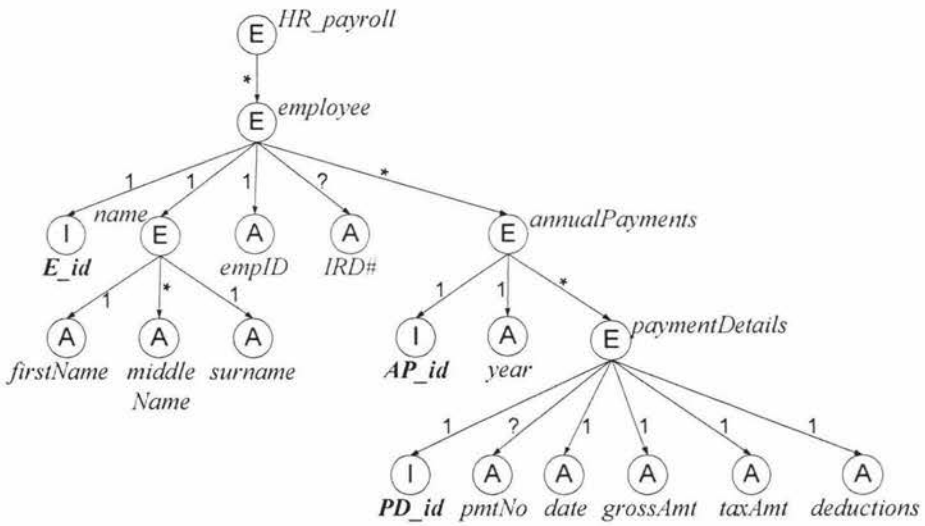


Figure 25: A modified XML schema tree PAYMENTID2

Suppose we modify PAYMENTID *into* PAYMENTID2 *shown in Figure 25. Instead of* $\ll name\gg$, *we model that a name consist of a first name together with potentially many middle names and a surname. Under* PAYMENTID2, *everything in the preceding discussion still holds. In particular,* $U^X_{\ll pmtNo\gg}$ *is the same as shown in Figure 24(c). Again* $\ll AP\_id\gg$ *is still the only identifier in* X_{ID} *which shares with each* $r_{\text{PAYMENTID2}}$*-walk in* $U_{\ll pmtNo\gg} - U^X_{\ll pmtNo\gg}$ *the last */+-arc which that* $r_{\text{PAYMENTID2}}$*-walk shares with* $U^X_{\ll pmtNo\gg}$. *Therefore, for any two almost copies of* PAYMENTID2 *which are equivalent and not missing a copy of* $\ll AP\_id\gg$, *it is possible to form two almost copies by exchanging their almost copies of* $U^X_{\ll pmtNo\gg}$ *and* PAYMENTID2 $- U^X_{\ll pmtNo\gg}$. □

Lemma 4.2. *Let T be an XML schema graph, X an r_T-subgraph of T and B an r_T-walk of T. The* generalised noname rule *defined as follows is sound for the derivation of XFDs:*

$$\frac{\left((X \cup B) \cup T_{\geq 1} - U_B^X\right) \cup X \to B}{X \to B}$$

Proof. The proof is analogous to that of Lemma 3.4. Suppose there is some data tree $T' \rhd T$ with $\models_{T'} \left((X \cup B) \cup T_{\geq 1} - U_B^X\right) \cup X \to B$ but $\not\models_{T'} X \to B$. That is, T' has two almost copies T'_1, T'_2 of T such that $T'_1|_X = T'_2|_X \cong X$ and $T'_1|_B \neq T'_2|_B$. We have $\left((X \cup B) \cup T_{\geq 1} - U_B^X\right) \not\subseteq X$, otherwise $\left((X \cup B) \cup T_{\geq 1} - U_B^X\right) \cup X = X$ which would immediately mean $\models_{T'} X \to B$ by assumption. Therefore there is some non-empty r_T-subgraph $W \subseteq \left((X \cup B) \cup T_{\geq 1} - U_B^X\right)$ such that $W \cap X = \emptyset$. We can then rewrite the generalised noname rule as follows:

$$\frac{\left((X \cup B) \cup T_{\geq 1} - U_B^X\right) \cup X \to B}{X \to B} = \frac{XW \to B}{X \to B}$$

One of T'_1 and T'_2 must not be missing a copy of B and thus not missing a copy of every r_T-walk of $(X \cup B) \cup T_{\geq 1}$. Without loss of generality, assume $T'_1|_B \cong B$ and so $T'_1|_W \cong W$ since $W \subseteq (X \cup B) \cup T_{\geq 1}$. There are now two cases to consider: $T'_1|_W = T'_2|_W \cong W$ and $T'_1|_W \neq T'_2|_W$.

Firstly, suppose $T'_1|_W = T'_2|_W \cong W$. Recall that by assumption $T'_1|_X = T'_2|_X \cong X$, therefore we have $T'_1|_{XW} = T'_2|_{XW} \cong XW$. Since $\models_{T'} XW \to B$ it follows that $T'_1|_B = T'_2|_B$ which contradicts our assumption that $\not\models_{T'} X \to B$.

So, let us consider instead that $T'_1|_W \neq T'_2|_W$. Since $W \subseteq \left((X \cup B) \cup T_{\geq 1} - U_B^X\right)$ $\subseteq T - U_B^X$, it follows that no r_T-walk of W is in U_B^X. If $B \notin U_B^X$ then there is no $*/+$-arc which is in B but not in an r_T-walk of X_{ID}. That is $B \in R$ if X_{ID} is empty, or $B \in X_{ID} \cup T_{\leq 1}$ and application of the noname2 axiom followed by application of the subtree rule states that $\models_{T'} X_{ID} \to B$. In either case, $T'_1|_B = T'_2|_B$ which would contradict our assumption. Hence $B \in U_B^X$. From $X_{ID} \subseteq X$ we obtain $T'_1|_{X_{ID}} = T'_2|_{X_{ID}} \cong X_{ID}$. Therefore we can form two other almost copies of T as follows:

- $T'_3 = T'_1|_{T - U_B^X} \cup T'_2|_{U_B^X}$
- $T'_4 = T'_2|_{T - U_B^X} \cup T'_1|_{U_B^X}$

Consider the pair of almost copies T'_1 and T'_3. We have $T'_1|_W = T'_3|_W$ and altogether $T'_1|_{XW} = T'_3|_{XW} \cong XW$. From $\models_{T'} XW \to B$, it follows that $T'_1|_B = T'_3|_B$. Since $T'_3|_B$ is just $T'_2|_B$ then in fact $T'_1|_B = T'_2|_B$ which contradicts our assumption. □

Note that if X_{ID} is an empty r_T-subgraph then the generalised noname rule reduces to the noname rule of Lemma 3.4.

4.3 A Sound & Complete Rule System

Let the $\mathcal{I}$-rule system for XFDs in the presence of frequencies and identifiers be the set containing the following inference rules: *reflexivity axiom, root axiom, subtree rule, supertree rule, union rule, restricted-transitivity rule, noname2 axiom* and *generalised noname rule.*

Similar to Section 3.2, consider a sequence of restricted pre-closures $X_0, X_1, X_2 \ldots, X_{n-1}, X_n, X_{n+1}, X_{n+2}, \ldots$ where $X_n = X_{n+1} = X_{n+2} \ldots$ for $n \geq 0$, now computed using the $\mathcal{I}$-rule system:

$$\begin{aligned}
X_0 &= \left((X \cup B) \cup T_{\geq 1} - U_B^X\right) \cup X \\
X_1 &= (X_0)_{\mathcal{I}}^+ \cap \left((X \cup B) \cup T_{\geq 1}\right) \\
X_2 &= (X_1)_{\mathcal{I}}^+ \cap \left((X \cup B) \cup T_{\geq 1}\right) \\
&\vdots \\
X_{n-1} &= (X_{n-2})_{\mathcal{I}}^+ \cap \left((X \cup B) \cup T_{\geq 1}\right) \\
X_n &= (X_{n-1})_{\mathcal{I}}^+ \cap \left((X \cup B) \cup T_{\geq 1}\right) \\
&\vdots
\end{aligned}$$

Let $X_{\mathcal{I}}^B = ((X_n \cup B) \cup T_{\geq 1} - U_B^{X_n}) \cup X_n$. Due to the soundness of the generalised noname axiom, if $X_n \rightarrow A \notin \Sigma_{\mathcal{I}}^+$ then $X_{\mathcal{I}}^B \rightarrow A \notin \Sigma_{\mathcal{I}}^+$ or $A \subseteq X_{\mathcal{I}}^B$. In particular, this means that $X_{\mathcal{I}}^B = (X_{\mathcal{I}}^B)_{\mathcal{I}}^+ \cap ((X \cup B) \cup T_{\geq 1})$ and so it is sufficient for us to consider $X_{\mathcal{I}}^B$.

There may be r_T-walks in $X_{\mathcal{I}}^B$ which are not functionally determined by X. Analogous to Section 3.2, we nevertheless construct a counterexample data tree from two almost copies of T which are equivalent on $X_{\mathcal{I}}^B$ in order to ensure that the resulting data tree consists of exactly two almost copies of T. An abstract illustration of the structure of an XML schema graph and a constructed XML data tree compatible with the schema graph is shown in Figure 26 and Figure 27.

Theorem 4.3. *The $\mathcal{I}$-rule system consisting of the* reflexivity axiom, root axiom, subtree rule, supertree rule, union rule, restricted-transitivity rule, noname2 axiom *and* generalised noname rule *is sound and complete for the derivation of XFDs in the presence of frequencies and identifiers.*

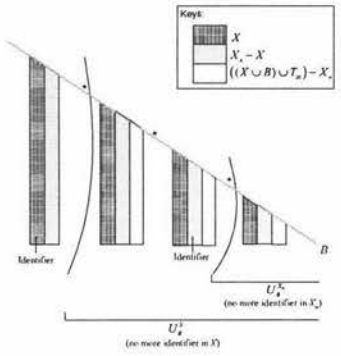


Figure 26: A diagram illustrating the structure of an XML schema graph T and the various subgraphs of $(X \cup B) \cup T_{\geq 1}$ identifiable when computing $X_{\mathcal{I}}^{B}$.

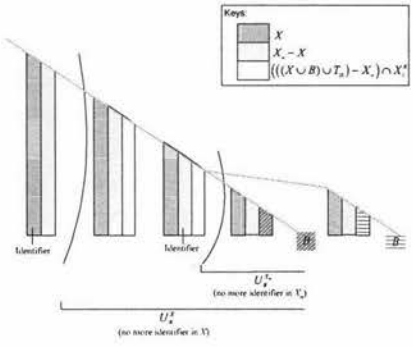


Figure 27: A diagram illustrating the structure of a constructed XML data tree $T' \rhd T$. We use diagonal stripes versus horizontal stripes on the bottom-right to identify two isomorphic but non-equivalent r_T-subgraphs.

Proof. Soundness of the inference rules has already been proven in Lemma 3.1, Lemma 3.2, Lemma 3.3, Lemma 4.1 and Lemma 4.2. It remains to show that the $\mathcal{I}$-rule system is complete in the presence of frequencies and identifiers. Let T be an XML schema graph and Σ a set of XFDs on T. Suppose $X \to Y \notin \Sigma_{\mathcal{I}}^{+}$. That is, there is some r_T-walk $B \in Y$ such that $X \to B$ cannot be derived. We construct a counterexample data tree $T' \rhd T$ such that $\models_{T'} \Sigma$ and $\not\models_{T'} X \to B$.

Consider $X_{\mathcal{I}}^{B}$ as outlined above. Consider the data tree $T' = T'_a \amalg_{[\,(X_n \cup B) \cup T_{\geq 1} - U_B^{X_n}\,]} T'_b$ with T'_a, T'_b being two copies of $(X \cup B) \cup T_{\geq 1}$ such that

$$T'_a|_W = T'_b|_W \text{ if and only if } W \subseteq X_{\mathcal{I}}^{B}$$

This is always possible and the resulting data tree T' will contain exactly two almost copies of T: $T'_1 = T'_a$ and $T'_2 = T'_b$.

Firstly, we need to show that T' satisfies every XFD in Σ. Assume that some $U \to V \in \Sigma$ is not satisfied by T'. Then there must be some r_T-walk $C \in V$ such that $\not\models_{T'} U \to C$. We have $U \subseteq X_{\mathcal{I}}^{B}$ and $C \notin X_{\mathcal{I}}^{B}$ with $C \in ((X \cup B) \cup T_{\geq 1})$, otherwise $U \to C$ cannot be violated. By application of the *supertree rule* to $U \to C \in \Sigma$ we obtain $X_{\mathcal{I}}^{B} \to C \in \Sigma_{\mathcal{I}}^{+}$. Since $C \notin X_{\mathcal{I}}^{B} = ((X_n \cup B) \cup T_{\geq 1} - U_B^{X_n}) \cup X_n$ we have $C \in U_B^{X_n}$. This gives $U_C^{X_n} = U_B^{X_n}$ which implies $((X_n \cup C) \cup T_{\geq 1} - U_C^{X_n}) \cup X_n = X_{\mathcal{I}}^{B}$. Therefore it follows from the *generalised noname rule* that $X_n \to C \in \Sigma_{\mathcal{I}}^{+}$, that is, $C \in (X_n)_{\mathcal{I}}^{+}$. Altogether $C \in (X_n)_{\mathcal{I}}^{+} \cap ((X \cup B) \cup T_{\geq 1}) = X_{n+1} = X_n$, that is, $C \in X_{\mathcal{I}}^{B}$. This is a contradiction hence we may conclude that $\models_{T'} U \to V$.

Secondly, we show that the XFD $X \to Y \notin \Sigma_{\mathcal{I}}^{+}$ is not satisfied by T'. Suppose $B \in X_{\mathcal{I}}^{B}$. Then $X_{\mathcal{I}}^{B} \to B$ and by means of the *generalised noname rule* $X_n \to B$. Clearly $B \in (X \cup B) \cup T_{\geq 1}$ and so $B \in (X_n)_{\mathcal{I}}^{+} \cap ((X \cup B) \cup T_{\geq 1}) = X_{n+1} = X_n$. It follows that $X_{n-1} \to B \in \Sigma_{\mathcal{I}}^{+}$ from the definition of X_n. Further $X_{n-1} \subseteq (X_{n-2})_{\mathcal{I}}^{+}$, therefore it follows that $X_{n-2} \to X_{n-1}$. Also $X_{n-1} \subseteq (X_{n-2} \cup B) \cup T_{\geq 1}$ since $X \subseteq X_{n-2}$ and $X_{n-1} \subseteq (X \cup B) \cup T_{\geq 1}$. This means X_{n-1} is X_{n-2}, B-compliant. Therefore $X_{n-2} \to B \in \Sigma_{\mathcal{I}}^{+}$ is derived by means of the *restricted-transitivity rule*. Following the same reasoning, we get $X_{n-3} \to B \in \Sigma_{\mathcal{I}}^{+}$, then $X_{n-4} \to B \in \Sigma_{\mathcal{I}}^{+}$, and so on until eventually we obtain $X_0 \to B \in \Sigma_{\mathcal{I}}^{+}$. $X_0 = ((X \cup B) \cup T_{\geq 1} - U_B^X) \cup X$ and so another application of the *generalised noname rule* yields $X \to B \in \Sigma_{\mathcal{I}}^{+}$ which contradicts our assumption. Hence $B \notin X_{\mathcal{I}}^{B}$. Then by construction $\not\models_{T'} X \to B$ and thus $\not\models_{T'} X \to Y$. □

5 XML with Identifiers Normal Form

In addition to the axiomatisation of functional dependencies, another important objective of design theory is the design of "good" database schemas. For various data models, a primary justification for a schema being "good" (i.e. a desirable property), apart from the absence of update anomalies and simplified integrity checking, is the absence of redundancy. Normal forms have been proposed as syntactic characterisations with which to determine whether a schema has certain desirable properties. In this section, we derive a new XML normal form for XFDs in the presence of frequencies and identifiers which is both necessary and sufficient for the absence of redundancy. We will define a notion of redundancy in Section 5.2, however we begin with a discussion of trivial XFDs.

5.1 Trivial XFDs

An XFD on an XML schema graph T is called *trivial* if and only if they are satisfied by every XML data tree $T' \rhd T$.

Proposition 5.1. *Let T be an XML schema tree and X, Y be two non-empty r_T-subgraphs in T. An XFD $X \to Y$ on T is trivial if and only if one of the following conditions holds:*

1) $Y \subseteq X$ *2)* $Y \subseteq X_{ID} \cup T_{\leq 1}$

Proof. (If) For each of the cases above, we show that every data tree $T' \rhd T$ satisfies the XFD $X \to Y$ and is therefore trivial. Firstly, if $Y \subseteq X$, the *reflexivity axiom* states that every data tree $T' \rhd T$ satisfies the XFD $X \to Y$.

For the second case where $Y \subseteq X_{ID} \cup T_{\leq 1}$. If X_{ID} is empty then $Y \subseteq R$. The *root axiom* states that every data tree satisfies the XFD $X \to R$. Since $Y \subseteq R$, the *subtree rule* further states that every data tree $T' \rhd T$ also satisfies the XFD $X \to Y$. Suppose instead that X_{ID} is non-empty, then the *noname2 axiom* states that every data tree satisfies the XFD $X_{ID} \to X_{ID} \cup T_{\leq 1}$. Since $X_{ID} \subseteq X$, by *supertree rule*, we have that every data tree also satisfies $X \to X_{ID} \cup T_{\leq 1}$. Finally since $Y \subseteq X_{ID} \cup T_{\leq 1}$, the *subtree rule* again states that every data tree also satisfies $X \to Y$.

Under each condition, we have shown that $X \to Y$ is a trivial dependency.

(Only If) It remains to show that all trivial XFDs must satisfy at least one of the specified conditions. Assume that $X \to Y$ is a trivial XFD on T and none of the two conditions hold, that is, $Y \not\subseteq X$ and $Y \not\subseteq X_{ID} \cup T_{\leq 1}$.

We need to construct an XML data tree $T' \rhd T$ in which $X \to Y$ is violated. Consider the data tree $T' = T'_a \amalg_{[X_{ID} \cup T_{\leq 1}]} T'_b$ with T'_a, T'_b being copies of T such that

$$T'_a|_W = T'_b|_W \text{ if and only if } W \subseteq X \text{ or } W \subseteq X_{ID} \cup T_{\leq 1}$$

Observe that the amalgamation is always possible because every r_T-walk not in $X_{ID} \cup T_{\leq 1}$ must contain at least one */+-arc and T'_a, T'_b may only be non-equivalent on those r_T-walks.

Since $Y \not\subseteq X$ and $Y \not\subseteq X_{ID} \cup T_{\leq 1}$ hold, it is clear that $\not\models_{T'} X \to Y$. Therefore by contradiction we may conclude that trivial XFDs must satisfy one of the two conditions. □

5.2 Redundancy with respect to XFDs

In the RDM, redundancy with respect to functional dependencies considers each functional dependency to be describing a basic unit of information when retrieving or updating data. A relation which satisfies a given set of functional dependencies, and contains two tuples identical on such a unit of information is said to be redundant. If we simply rewrite this notion for the XML graph model, we obtain the following definition for redundancy:

> Let T be an XML schema graph and Σ a set of XFDs on T. We say T *is redundant with respect to* Σ if and only if there is some XML data tree $T' \rhd T$ such that $\models_{T'} \Sigma$ and for some non-trivial XFD $X \to Y \in \Sigma$ with r_T-walk $B \in Y$, there exists two almost copies T'_1, T'_2 of T in T' with $T'_1|_{XB} = T'_2|_{XB} \cong XB$.

Example 5.1. *All XFDs in* Σ(PAYMENTID) *on* PAYMENTID *[Figure 22] are non-trivial. Consider the XFD:*

$$(P\_XFD1) \quad \ll empID \gg \to \ll E\_id \gg$$

The two almost copies of PAYMENTID *in* PAYMENTID' *shown in Figure 28 are equivalent and not missing a copy of both* $r_{\text{PAYMENTID}}$*-walk of* $\ll empID \quad E\_id \gg$ *. This means* P_XFD1 *causes* PAYMENT *to be redundant with respect to* Σ(PAYMENTID) *according the above definition. However the two almost copies actually share the same copy of* $\ll empID \quad E\_id \gg$ *in* PAYMENTID'. □

Example 5.1 illustrates that the above definition is not quite adequate in the framework of XML. The unit of information is not redundant since we cannot delete it from either

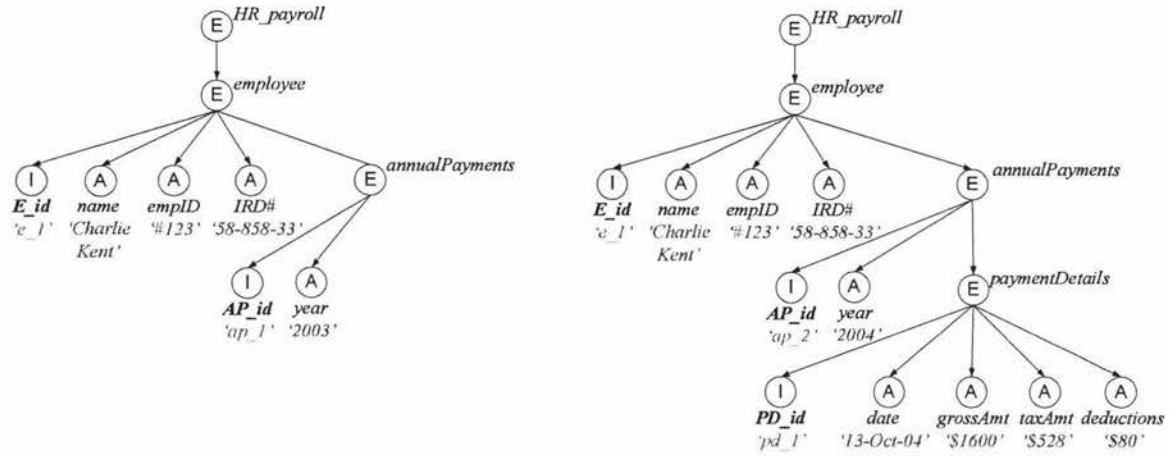


Figure 28: Two almost copies of PAYMENTID in PAYMENTID'.

almost copies without losing information. Similar to nested relations, the hierarchical structure in XML enables us to avoid some redundancy which would be unavoidable in the RDM. In order to adapt a notion of redundancy, we need the additional requirement that the two almost copies must not share the same almost copy of the right-hand-side of the XFD, that is, they do not coincide [Definition 3.4] on the right-hand-side.

Definition 5.1. Let T be an XML schema graph and Σ a set of XFDs on T. We say *T is redundant with respect to Σ* if and only if there is some XML data tree $T' \rhd T$ such that $\models_{T'} \Sigma$ and for some non-trivial XFD $X \to Y \in \Sigma$ with r_T-walk $B \in Y - X$ there exists two almost copies T'_1, T'_2 of T in T' with $T'_1|_{XB} = T'_2|_{XB} \cong XB$ but T'_1 and T'_2 do not coincide on B. □

Example 5.2. *According to Definition 5.1 the XFD*

$$(P\_XFD1) \quad \ll empID \gg \to \ll E\_id \gg$$

is no longer seen to cause PAYMENTID *to be redundant with respect to* Σ(PAYMENTID).

If the right-hand-side of an XFD contains only identifiers then the XFD cannot cause an XML schema tree to be redundant due to the unique identifier value constraint. So the only XFD in Σ(PAYMENTID) *that may cause some concern is*

$$(P\_XFD2) \quad \ll IRD\# \gg \to \ll empID \gg$$

Since $\ll empID \gg$ *is* $\ll IRD\# \gg, \ll E\_id \gg$*-compliant, by means of the restricted-transitivity rule we can derive* $\ll IRD\# \gg \to \ll E\_id \gg$ *where* $\ll E\_id \gg$ *is an identifier. It is easy to see that for any data tree compatible with* PAYMENTID, *for any*

two almost copies of PAYMENTID *which are equivalent and not missing a copy of* $\ll IRD\#\ \ empID\gg$ *, they must be equivalent to and not missing a copy of* $\ll E\_id\gg$ *. In fact, because* $\ll E\_id\gg$ *is an identifier, any two almost copies of* PAYMENTID *coincide on* $\ll E\_id\gg$ *. Since* $\ll empID\gg \in \ll E\_id\gg \cup \text{PAYMENTID}_{\leq 1}$*, the two almost copies must also coincide on* $\ll empID\gg$ *. This means* PAYMENTID *is non-redundant with respect to* Σ(PAYMENTID). □

Example 5.3. *Now consider the XML schema tree* BANKID *from Figure 21. Suppose we have the XML data tree modified from the one in Figure 13 such that* Σ(BANKID) *and the unique identifier value constraint are satisfied. The resulting data tree would contain information about two employees who have supplied the same bank account details. One explanation for this would be that a married couple working in the company have both given their joint-account details. Therefore,* B_XFD6 *to* B_XFD13 *cause* BANKID *to be redundant with respect to* Σ(BANKID). □

Similar to the RDM, we can prove that a schema graph T is redundant with respect to Σ precisely if T is redundant with respect to Σ^*. In other words, a schema graph being redundant is invariant under the replacement of a set of XFDs by an equivalent set. It should be noted that according to Theorem 4.3, $\Sigma^* = \Sigma_{\mathcal{I}}^+$ in the presence of frequencies and identifiers.

Theorem 5.1. *Let* T *be an XML schema graph and* Σ *a set of XFDs on* T*. Then* T *is redundant with respect to* Σ *if and only if* T *is redundant with respect to* Σ^**.*

Proof. (Only If) The redundancy of T with respect to Σ^* whenever T is redundant with respect to Σ is obvious since $\Sigma \subseteq \Sigma^*$, and $\models_{T'} \Sigma$ implies $\models_{T'} \Sigma^*$ by definition of Σ^*.

(If) It remains to show that redundancy of T with respect to Σ^* also implies that T is redundant with respect to Σ. Consider the chain

$$\Sigma = \Sigma_0 \subset \Sigma_1 \subset \ldots \subset \Sigma_k = \Sigma_{\mathcal{I}}^+ = \Sigma^*$$

where Σ_{i+1} results from Σ_i for $i \geq 0$, by a single application of one of the inference rules from the $\mathcal{I}$-rule system. We show that there is already some XFD in Σ_i which causes T to be redundant with respect to Σ_i whenever T is redundant with respect to Σ_{i+1} for any $i \geq 0$.

Assume that T is redundant with respect to Σ_{i+1}. This means that some non-trivial XFD $X \rightarrow Y \in \Sigma_{i+1}$ which causes T to be redundant with respect to Σ_{i+1}. Further, for some r_T-walk $B \in Y - X$, there is an XML data tree $T' \rhd T$ such that $\models_{T'} \Sigma_{i+1}$ which contains two almost copies T_1', T_2' of T such that $T_1'|_{XB} = T_2'|_{XB} \cong XB$ but T_1' and T_2' do

not coincide on B. If $X \to Y \in \Sigma_i$, then there is nothing to show. It remains to look at $X \to Y \in \Sigma_{i+1} - \Sigma_i$.

Clearly $\models_{T'} \Sigma_{i+1}$ implies $\models_{T'} \Sigma_i$ since $\Sigma_i \subset \Sigma_{i+1}$. Since $X \to Y$ is non-trivial, Proposition 5.1 implies that it has not been derived by means of the *reflexivity axiom*, *root axiom* or *noname2 axiom*. We proceed by considering the cases where $X \to Y$ has been derived by means of any of the remaining axioms in the $\mathcal{I}$-rule system.

Assume that $X \to Y$ has been derived using the *union rule*, i.e. $Y = W \cup Z$ and $X \to W, X \to Z \in \Sigma_i$. It must be that $B \in W - X$ or $B \in Z - X$. If $B \in W - X$ then $X \to W$ causes T to be redundant with respect to Σ_i. Similarly if $B \in Z - X$ then $X \to Z$ causes T to be redundant with respect to Σ_i. In either case, T is redundant with respect to Σ_i.

Next assume that $X \to Y$ has been derived by means of the *subtree rule*, i.e. $Y \subseteq W$ with $X \to W \in \Sigma_i$. From $B \in Y - X$ and $Y \subseteq W$ we infer $B \in W - X$. It immediately follows that T is redundant with respect to Σ_i due to $X \to W$.

Thirdly, suppose that $X \to Y$ has been derived using the *supertree rule*, i.e. $W \subseteq X$ with $W \to Y \in \Sigma_i$. Since $W \subseteq X$ and $T_1'|_X = T_2'|_X \cong X$, it is the case that $T_1'|_W = T_2'|_W \cong W$. From $B \in Y - X$ and $W \subseteq X$ we infer $B \in Y - W$. Therefore T is redundant with respect to Σ_i because of $W \to Y$.

Suppose $X \to Y$ has been derived by means of the *restricted-transitivity rule*, i.e. W is X, Y-compliant and $X \to W, W \to Y \in \Sigma_i$. Recall that W is X, Y-compliant if and only if $W \subseteq (X \cup C) \cup T_{\geq 1}$ for each r_T-walk $C \in Y$. In particular, we have $W \subseteq (X \cup B) \cup T_{\geq 1}$. From this and the fact that $\models_{T'} X \to W$ we get $T_1'|_W = T_2'|_W \cong W$. Either $B \notin W$ or $B \in W$. If $B \notin W$ then $B \in Y - W$ and it immediately follows that $W \to Y$ causes T to be redundant with respect to Σ_i. Suppose instead that $B \in W$. From assumption $B \notin X$, therefore $B \in W - X$, in which case $X \to W$ causes T to be redundant with respect to Σ_i.

Finally, assume that $X \to Y$ has been derived using the *generalised noname rule*, i.e. $X' = ((X \cup Y) \cup T_{\geq 1} - U_Y^X) \cup X$ with $Y = B$ being a single r_T-walk and $X' \to Y \in \Sigma_i$. Rewriting our initial assumption gives $T_1'|_{XY} = T_2'|_{XY} \cong XY$ but T_1' and T_2' do not coincide on Y, and $Y \notin X$. If $Y \in X'$ then there is no $*/+$-arc which is in Y but not in X_{ID}. That is $Y \in X_{ID} \cup T_{\leq 1}$meaning that $X \to Y$ is trivial by Proposition 5.1. It therefore remains to consider the case $Y \notin X'$.

T_1' and T_2' are not missing a copy of every r_T-walk of X' since $X' \subseteq (X \cup Y) \cup T_{\geq 1}$ and neither a copy of X nor a copy of Y are missing from either almost copies. It

is the case that either $T_1'|_{X'} = T_2'|_{X'}$ or $T_1'|_{X'} \neq T_2'|_{X'}$. In the first case, we know that $X' \to Y$ causes T to be redundant with respect to Σ_i. So consider the latter case where $T_1'|_{X'} \neq T_2'|_{X'}$. Since $X_{ID} \subseteq X$ we have $T_1'|_{X_{ID}} = T_2'|_{X_{ID}} \cong X_{ID}$. This allows us to apply the observation from Section 4.2, that is, from T_1' and T_2' we obtain another almost copy of T, namely $T_3' = T_1'|_{T-U_Y^X} \cup T_2'|_{U_Y^X}$. It is easy to see that $T_1'|_{X'} = T_3'|_{X'} \cong X'$ since $X' \subseteq (T - U_Y^X) \cup X$. From $Y \notin X'$ we infer $Y \in U_Y^X$. This means $T_3'|_Y$ is just $T_2'|_Y$, so it follows that T_1' and T_3' do not coincide on Y. But this means that $X' \to Y$ causes T to be redundant with respect to Σ_i. □

5.3 X^{i}NF: An XML Normal Form Utilising Identifiers

In the RDM, a syntactic characterisation of a relation schema being non-redundant with respect to a set of FDs is the Boyce-Codd Normal Form (BCNF). The definition of BCNF is not easily adaptable to the XML framework due to the difficulty of adapting the notion of keys from the RDM to XML. As far as we are aware, the notion of keys in XML (e.g. [1, 7, 8, 12]) currently only provide a means for identifying and referencing specific vertices.

Our proposed normal form for XML definition avoids referring to any notion of keys in XML. Instead, we propose to make use of the uniqueness of identifier values in order to guarantee non-redundancy. Consider a non-trivial XFD of the form $X \to Y$. Our proposed normal form requires that $Y \in I \cup T_{\leq 1}$ for some identifier I, where for any two almost copies T_1', T_2' of the schema tree such that $T_1'|_X = T_2'|_X \cong X$ it is the case that T_1', T_2' are equivalent and not missing a copy of I. Due to the uniqueness of identifier value, T_1' and T_2' would then necessarily coincide on the identifier and thus also coincide on Y. Clearly the identifier I can be in $X_{\mathcal{I}}^+$, but I can also be in $X_{\mathcal{I}}^B$ defined as follows.

Let $X_{\mathcal{I}}^B = ((X_n \cup B) \cup T_{\geq 1} - U_B^{X_n}) \cup X_n$ where the sequence $X_0, X_1, X_2, \ldots, X_{n-1}, X_n, X_{n+1}, \ldots$ of restricted pre-closures with $X_n = X_{n+1} = X_{n+2} = \ldots$ is computed as follows:

$$
\begin{aligned}
X_0 &= ((X \cup B) \cup T_{\geq 1} - U_B^X) \cup X \\
X_1 &= (X_0)_{\mathcal{I}}^+ \cap ((X \cup B) \cup T_{\geq 1}) \\
X_2 &= (X_1)_{\mathcal{I}}^+ \cap ((X \cup B) \cup T_{\geq 1}) \\
&\vdots \\
X_{n-1} &= (X_{n-2})_{\mathcal{I}}^+ \cap ((X \cup B) \cup T_{\geq 1}) \\
X_n &= (X_{n-1})_{\mathcal{I}}^+ \cap ((X \cup B) \cup T_{\geq 1}) \\
&\vdots
\end{aligned}
$$

Definition 5.2. Let T be an XML schema graph, X an r_T-subgraph and B an r_T-walk of T. B is said to be *relatively identified by* X if and only if there exists some identifier I in T such that $I \in X_{\mathcal{I}}^B$ and $B \in I \cup T_{\leq 1}$. □

Remark. For an XFD $X \to B$, if B is an identifier then B is relatively identified by X.

Definition 5.3. Let T be an XML schema graph and Σ a set of XFDs on T. We say that T *is in XML with Identifiers Normal Form (X^{i}NF) with respect to* Σ if and only if for every non-trivial XFD $X \to B \in \Sigma^*$ where B is a single r_T-walk of T, it is the case that B is relatively identified by X. □

Remark. If there are no identifiers in T then T is in X^{i}NF with respect to Σ if and only if every XFD $X \to B \in \Sigma^*$ is trivial.

With the following Lemma and Theorem we show that X^{i}NF is a necessary and sufficient condition for non-redundancy.

Lemma 5.2. *Let T be an XML schema graph and Σ a set of XFDs on T. Consider a data tree $T' \rhd T$ such that $\models_{T'} \Sigma$ and the sequence of restricted pre-closures used to compute $X_{\mathcal{I}}^B$ (see above). For every r_T-walk $A \in X_n - X_0$, any two almost copies T'_1, T'_2 of T with $T'_1|_X = T'_2|_X \cong X$ and $T'_i|_B \cong B$ for $i = 1$ or 2, it is the case that $T'_1|_A = T'_2|_A \cong A$.*

Proof. Let $A \in X_n - X_0$. From $A \notin X_0 = ((X \cup B) \cup T_{\geq 1} - U_B^X) \cup X$ we infer that $A \in U_B^X$. Without loss of generality assume that $T'_1|_B \cong B$. This means $T'_1|_{X_0} \cong X_0$ since T'_1 is not missing a copy of every r_T-walk of X and B. If $T'_1|_{X_0} = T'_2|_{X_0} \cong X_0$ then $T'_1|_{X_n} = T'_2|_{X_n} \cong X_n$ and of course $T'_1|_A = T'_2|_A \cong A$. Therefore suppose $T'_1|_{X_0} \neq T'_2|_{X_0}$. Since $T'_1|_{X_{ID}} = T'_2|_{X_{ID}} \cong X_{ID}$ we can consider a third almost copy of T, namely $T'_3 = T'_1|_{T-U_B^X} \cup T'_2|_{U_B^X}$. We now have $T'_1|_{X_0} = T'_3|_{X_0} \cong X_0$ and thus $T'_1|_{X_n} = T'_3|_{X_n} \cong X_n$. In particular, this means $T'_1|_A = T'_3|_A \cong A$. But $T'_3|_A$ is just $T'_2|_A$ so in fact $T'_1|_A = T'_2|_A \cong A$. □

Theorem 5.3. *Let T be an XML schema graph and Σ a set of XFDs on T. Then T is in X^{i}NF with respect to Σ if and only if T is non-redundant with respect to Σ.*

Proof. (Only If) Let T be in X^{i}NF and assume that T is redundant with respect to Σ. A direct implication of Theorem 5.1 is that T is redundant with respect to Σ^*. Therefore there is some XML data tree $T' \rhd T$ such that $\models_{T'} \Sigma$ and for some non-trivial XFD $X \to B \in \Sigma^*$ with r_T-walk $B \notin X$, there exists two almost copies T'_1, T'_2 of T in T' with $T'_1|_{XB} = T'_2|_{XB} \cong XB$ but T'_1 and T'_2 do not coincide on B. We can assume that B is a single r_T-walk without loss of generality. Since T is in X^{i}NF and $X \to B$ is non-trivial,

we have B is relatively identified by X. That is, there is some identifier I in T such that $I \in X_{\mathcal{I}}^B$ and $B \in I \cup T_{\leq 1}$.

If $T_1'|_I = T_2'|_I \cong I$ then T_1' and T_2' must coincide on I because of the unique identifier value constraint. Then $B \in I \cup T_{\leq 1}$ would further imply that T_1' and T_2' coincide on B. This would contradict our assumption, therefore assume $T_1'|_I \neq T_2'|_I$. As a direct result of Lemma 5.2, $I \notin X_n - X_0$. Moreover $I \not\subseteq X$ since $T_1'|_X = T_2'|_X \cong X$ by assumption. This leaves the cases where either (1) $I \in X_0 - X$, or (2) $I \in X_{\mathcal{I}}^B - X_n$.

Firstly suppose $I \in X_0 - X$, that is $I \in ((X \cup B) \cup T_{\geq 1} - U_B^X) - X$. In particular $I \notin U_B^X$ and therefore $B \notin U_B^X$ since $B \in I \cup T_{\leq 1}$. This means that there is no */+-arc which is in B but not in X_{ID}. If X_{ID} is empty then there is no */+-arc in B and $B \in R$. But then $X \to B$ would be trivial and also T_1', T_2' would coincide on B thus contradicting our assumptions. Hence there is some identifier $J \in X_{ID} \subseteq X$ such that $B \in J \cup T_{\leq 1}$. Clearly $T_1'|_J = T_2'|_J \cong J$. Then T_1' and T_2' coincide on J and therefore also coincide on B because $B \in J \cup T_{\leq 1}$. This is a contradiction to our assumption that there are two almost copies T_1', T_2' of T in T' with $T_1'|_{XB} = T_2'|_{XB} \cong XB$ but T_1' and T_2' not coinciding on B.

Now consider the case where $I \in X_{\mathcal{I}}^B - X_n$, that is $I \in ((X_n \cup B) \cup T_{\geq 1} - U_B^{X_n}) - X_n$. Similar to before, $I \notin U_B^{X_n}$ so $B \notin U_B^{X_n}$ and there is no */+-arc in B but not in $(X_n)_{ID}$. Like above, $(X_n)_{ID}$ must not be empty and there is some identifier $J \in (X_n)_{ID}$ with $B \in J \cup T_{\leq 1}$. Either $J \in X_n - X_0$ or $J \in X_0 - X$ or $J \in X$.

For $J \in X$, it is obvious that $T_1'|_J = T_2'|_J \cong J$. If $J \in X_n - X_0$ then $T_1'|_J = T_2'|_J \cong J$ by Lemma 5.2. By the same argument as above, both cases result in T_1' and T_2' coinciding on J and thus coinciding on B. Finally consider the case where $J \in X_0 - X$. This is just the first case discussed above. Following the same argument as previously, we conclude that T_1' and T_2' coincide on B.

In each case we obtain a contradiction, therefore it must be that T is non-redundant with respect to Σ.

(If) On the other hand, we need to show that if T is not in X^{i}NF with respect to Σ then T is redundant with respect to Σ. Assume that T is not in X^{i}NF with respect to Σ. Let $X \to B \in \Sigma^*$ be a non-trivial XFD where B is not relatively identified by X. We show that it is possible to construct an XML data tree $T' \rhd T$ witnessing that T is redundant with respect to Σ^*. More specifically, it will be the case that $\models_{T'} \Sigma$ and T' contains two almost copies T_1', T_2' such that $T_1'|_{XB} = T_2'|_{XB} \cong XB$ but T_1', T_2' do not coincide on B.

Recall the XML data tree construction from Theorem 4.3: namely $T' = T_a' \amalg_{[(X_n \cup B) \cup T_{\geq 1} - U_B^{X_n}]} T_b'$ with T_a', T_b' being two copies of $(X \cup B) \cup T_{\geq 1}$ such

that

$$T'_a|_W = T'_b|_W \text{ if and only if } W \subseteq X_{\mathcal{I}}^B$$

The resulting data tree T' contains exactly two almost copies of T: $T'_1 = T'_a$ and $T'_2 = T'_b$.

Analogous to the proof of Theorem 4.3, we can show that $\models_{T'} \Sigma$. Obviously $X \subseteq X_{\mathcal{I}}^B$ and so $T'_1|_X = T'_2|_X \cong X$. It follows from $X \to B \in \Sigma^*$ that $B \in X_{\mathcal{I}}^B$ and therefore $T'_1|_B = T'_2|_B \cong B$. Suppose $B \in (X_n \cup B) \cup T_{\geq 1} - U_B^{X_n}$. Then $B \notin U_B^{X_n}$ and it follows that there is no */+-arc in B but not in $(X_n)_{ID}$. In other words $B \in (X_n)_{ID} \cup T_{\leq 1}$. Since $(X_n)_{ID} \subseteq X_{\mathcal{I}}^B$, if $(X_n)_{ID}$ is non-empty then there is some identifier $I \in (X_n)_{ID} \subseteq X_{\mathcal{I}}^B$ such that $B \in I \cup T_{\leq 1}$. This means B would be relatively identified by X which would contradict our assumption. On the other hand, if $(X_n)_{ID}$ is empty then there is actually no */+-arc in B and $B \in R \subseteq (X_n)_{ID} \cup T_{\leq 1}$. This would contradict that $X \to B$ is non-trivial. Hence $B \notin (X_n \cup B) \cup T_{\geq 1} - U_B^{X_n}$. In particular $B \in U_B^{X_n}$ and therefore by construction T'_1 and T'_2 do not coincide on B. □

5.4 Elegantly Checking X^{i}NF

According to Definition 5.3, in order to check whether an XML schema graph T is in X^{i}NF we need to examine all XFDs in Σ^* with a single r_T-walk on the right-hand-side. This is rather impractical because we must first compute Σ^*, therefore we investigate a more elegant approach to check for X^{i}NF.

Let Σ be a set of XFDs and let $X^\Sigma = \bigcup\{Y \mid X \to Y \in \Sigma\}$ denote the union of r_T-subgraphs specified in Σ to be functionally determined by X . This allows us to restate the definition of X^{i}NF as follows: for every r_T-subgraph X such that there is an XFD $X \to Y \in \Sigma^*$ and for every r_T-walk $B \in X^{\Sigma^*}$ if $X \to B$ is non-trivial then B is relatively identified by X. In the next theorem, we show that in order to determine whether or not T is in X^{i}NF, it is sufficient to examine all XFDs in Σ. That is, we would need to check that for every r_T-subgraph X such that there is an XFD $X \to Y \in \Sigma$, and for every r_T-walk $B \in X^\Sigma$ if $X \to B$ is non-trivial then B is relatively identified by X.

We say an XFD $X \to Y \in \Sigma$ *causes T to violate X^{i}NF* if there is an r_T-walk $B \in Y$ such that $X \to B \in \Sigma^*$ is non-trivial and B is not relatively identified by X. Naturally, there is no XFD $X \to Y \in \Sigma^*$ which causes T to violate X^{i}NF if and only if T is in X^{i}NF with respect to Σ. Similarly, there is no XFD $X \to Y \in \Sigma$ which causes T to violate X^{i}NF if and only if for every r_T-subgraph X such that there is an XFD $X \to Y \in \Sigma$, and for every r_T-walk $B \in X^\Sigma$ if $X \to B$ is non-trivial then B is relatively identified by

X. This alternative characterisation of X^{i}NF enables us to provide a simple proof for the next theorem.

Theorem 5.4. *Let T be an XML schema graph and Σ a set of XFDs on T. T is in X^{i}NF with respect to Σ if and only if for every r_T-subgraph X such that there is an XFD $X \rightarrow Y \in \Sigma$, and for every r_T-walk $B \in X^{\Sigma}$ if $X \rightarrow B$ is non-trivial then B is relatively identified by X.*

Proof. (Only If) Since $\Sigma \subseteq \Sigma^*$, it is easy to see that if no XFD in Σ^* causes T to violate X^{i}NF, it must be the case that no XFD in Σ causes T to violate X^{i}NF.

(If) We need to show that if no XFD in Σ causes T to violate X^{i}NF then no XFD in Σ^* causes T to violate X^{i}NF. Let there be a chain

$$\Sigma = \Sigma_0 \subset \Sigma_1 \subset \ldots \subset \Sigma_k = \Sigma_{\mathcal{I}}^+ = \Sigma^*$$

where Σ_{i+1} results from $\Sigma_i, i \geq 0$ by a single application of one of the inference rules in the $\mathcal{I}$-rule system. We will show that if there is an XFD $X \rightarrow Y \in \Sigma_{i+1}$ which causes T to violate X^{i}NF, then there is already some XFD in Σ_i which causes T to violate X^{i}NF.

Assume that no XFD in Σ_i causes T to violate X^{i}NF while some XFD in Σ_{i+1} causes T to violate X^{i}NF. This means the single XFD $X \rightarrow Y \in \Sigma_{i+1} - \Sigma_i$ with $i \geq 0$ causes T to violate X^{i}NF. That is, there is an r_T-walk $B \in Y$ such that $X \rightarrow B$ is non-trivial and B is not relatively identified by X. Since $X \rightarrow B$ is non-trivial $X \rightarrow Y$ must be non-trivial. Therefore $X \rightarrow Y$ has not been derived by means of the *reflexivity axiom*, *root axiom* or *noname2 axiom* in accordance with Proposition 5.1. We now consider the cases where $X \rightarrow Y$ has been derived by any of the remaining inference rules in the $\mathcal{I}$-rule system .

Assume that $X \rightarrow Y$ has been derived by means of the *union rule*, i.e. $Y = W \cup Z$ and $X \rightarrow W, X \rightarrow Z \in \Sigma_i$. We have $B \in W$ or $B \in Z$. Due to the initial assumptions, if $B \in W$ then $X \rightarrow W \in \Sigma_i$ causes T to violate X^{i}NF. Similarly, if $B \in Z$ then $X \rightarrow Z \in \Sigma_i$ causes T to violate X^{i}NF.

Secondly assume that $X \rightarrow Y$ has been derived using the *subtree rule*, i.e. $Y \subseteq Y'$ and $X \rightarrow Y' \in \Sigma_i$. Since $Y \subseteq Y'$ and $B \in Y$, it follows that $B \in Y'$. Again the initial assumptions immediately means $X \rightarrow Y' \in \Sigma_i$ causes T to violate X^{i}NF.

The remainder of the proof will utilise the following observation. Let V and W be two r_T-subgraphs in T. If $V \subseteq W$ then it follows from $V_{ID} \subseteq W_{ID}$ that $U_B^W \subseteq U_B^V$. Further, we have $(V \cup B) \cup T_{\geq 1} \subseteq (W \cup B) \cup T_{\geq 1}$ and thus $V_0 = ((V \cup B) \cup T_{\geq 1} - U_B^V) \cup V \subseteq ((W \cup B) \cup T_{\geq 1} - U_B^W) \cup W = W_0$. It immediately follows that $V_{\mathcal{I}}^B \subseteq W_{\mathcal{I}}^B$.

Suppose $X \to Y$ has been derived using the *supertree rule*, i.e. $X' \subseteq X$ and $X' \to Y \in \Sigma_i$. Suppose B is relatively identified by X', that is, there is some identifier I such that $I \in (X')^B_{\mathcal{I}}$ and $B \in I \cup T_{\leq 1}$. The above observation yields $(X')^B_{\mathcal{I}} \subseteq X^B_{\mathcal{I}}$ from $X' \subseteq X$. Hence $I \in X^B_{\mathcal{I}}$. This would mean B is relatively identified by X, contradicting our assumption. Hence B is not relatively identified by X'. But this means that $X' \to Y \in \Sigma_i$ causes T to violate X^{i}NF.

Next assume $X \to Y$ has been derived by means of the *restricted-transitivity rule*, i.e. W is X, Y-compliant and $X \to W, W \to Y \in \Sigma_i$. Recall that W being X, Y-compliant implies $W \subseteq (X \cup B) \cup T_{\geq 1}$. Together with $X \to W \in \Sigma_i$ we get $W \subseteq X_n$ where X_n occurs in the sequence of restricted pre-closures used to compute $X^B_{\mathcal{I}}$. The observation above gives $W^B_{\mathcal{I}} \subseteq (X_n)^B_{\mathcal{I}} = X^B_{\mathcal{I}}$. Suppose B is relatively identified by W. Then there is some identifier $I \in W^B_{\mathcal{I}}$ such that $B \in I \cup T_{\leq 1}$. It follows that $I \in X^B_{\mathcal{I}}$ and B is relatively identified by X which would be a contradiction. Hence B is not relatively identified by W, in which case $W \to Y \in \Sigma_i$ causes T to violate X^{i}NF.

Finally, suppose that $X \to Y$ has been derived using the *generalised noname rule*, i.e. $X' = ((X \cup Y) \cup T_{\geq 1} - U^X_Y) \cup X$ with $Y = B$ being a single r_T-walk of T and $X' \to Y \in \Sigma_i$. We have $X' = ((X \cup Y) \cup T_{\geq 1} - U^X_Y) \cup X = X_0$. In particular, $X' \subseteq X_0$ and therefore $(X')^B_{\mathcal{I}} \subseteq (X_0)^B_{\mathcal{I}} = X^B_{\mathcal{I}}$. Suppose B is relatively identified by X', that is, there is some identifier $I \in (X')^B_{\mathcal{I}}$ with $B \in I \cup T_{\leq 1}$. Consequently $I \in X^B_{\mathcal{I}}$ and B is relatively identified by X, a contradiction. Therefore B is not relatively identified by X'. Then $X' \to Y \in \Sigma_i$ causes T to violate X^{i}NF. □

Example 5.4. *Recall that in Example 5.2 we observed that all XFDs in* Σ(PAYMENTID) *apart from P_XFD2 contain only identifiers on the right-hand-side. Further, we saw that* $\ll IRD\# \gg \to \ll E\_id \gg$ *is derivable from* Σ(PAYMENTID). *It is the case that* $(\ll IRD\# \gg \cup \ll empID \gg) \cup \text{PAYMENTID}_{\geq 1}$ *includes* $\ll E\_id \gg$. *Particularly, this means* $\ll E\_id \gg$ *will be in* $(\ll IRD\# \gg)^{\ll empID \gg}_{\mathcal{I}}$. *It is easy to see that* $\ll E\_id \gg \cup \text{PAYMENTID}_{\leq 1}$ *includes* $\ll empID \gg$. *Therefore altogether we obtain that* $\ll empID \gg$ *is relatively identified by* $\ll IRD\# \gg$.

In fact, every non-trivial XFD in Σ(PAYMENTID) *each* $r_{\text{PAYMENTID}}$*-walk on the right-hand-side is relatively identified by the left-hand-side. Therefore we conclude that* PAYMENTID *is in X^{i}NF with respect to* Σ(PAYMENTID). *This confirms the observation in Example 5.2 that* PAYMENTID *is non-redundant with respect to* Σ(PAYMENTID). □

Example 5.5. *We can also verify that* BANKID *is not in X^{i}NF with respect to* Σ(BANKID), *i.e. supporting that* BANKID *is indeed redundant with respect to* Σ(BANKID) *by Theorem 5.3.*

For a simple example, consider the non-trivial XFD

$$(B\_XFD6) \quad \ll bankName \gg \rightarrow \ll bankNo \gg$$

From the sequence of restricted pre-closures, we obtain

$$\ll bankName \gg_{\mathcal{I}}^{\ll bankNo \gg} = \ll bankName \quad bankNo \gg$$

There is no identifier at all, hence $\ll bankNo \gg$ *is not relatively identified by* $\ll bankName \gg$. *So* B_XFD6 *is witness that* BANKID *is not in* X^iNF *with respect to* Σ(BANKID).

As another example, consider the XFD

$$(B\_XFD12) \quad \ll bankName \quad acctNo \gg \rightarrow \ll brName \quad acctName \gg$$

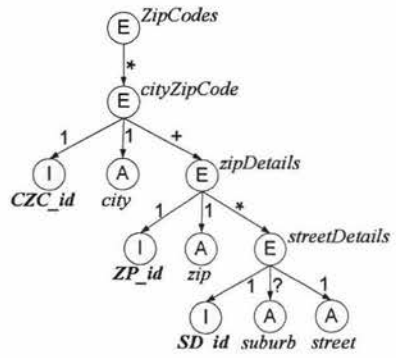
For convenience, we simply use X *to refer to* $\ll bankName \quad acctNo \gg$. *It is easy to verify that* $X \rightarrow \ll brName \gg$ *is non-trivial. The sequence of restricted pre-closures for* X *relative to* $\ll brName \gg$ *is as follows:*

$$\begin{array}{lcll} X_0 & = & X & \\ X_1 & = & X \cup \ll brName \quad acctName \gg & by B\_XFD12 \\ X_2 & = & X \cup \ll brName \quad acctName \quad branchNo \gg = X_3 = X_4 = \ldots & by B\_XFD10 \end{array}$$

Again there is no identifier in X_n *(where* $n = 2$*) therefore* $\ll brName \gg$ *is not relatively identified by* $\ll bankName \quad acctNo \gg$ *and* B_XFD12 *shows that* BANKID *is not in* X^iNF *with respect to* Σ(BANKID). □

5.5 "Redundancy" as a Design Quality

In our definition of redundancy, we have considered XFDs as representing a unit of information. In accordance with the same notion in the RDM, we have considered a piece of information to be redundant if it is inferrable from other information stored in an XML data tree which satisfies the XFD representing the unit of information. However, this means we are only required to ensure that data corresponding to the right-hand-side of each dependency is not stored redundantly. The replication of data corresponding to the left-hand-side is ignored.



Σ(ZipA) consists of:

(ZA_XFD1)	≪street zip≫	→	≪city≫
(ZA_XFD2)	≪street suburb≫	→	≪zip≫
(ZA_XFD3)	≪city≫	→	≪CZC_id≫
(ZA_XFD4)	≪city zip≫	→	≪ZP_id≫
(ZA_XFD5)	≪street suburb zip≫	→	≪SD_id≫

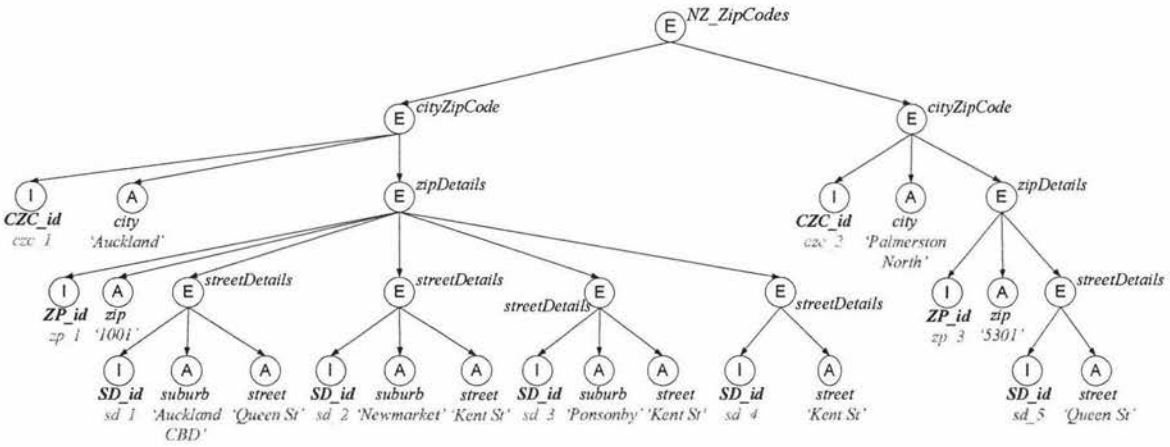


Figure 29: One XML schema tree ZipA for the zip code example, together with a set Σ(ZipA) of XFDs on ZipA and a compatible XML data tree.

Example 5.6. *Let us consider a simplified zip code system. Suppose we need to store data about the zip code of each location in New Zealand where the location details include the name of the city, suburb and street. It happens that the street together with the zip code determines the city in which the street is located and the street together with the suburb determines the zip code. We can model this in various ways. Consider the two approaches presented in Figure 29 and Figure 30.*

We can verify that ZipA *is in* $X^{\mathrm{i}}NF$ *with respect to* Σ(ZipA) *and* ZipB *is in* $X^{\mathrm{i}}NF$ *with respect to* Σ(ZipB). *According to Theorem 5.3, this means that both schema trees are considered to be non-redundant with respect to their corresponding set of XFDs.*

Observe that in each of the given XML data tree, there are three almost copies of the compatible schema tree with valuation of "Kent St" and "1001" for ≪*street zip*≫ . *In each case, the three almost copies coincide on* ≪*city*≫ *but do not coincide on* ≪*street zip*≫ . □

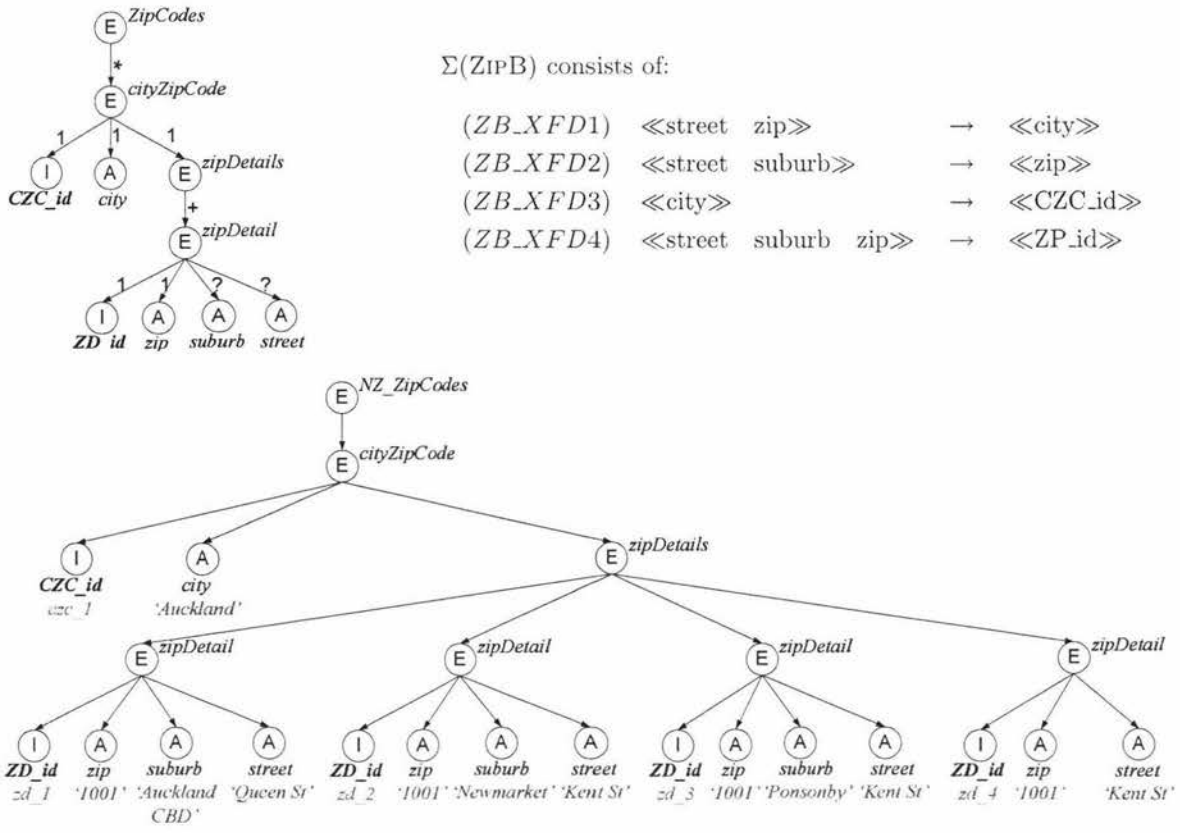


Figure 30: Another XML schema tree ZIPB for the zip code example, together with a set Σ(ZIPB) of XFDs on ZIPB and a compatible XML data tree.

The above example suggests that the generalisation of non-redundancy in Definition 5.1 may be too weak. We can alternatively consider the following definition of redundancy:

> Let T be an XML schema graph and Σ a set of XFDs on T. We say that T *is redundant with respect to* Σ if and only if there is some XML data tree $T' \rhd T$ such that $\models_{T'} \Sigma$ and for some non-trivial XFD $X \to Y \in \Sigma$ with r_T-walk $B \in Y - X$ there exist two almost copies T'_1, T'_2 of T in T' with $T'_1|_{XB} = T'_2|_{XB} \cong XB$ but T'_1 and T'_2 do not coincide on any r_T-walk of XB.

However, there is one simple argument against using such a definition: if the information about the left-hand-side of an XFD is lost or missing, there is no way to derive this information from the rest/remaining information.

The main question that needs to be asked as a result of this discussion is how redundancy should be defined in the context of XML and whether non-redundancy should be an important quality aspect for the design of XML schemas.

6 Related Work

6.1 Relational Databases with Null Values

We refer the reader to [19] for a general introduction to relations with null values.

Work on functional dependencies in relational databases with null values dates back to as early as 1979, carried out by Lien [20]. This work was continued in [21]. Lien provides an axiomatisation for functional dependencies and an axiomatisation for multivalued dependencies in relational databases with null values. It is one of the earliest papers to remark that the transitivity rule is not sound in the presence of missing information.

The axiomatisation of Lien can also be found in [6] where it is generalised by Athena and Morfuni for functional dependencies in the presence of null values (NFDs) together with existence constraints (ECs) over flat relations. An existence constraint is an expression of the form $e : X \vdash Y$ which is satisfied by a relation if each tuple t which is X-total (i.e. $t[X]$ does not contain any null values) is also Y-total. Like frequencies, ECs offer a means to control the presence of missing values. In particular, ECs are required for defining a restricted form of the transitivity rule which is sound in the presence of null values. ECs may express some, but not all, frequencies and conversely frequencies may express some, but not all, ECs.

Consequently the discussion of NFDs with ECs in [6] has some similarities with the discussion in Section 3. The restricted-transitivity rule and restricted-pseudo-transitivity rule resembles the transitivity rule with ECs [6, pg.15] and the J_3 rule [6, pg.19] respectively. The sound and complete Athena-Morfuni rule system for NFDs and ECs (see Theorem 11 in [6, pg.20]) has some similarity with the $\mathcal{F}$-rule system.

However the Athena-Morfuni rule system also contains additional inference rules for the derivation of ECs which are not required for frequencies. More importantly, the $\mathcal{F}$-rule system contains two additional inference rules to the Athena-Morfuni rule system: the root axiom, and the noname rule. The root axiom exists due to the cardinality constraint of "at most once" specified by frequencies, while the noname rule arises from the nesting structure of data. There is no comparable feature in flat relations with nulls; this may explain the non-availability of rules similar to the root axiom and noname rule in the Athena-Morfuni rule system.

6.2 XML and Semistructured Databases

In this section, we identify more specifically related work focussing on functional dependencies in the context of XML or semistructured databases.

One of the first papers to address the issue of designing "good" XML schemas is [10] by Embley and Mok. The main purpose of the paper is the translation of arbitrary conceptual-model hypergraphs, used to express sets of integrity constraints, into scheme-tree forests.[2] No notion of functional dependency is defined; instead the concept of functional constraints expressible by conceptual-model hypergraphs is considered. The paper includes a definition of a normal form called XNF. A scheme-tree forest is said to be in XNF if it does not contain "potential redundancy" and is as compact as possible in terms of the number of scheme trees. The notion of potential redundancy with respect to a functional constraint $X \rightarrow Y$ checks whether each value assigned to an attribute in Y occurs exactly once in each scheme-tree instance, particularly we note that the definition disregards the values for X. One significant limitation of this work by Embley and Mok is that it is difficult to verify syntactically whether a given XML schema can be considered as a "good" design.

In [12], a notion of path constraints was briefly mentioned. However, to the best of our knowledge, the first work to define a notion of functional dependencies for XML is Lee et al [16]. In this paper, functional dependencies for XML (called FD_{XML}) are defined over paths rooted at any element in an XML document. This is facilitated by including a "header path" within each FD_{XML} expression. The remaining part of an FD_{XML} is a functional constraint of the form $P_{x_1}, \ldots, P_{x_n} \rightarrow P_y$ where each P_i must be a path to an element. Attributes can only be optionally included if they are key for the element to which they are attached. Lee et al also attempt to characterise "well-structured FD_{XML}". One condition for an FD_{XML} being well-structured is that the header path, followed by each path in the left-hand-side of the FD_{XML} followed by the single path on the right-hand-side must form a path. This condition is based on earlier works on functional dependencies in semistructured databases including [17, 38]. A limitation of FD_{XML} being well-structured is that we are not able to express functional dependencies between subelements of different elements or functional dependencies where subelements determine an element in which they are nested. An example provided in [16] shows that for "flat XML data", it is not meaningful to have well-structured FD_{XML}. In particular, well-structured FD_{XML} is not a generalisation of the notion of functional dependencies in the RDM. Also well-structured FD_{XML} does not recognise some functional dependencies expressible over nested relations.

[2] Embley and Mok later discuss how DTDs can be generated from scheme-tree forests in [11]

Overall, the discussions in [16] are quite informal.

One of the best-known research works on functional dependencies for XML is [2], a short version of the journal paper [4], by Arenas and Libkin. The focus of the paper is on how to design DTDs in order to avoid two "commonly present" problems in poor XML schema design, namely redundancy and update anomalies. Arenas and Libkin define the notions of functional dependencies in XML (which we will refer to as XML FDs) and their satisfaction using the concept of "tree tuple", which is similar to the total unnesting of a nested relation. A normal form for XML (called XNF) is also introduced which is later shown to guarantee the absence of redundancy using an information theoretic approach [3]. In [2, 4] it is illustrated and formally shown that XNF is a generalisation of BCNF and the nested normal form NNF introduced by Mok et al in [26].

Among all the literature, XML FDs based on "tree tuples" are the most similar to subgraph-based XFDs from [14, 15]. In particular, the concept of "tree tuple" closely corresponds to the concept of almost copies of an XML schema graph, and both generalise a "no information" interpretation of null values from the RDM [5, Chapt. 6] for defining satisfaction of functional dependencies in XML. We refer the reader to [5] for definitions of strong and weak satisfaction of functional dependencies in relations with null values and various different interpretations of null values. The "no information" interpretation of null values is common for functional dependencies on relations with null values, for example in [6, 18, 20, 21].

One difference between the "tree tuple" approach and the subgraph-based approach is that the former requires the existence of DTDs while the latter does not. With DTDs, disjunction is permitted, which is not considered in the XML graph model. It should be noted that, in contrast to XFDs, XML FDs may include paths from the unique root vertex to any vertices, including internal vertices. One motivation for not considering such paths to internal vertices is that internal vertices do not have counterparts in the RDM. Instead identifier-attributes may be used to refer to particular vertices. This is more in line with original intentions of XML and provides more flexibility: identifier-attributes are specified if desired, but not compulsory. Overall, we expect that X^iNF proposed in this thesis implies Arenas and Libkin's XNF if identifier-attributes are compulsory.

No axiomatisation of XML FDs was given by Arenas and Libkin. However, some complexity results were given in [2, 4] about the implication problem for XML functional dependencies over various classes of DTDs. One important result is that XML functional dependencies cannot be finitely axiomatised. The finite axiomatisation of XFDs is one of the major results of this thesis. The provided proof in [4] suggests that arbitrary disjunction contributes to the XML functional dependencies not being finitely axiomatisable.

Recall that we do not consider disjunction. Our resolution of disjunction considers XML schemas which are similar to "simple" DTDs whereby every disjunctive expression can be rewritten as a simple regular expression containing no disjunction. The implication problem for functional dependencies over simple DTDs is shown to be solvable in quadratic time.

The normalisation strategy of Arenas and Libkin assumes a certain degree of non-missing information in order to ensure losslessness; any path which participates in an XML FD is assumed to be non-missing in the corresponding XML data tree. In [4], it is suggested that we can relax this assumption by converting a schema permitting missing information into a collection of subschemas identifying every possible data structure which may occur due to missing information. The XML FD then only applies to the subschema which is the same as the original schema. We remark that this approach may lead to exponential growth in the size of an XML schema which is hardly desirable. Another problem with the normalisation approach of Arenas and Libkin is that it is unlikely to result in an ideal data structure as the normalisation process usually generates very flat structures.

In the last two years, Vincent et al is another group in the research community who have investigated how to adapt functional dependencies for XML. Recent works include: [22, 28, 31, 33, 34, 35, 36]. The main focus of research has been directed towards the study of functional dependencies in XML which are strongly satisfied (called strong XFDs). A normal form is given which guarantees the absence of redundancy and also an axiomatisation of strong XFDs. The normal form with respect to strong XFDs (called XNF) proposed in these papers is a slight modification of the one introduced in [2, 4]. This alteration is required because of the mixed-content elements permitted by Vincent et al but ignored by Arenas and Libkin.

Similar to [2, 4], functional dependencies are defined in terms of paths to arbitrary vertices including internal vertices. But unlike Arenas and Libkin, no notion of frequencies is considered. A major distinguishing feature of works by Vincent et al is the concept of strong satisfaction. Similar to relations with null values, Vincent et al propose that we think of an XML tree with missing information as representing a collection of "complete" XML trees with non-missing information. An XML tree T then strongly satisfies an XFD φ if every complete XML trees represented by T satisfy φ. Expressed in the framework of this thesis, an XML data tree containing at least one almost copy which is not a copy of the given XML schema graph is said to contain missing information. In particular, this implies that the "unknown" interpretation (values exist but not currently known) is used for information deemed to be missing. We remark that the "no information" interpretation is more general than the "unknown" interpretation as it also includes the

"does not exist" interpretation (values do not exist). Due to the flexibility of XML data structure, we argue that it is more natural to use the "no information" or "does not exist" interpretation of missing information.

Although the inference rules investigated by Vincent et al are shown to be sound for strong XFDs, they are only proven to be complete for strong XFDs with a single path on the left-hand-side (called unary XFDs). In contrast to the inference rules in Section 3 and Section 4, the inference rules proposed by Vincent et al are not easily compared to inference rules for functional dependencies in the RDM. We also highlight that the transitivity rule is sound for strong XFDs but not sound for XFDs discussed in this thesis.

In [22], Vincent et al have also studied primary XFDs (only paths to leaves) which is similar to subgraph-based XFDs without identifiers and frequencies. In [22], the focus is placed on the issue of designing XML documents. An analogy of transitive dependencies and partial dependencies is generalised from the RDM for primary XFDs. It is suggested that the presence of transitive and partial dependencies leads to redundancy and therefore should be eliminated/avoided. The paper illustrated this with an example but provides no formal justification. Vincent et al have also introduced a new notion of functional dependencies called "local XFDs" [23], and give a possible generalisation of the notion of multivalued dependencies and defined a normal form with respect to these multivalued dependencies in [29, 30, 32].

Finally there is the subgraph-based approach to functional dependencies for XML [14, 15] on which we have based the work of this thesis. [15] includes a sound and complete rule system for XFDs on XML schema graphs with no frequencies and no identifiers and also suggests what inference rules should constitute an axiomatisation of XFDs with frequencies and XFDs with identifiers. In contrast to this thesis, leaves in [14, 15] may be of kind E. An important contribution of [14, 15] is to provide a framework in which various notions of functional dependencies in XML can be defined and studied.

7 Conclusion

In this thesis, we have studied a subgraph-based approach toward functional dependencies for XML, which we have referred to as XFDs. We have utilised the definition of XFDs for the XML graph model introduced in [14, 15].

The first main result of the research is an axiomatisation for XFDs in the presence of frequencies and also an axiomatisation for XFDs in the presence of frequencies and identifiers. In the presence of frequencies, we have presented a sound and complete rule system, denoted the $\mathcal{F}$-rule system, which includes the reflexivity axiom, subtree rule, supertree rule, union rule, restricted-transitivity rule and noname rule. In the presence of identifiers, we have generalised the noname rule and added a new noname2 axiom to obtain a sound and complete rule system denoted as the $\mathcal{I}$-rule system.

In discussing a completeness proof for the $\mathcal{F}$-rule system, we have observed that there may exist XFDs which are not implied in general but may be satisfied due to the non-trivial interaction between the XML schema and the XFDs. This has necessitated the consideration of a sequence of pre-closures in addition to a simple analogy of an attribute closure in the RDM.

We have also discussed alternative sound and complete rule systems for implication of XFDs in the presence of frequencies by showing that certain sets of inference rules are equivalent. The notion of Armstrong relation has also been generalised for XML data trees and with a simple proof we have shown that XFDs in the presence of frequencies do not enjoy Armstrong XML data trees.

For the second part of our investigation, we have defined a notion of redundancy in the context of XML, and have devised a normal form called X^{i}NF which guarantees non-redundancy. In the definition of redundancy, we have considered an XFD to be representing a unit of information. An XML schema graph is then said to contain redundancy with respect to some XFD $X \rightarrow Y$ if it contains two distinguishable non-empty almost copies of Y, with the copies of X attached to each almost copy of Y being equivalent. X^{i}NF makes use of identifiers to guarantee that this cannot occur. We have also discussed a more elegant approach to verifying whether an XML schema graph is in X^{i}NF. Similar to the RDM, it is sufficient to examine XFDs in a given set of XFDs, rather than all implications of the given set of XFDs.

7.1 Future Work

There are many other areas of research which still need to be investigated to proceed towards developing a theory for the design of XML schemas.

The implication problem in the context of XML relates to the ability to decide whether an XFD $X \rightarrow Y$ is implied by a given set Σ of XFDs. In our framework, due to Theorem 3.5 and Theorem 4.3, this is equivalent to deciding whether $Y \in X^+_{\mathcal{R}}$ where $\mathcal{R}$ is either the $\mathcal{F}$-rule system or the $\mathcal{I}$-rule system. Using the inference rules to compute the set of all derivable XFDs is quite impractical. Therefore, it is necessary to develop a simple algorithm for generating $X^+_{\mathcal{R}}$ for an arbitrary schema graph T, an arbitrary set Σ of XFDs and an arbitrary r_T-subgraph X in T. The study of the related work suggests that an initial algorithm may be derivable from algorithms in [6] since many inference rules in this thesis can be easily mapped to rules in [6].

In this thesis, we have proposed a normal form in the presence of frequencies and identifiers. In particular, redundancy is prevented by the uniqueness property of identifiers. Another direction of research is to investigate a normal form for XML schemas in the absence of identifiers. This may involve generalising the notion of multivalued dependencies to the context of XML and examining existing normal forms for nested relations.

In Section 5.5 we have suggested that one important area still to be addressed is the formal characterisation of properties of "bad" XML schema design. Up to now, it has been assumed that, similar to the RDM, redundancy and update anomalies are two important quality aspects of XML schemas. However, apart from [33] there has been no formal definition of any update anomalies in the context of XML, and little to no formal definition of redundancy can be found in the literature. It is vital in the study of how to design "good" XML schema that we are able to identify "good" schemas. It is important to investigate whether redundancy and update anomalies carry the same importance for XML as they did for the RDM, and also to identify other potentially desirable properties for XML data which may not exist for the RDM.

In this thesis, we have not considered how to transform an XML schema which violates $\mathrm{X^iNF}$ into one that does not. We have left this for future research. It is likely that the normalisation approach introduced by Arenas and Libkin [2, 4] can be translated to the framework in this thesis. Another area for future research is the study of XFDs in the presence of references. References may be used to ensure losslessness during normalisation, for minimising redundancy when designing XML schemas and for increasing the expressiveness of XFDs.

With respect to functional dependencies, it is also possible to investigate other notions of functional dependencies, such as XFDs based on the notion of pre-image and XFDs containing disjunctions. As with the RDM, different classes of dependencies interact with one another. Currently no research has been directed towards studying the interaction among different notions of dependencies. Another possible research direction is to study XFDs under different assumptions. For example, also to consider disjunction and recursion in XML schemas, permit mixed-content data and consider document order of XML elements.

References

[1] ARENAS, M., FAN, W., AND LIBKIN, L. What's hard about XML schema constraints? In *Proceedings of the 13th International Conference on Database and Expert Systems Applications* (2002), vol. 2453 of *Lecture Notes in Computer Science (LNCS)*, Springer-Verlag, pp. 269–278.

[2] ARENAS, M., AND LIBKIN, L. A normal form for XML documents. In *Proceedings of the twenty-first ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems* (2002), ACM Press, pp. 85–96.

[3] ARENAS, M., AND LIBKIN, L. An information-theoretic approach to normal forms for relational and XML data. In *Proceedings of the twenty-second ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems* (2003), ACM Press, pp. 15–26.

[4] ARENAS, M., AND LIBKIN, L. A normal form for XML documents. *ACM Trans. Database Syst. 29*, 1 (2004), 195–232.

[5] ATZENI, P., AND ANTONELLIS, V. D. *Relational database theory.* Benjamin-Cummings Publishing Co., Inc., 1993.

[6] ATZENI, P., AND MORFUNI, N. M. Functional dependencies and constraints on null values in database relations. *Information and Control 70*, 1 (July 1986), 1–31.

[7] BUNEMAN, P., DAVIDSON, S., FAN, W., HARA, C., AND TAN, W.-C. Keys for XML. In *Proceedings of the tenth international conference on World Wide Web* (2001), ACM Press, pp. 201–210.

[8] BUNEMAN, P., DAVIDSON, S. B., FAN, W., HARA, C. S., AND TAN, W. C. Reasoning about keys for XML. In *Revised Papers from the 8th International Workshop on Database Programming Languages* (2002), Springer-Verlag, pp. 133–148.

[9] EDWARDS, R., AND HOPE, S. Persistent DOM: An architecture for XML repositories in relational databases. In *IDEAL* (2000), K.-S. Leung, L.-W. Chan, and H. Meng, Eds., vol. 1983 of *Lecture Notes in Computer Science (LNCS)*, Springer, pp. 416–421.

[10] EMBLEY, D. W., AND MOK, W. Y. Developing XML documents with guaranteed "good" properties. In *Proceedings of the 20th International Conference on Conceptual Modeling (ER2001)* (November 2001), H. S. Hunii, S. Jajodia, and A. Slvberg, Eds.,

vol. 2224 of *Lecture Notes in Computer Science (LNCS)*, Springer-Verlag, pp. 426–441.

[11] EMBLEY, D. W., AND MOK, W. Y. Producing XML documents with guaranteed "Good" properties. In *Proceedings of the 7th World Multiconference on Systemics, Cybernetics and Informatics (SCI2003)* (2003), N. Callaos, W. Lesso, S. Rahimi, V. Boonjiing, J. Mohamad, T.-K. Liu, and K.-D. Schewe, Eds., vol. IX of *Computer Science and Engineering: II*, International Institute of Informatics and Systemics (IIIS). Invited paper.

[12] FAN, W., AND SIMÉON, J. Integrity constraints for XML. In *Proceedings of the nineteenth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems (PODS2000* (2000), ACM Press, pp. 23–34.

[13] FLORESCU, D., AND KOSSMANN, D. Storing and querying XML data using an RDMBS. *IEEE Data Engineering Bulletin 22*, 3 (1999), 27–34.

[14] HARTMANN, S., AND LINK, S. More functional dependencies for XML. In *ADBIS* (2003), L. A. Kalinichenko, R. Manthey, B. Thalheim, and U. Wloka, Eds., vol. 2798 of *Lecture Notes in Computer Science*, Springer, pp. 355–369.

[15] HARTMANN, S., LINK, S., AND KIRCHBERG, M. A subgraph-based approach towards functional dependencies for XML. In *Proceedings of the 7th World Multiconference on Systemics, Cybernetics and Informatics (SCI2003)* (2003), N. Callaos, W. Lesso, S. Rahimi, V. Boonjiing, J. Mohamad, T.-K. Liu, and K.-D. Schewe, Eds., vol. IX of *Computer Science and Engineering: II*, International Institute of Informatics and Systemics (IIIS), pp. 200–211.

[16] LEE, M.-L., LING, T. W., AND LOW, W. L. Designing functional dependencies for XML. In *Advances in Database Technology - EDBT2002: Proceedings of the VIII International Conference on Extending Database Technology* (March 2002), vol. 2287 of *Lecture Notes in Computer Science (LNCS)*, Springer-Verlag, pp. 124–141.

[17] LEE, S. Y., LEE, M.-L., LING, T. W., AND KALINICHENKO, L. A. Designing good semi-structured databases. In *18th International Conference on Conceptual Modeling (ER'99)* (November 1999), International Conference on Conceptual Modeling, pp. 131–145.

[18] LEVENE, M., AND LOIZOU, G. Semantics for null extended nested relations. *ACM Transactions on Database Systems 18*, 3 (1993), 414–459.

[19] LEVENE, M., AND LOIZOU, G. *A Guided Tour of Relational Databases and Beyond.* Springer-Verlag, 1999.

[20] LIEN, Y. E. Multivalued dependencies with null values in relational data bases. In *Fifth International Conference on Very Large Data Bases* (October 1979), Very Large Data Bases, pp. 61–66.

[21] LIEN, Y. E. On the equivalence of database models. *Journal of the ACM (JACM) 29*, 2 (1982), 333–362.

[22] LIU, J., VINCENT, M., AND LIU, C. Functional dependencies, from relational to XML. In *Fifth International Conference Perspectives of System Informatics* (Novosibirsk, 2003). submitted to DASFAA.

[23] LIU, J., VINCENT, M., AND LIU, C. Local XML functional dependencies. In *Accepted for publication, Fifth ACM International Workshop on Web Information and Data Management* (New Orleans, USA, 2003), L. Ee-Peng, R. H. L. Chiang, and A. H. F. Laender, Eds., WIDM 2003, ACM Press, pp. 23–28.

[24] LU, S., SUN, Y., ATAY, M., AND FOTOUHI, F. A new inlining algorithm for mapping XML DTDs to relational schemas. In *ER (Workshops)* (2003), M. A. Jeusfeld and O. Pastor, Eds., vol. 2814 of *Lecture Notes in Computer Science*, Springer, pp. 366–377.

[25] MEN-HIN, Y., AND FU, A. W.-C. From XML to relational databases. In *Knowledge Representation Meets Databases* (2001).

[26] MOK, W. Y., NG, Y.-K., AND EMBLEY, D. W. A normal form for precisely characterizing redundancy in nested relations. *ACM Trans. Database Syst. 21*, 1 (1996), 77–106.

[27] SHANMUGASUNDARAM, J., TUFTE, K., ZHANG, C., HE, G., DEWITT, D. J., AND NAUGHTON, J. F. Relational databases for querying XML documents: Limitations and opportunities. In *VLDB* (1999), M. P. Atkinson, M. E. Orlowska, P. Valduriez, S. B. Zdonik, and M. L. Brodie, Eds., Morgan Kaufmann, pp. 302–314.

[28] VINCENT, M., AND LIU, J. Functional dependencies for XML. In *Web Technologies and Applications - The Fifth Asia Pacific Web Conference (APWEB 2003)* (April 23-25 2003), M. O. X. Zhou, Y. Zhang, Ed., vol. 2642 of *Lecture Notes in Computer Science (LNCS)*, Springer, pp. 22–34.

[29] VINCENT, M., AND LIU, J. Multivalued dependencies and a 4NF for XML. In *The 15th International Conference on Advanced Information Systems Engineering*

(CAISE) (Klagenfurt/Velden, Austria, 2003), M. M. J. Eder, Ed., vol. 2681 of *Lecture Notes in Computer Science (LNCS)*, Springer, pp. 14–29. Submitted to Journal of the ACM.

[30] VINCENT, M., AND LIU, J. Multivalued dependencies in XML. In *Twentieth British National Conference on Databases (BNCOD)* (2003), vol. 2712 of *Lecture Notes in Computer Science (LNCS)*, Springer, pp. 4–18.

[31] VINCENT, M., AND LIU, J. Strong functional dependencies and a redundancy free normal form for XML. In *7th World Multi-Conference on Systemics, Cybernetics and Informatics* (2003).

[32] VINCENT, M., LIU, J., AND LIU, C. A redundancy free 4NF for XML. In *First International XML Database Symposium, XSym 2003* (Berlin, Germany, 2003), R. Unland, A. B. Chaudhri, Z. Bellahsene, E. Rahm, and M. Rys, Eds., Springer-Verlag, pp. 254–266.

[33] VINCENT, M., LIU, J., AND LIU, C. Redundancy free mappings from relations to XML. In *Fourth International Conference on Web-Age Information Management (WAIM 2003)* (2003).

[34] VINCENT, M. W., AND LIU, J. Completeness and decidability properties for functional dependencies in XML. *CoRR cs.DB/0301017* (January 2003), 1017–+.

[35] VINCENT, M. W., LIU, J., AND LIU, C. Strong functional dependencies and a redundancy free normal form for XML. Technical report, Advanced Computing Research Centre, School of Computer and Information Science, The University of South Australia, Adelaide, Australia, 2003.

[36] VINCENT, M. W., LIU, J., AND LIU, C. Strong functional dependencies and their application to normal forms in XML. *ACM Trans. Database Syst. 29*, 3 (2004), 445–462.

[37] WIDOM, J. Data management for XML: Research directions. *IEEE Data Engineering Bulletin 22*, 3 (1999), 44–52.

[38] WU, X., LING, T. W., LEE, S. Y., AND LEE, M. L. NF-SS: A normal form for semistructured schema. In *Proceedings of the 20th International Conference on Conceptual Modeling (ER2001)* (November 2001), H. S. Hunii, S. Jajodia, and A. Slvberg, Eds., vol. 2224 of *Lecture Notes in Computer Science (LNCS)*, Springer-Verlag. In International Workshop on Data Semantics in Web Information Systems — DASWIS 2001.

[39] YOSHIKAWA, M., AND AMAGASA, T. XRel: a path-based approach to storage and retrieval of XML documents using relational databases. *ACM Trans. Inter. Tech. 1*, 1 (2001), 110–141.