A Redundancy Free 4NF for XML

Millist W. Vincent, Jixue Liu, and Chengfei Liu

School of Computer and Information Science
University of South Australia
{millist.vincent, jixue.liu, chengfei.liu}@unisa.edu.au

Abstract. While providing syntactic flexibility, XML provides little semantic content and so the study of integrity constraints in XML plays an important role in helping to improve the semantic expressiveness of XML. Functional dependencies (FDs) and multivalued dependencies (MVDs) play a fundamental role in relational databases where they provide semantics for the data and at the same time are the foundation for database design. In some previous work, we defined the notion of multivalued dependencies in XML (called XMVDs) and defined a normal form for a restricted class of XMVDs, called hierarchical XMVDs. In this paper we generalise this previous work and define a normal form for arbitrary XMVDs. We then justify our definition by proving that it guarantees the elimination of redundancy in XML documents.

1 Introduction

XML has recently emerged as a standard for data representation and interchange on the Internet [18, 1]. While providing syntactic flexibility, XML provides little semantic content and as a result several papers have addressed the topic of how to improve the semantic expressiveness of XML. Among the most important of these approaches has been that of defining integrity constraints in XML [3]. Several different classes of integrity constraints for XML have been defined including key constraints [3, 4], path constraints [6], and inclusion constraints [7] and properties such as axiomatization and satisfiability have been investigated for these constraints. However, one topic that has been identified as an open problem in XML research [18] and which has been little investigated is how to extended the traditional integrity constraints in relational databases, namely *functional dependencies* (FDs) and *multivalued dependencies* (MVDs), to XML and then how to develop a normalisation theory for XML. This problem is not of just theoretical interest. The theory of normalisation forms the cornerstone of practical relational database design and the development of a similar theory for XML will similarly lay the foundation for understanding how to design XML documents. In addition, the study of FDs and MVDs in XML is important because of the close connection between XML and relational databases. With current technology, the source of XML data is typically a relational database [1] and relational databases are also normally used to store XML data [9]. Hence, given that FDs and MVDs are the most important constraints in relational databases, the study

of these constraints in XML assumes heightened importance over other types of constraints which are unique to XML [5].

In this paper we extend some previous work [16, 15] and consider the problem of defining multivalued dependencies and normal forms in XML documents. Multivalued dependencies in XML (called XMVDs) were first defined in [16]. In that paper we extended the approach used in [13, 14] to define functional dependendencies and defined XMVDs in XML documents. We then formally justified our definition by proving that, for a very general class of mappings from relations to XML, a relation satisfies a multivalued dependency (MVD) if and only if the corresponding XML document satisfies the corresponding XMVD. The class of mappings considered was those defined by converting a flat relation to a nested relation by an arbitrary sequences of nest operators, and then mapping the nested relation to an XML document in the obvious manner. Thus our definition of a XMVD in an XML document is a natural extension of the definition of a MVD in relations. In [15] the issue of defining normal forms in the presence of XMVDs was addressed. In that paper we defined a normal form for a restricted class of XMVDs, namely what we termed *hierarchical XMVDs*. Also, extending some of our previous work on formally defining redundancy in flat relations ([11, 12, 8]) and in XML ([13]), we formally defined redundancy in [15] and showed that the normal form that we defined guaranteed the elimination of redundancy in the presence of XMVDs.

The main contribution of this paper is to extend the results obtained in [15]. As just mentioned, in [15] we considered only a restricted class of XMVDs called hierarchical XMVDs. Essentially, an XMVD is hierarchical if the paths on the r.h.s. of an XMVD are descendants of the path on the l.h.s. of the XMVD. In this paper we define a normal form for arbitrary XMVDs, i.e. no retriction is placed on the relationships between the paths in the XMVD. We then formally justify our definition by proving that it guarantees the elimination of redundancy.

The rest of this paper is organised as follows. Section 2 contains some preliminary definitions. Section 3 contains the definition of an XMVD. In Section 4 we define a 4NF for XML and prove that it eliminates redundancy. Finally, Section 5 contains some concluding comments.

2 Preliminary Definitions

In this section we present some preliminary definitions that we need before defining XFDs. We model an XML document as a tree as follows.

Definition 1. *Assume a countably infinite set* $\mathbf{E}$ *of element labels (tags), a countable infinite set* $\mathbf{A}$ *of attribute names and a symbol* $\mathtt{S}$ *indicating text. An* XML tree *is defined to be* $T = (V, lab, ele, att, val, v_r)$ *where* V *is a* finite *set of nodes in* T*;* lab *is a function from* V *to* $\mathbf{E} \cup \mathbf{A} \cup \{\mathtt{S}\}$*;* ele *is a partial function from* V *to a sequence of* V *nodes such that for any* $v \in V$*, if* $ele(v)$ *is defined then* $lab(v) \in \mathbf{E}$*;* att *is a partial function from* $V \times \mathbf{A}$ *to* V *such that for any* $v \in V$ *and* $l \in \mathbf{A}$*, if* $att(v, l) = v_1$ *then* $lab(v) \in \mathbf{E}$ *and* $lab(v_1) = l$*;* val *is a*

function such that for any node in $v \in V, val(v) = v$ *if* $lab(v) \in \mathbf{E}$ *and* $val(v)$ *is a string if either* $lab(v) = \mathcal{S}$ *or* $lab(v) \in \mathbf{A}$*;* v_r *is a distinguished node in* V *called the root of* T *and we define* $lab(v_r) = root$*. Since node identifiers are unique, a consequence of the definition of val is that if* $v_1 \in \mathbf{E}$ *and* $v_2 \in \mathbf{E}$ *and* $v_1 \neq v_2$ *then* $val(v_1) \neq val(v_2)$*. We also extend the definition of val to sets of nodes and if* $V_1 \subseteq V$*, then* $val(V_1)$ *is the set defined by* $val(V_1) = \{val(v) | v \in V_1\}$*.*

For any $v \in V$*, if* $ele(v)$ *is defined then the nodes in* $ele(v)$ *are called* subelements *of* v*. For any* $l \in \mathbf{A}$*, if* $att(v, l) = v_1$ *then* v_1 *is called an* attribute *of* v*. Note that an XML tree* T *must be a tree. Since* T *is a tree the set of ancestors of a node* v*, is denoted by* $Ancestor(v)$*. The children of a node* v *are also defined as in Definition 1 and we denote the parent of a node* v *by* $Parent(v)$*.*

We note that our definition of *val* differs slightly from that in [4] since we have extended the definition of the *val* function so that it is also defined on element nodes. The reason for this is that we want to include in our definition paths that do not end at leaf nodes, and when we do this we want to compare element nodes by node identity, i.e. node equality, but when we compare attribute or text nodes we want to compare them by their contents, i.e. value equality. This point will become clearer in the examples and definitions that follow.

We now give some preliminary definitions related to paths.

Definition 2. *A* path *is an expression of the form* $l_1.\cdots.l_n$*,* $n \geq 1$*, where* $l_i \in \mathbf{E} \cup \mathbf{A} \cup \{\mathcal{S}\}$ *for all* $i, 1 \leq i \leq n$ *and* $l_1 = root$*. If* p *is the path* $l_1.\cdots.l_n$ *then* $Last(p) = l_n$*.*

For instance, if $\mathbf{E}$ = {`root, Division, Employee`} and $\mathbf{A}$ = {`D#, Emp#`} then `root, root.Division, root.Division.D#,`
`root.Division.Employee.Emp#.S` are all paths.

Definition 3. *Let* p *denote the path* $l_1.\cdots.l_n$*. The function* $Parnt(p)$ *is the path* $l_1.\cdots.l_{n-1}$*. Let* p *denote the path* $l_1.\cdots.l_n$ *and let* q *denote the path* $q_1.\cdots.q_m$*. The path* p *is said to be a* prefix *of the path* q*, denoted by* $p \subseteq q$*, if* $n \leq m$ *and* $l_1 = q_1, \ldots, l_n = q_n$*. Two paths* p *and* q *are equal, denoted by* $p = q$*, if* p *is a prefix of* q *and* q *is a prefix of* p*. The path* p *is said to be a* strict prefix *of* q*, denoted by* $p \subset q$*, if* p *is a prefix of* q *and* $p \neq q$*. We also define the intersection of two paths* p_1 *and* p_2*, denoted but* $p_1 \cap p_2$*, to be the maximal common prefix of both paths. It is clear that the intersection of two paths is also a path.*

For example, if $\mathbf{E}$ = {`root, Division, Employee`} and $\mathbf{A}$ = {`D#, Emp#`} then `root.Division` is a strict prefix of `root.Division.Employee` and
`root.Division.D#` $\cap$ `root.Division.Employee.Emp#.S` = `root.Division`.

Definition 4. *A* path instance *in an XML tree* T *is a sequence* $v_1.\cdots.v_n$ *such that* $v_1 = v_r$ *and for all* $v_i, 1 < i \leq n$*,*$v_i \in V$ *and* v_i *is a child of* v_{i-1}*. A path instance* $v_1.\cdots.v_n$ *is said to be* defined over the path $l_1.\cdots.l_n$ *if for all* $v_i, 1 \leq i \leq n$*,* $lab(v_i) = l_i$*. Two path instances* $v_1.\cdots.v_n$ *and* $v'_1.\cdots.v'_n$ *are said to be* distinct *if* $v_i \neq v'_i$ *for some* i*,* $1 \leq i \leq n$*. The path instance* $v_1.\cdots.v_n$ *is said to be a* prefix *of* $v'_1.\cdots.v'_m$ *if* $n \leq m$ *and* $v_i = v'_i$ *for all* $i, 1 \leq i \leq n$*. The*

path instance $v_1.\cdots.v_n$ *is said to be a* strict prefix *of* $v'_1.\cdots.v'_m$ *if* $n < m$ *and* $v_i = v'_i$ *for all* $i, 1 \leq i \leq n$. *The set of path instances over a path* p *in a tree* T *is denoted by* $Paths(p)$

For example, in Figure 1, $v_r.v_1.v_3$ is a path instance defined over the path `root.Dept.Section` and $v_r.v_1.v_3$ is a strict prefix of $v_r.v_1.v_3.v_4$

We now assume the existence of a set of legal paths P for an XML application. Essentially, P defines the semantics of an XML application in the same way that a set of relational schema define the semantics of a relational application. P may be derived from the DTD, if one exists, or P be derived from some other source which understands the semantics of the application if no DTD exists. The advantage of assuming the existence of a set of paths, rather than a DTD, is that it allows for a greater degree of generality since having an XML tree conforming to a set of paths is much less restrictive than having it conform to a DTD. Firstly we place the following restriction on the set of paths.

Definition 5. *A set* P *of paths is* consistent *if for any path* $p \in P$, *if* $p_1 \subset p$ *then* $p_1 \in P$.

This is natural restriction on the set of paths and any set of paths that is generated from a DTD will be consistent.

We now define the notion of an XML tree conforming to a set of paths P.

Definition 6. *Let* P *be a consistent set of paths and let* T *be an XML tree. Then* T *is said to* conform *to* P *if every path instance in* T *is a path instance over some path in* P.

The next issue that arises in developing the machinery to define XFDs is the issue is that of missing information. This is addressed in [13] but in this we take the simplifying assumption that there is no missing information in XML trees. More formally, we have the following definition.

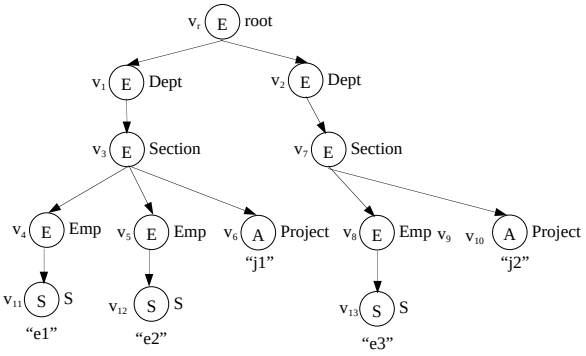

Fig. 1. A complete XML tree.

Definition 7. *Let P be a consistent set of paths, let T be an XML that conforms to P. Then T is defined to be* complete *if whenever there exist paths p_1 and p_2 in P such that $p_1 \subset p_2$ and there exists a path instance $v_1.\cdots.v_n$ defined over p_1, in T, then there exists a path instance $v'_1.\cdots.v'_m$ defined over p_2 in T such that $v_1.\cdots.v_n$ is a prefix of the instance $v'_1.\cdots.v'_m$.*

For example, if we take P to be {`root, root.Dept, root.Dept.Section, root.Dept.Section.Emp, root.Dept.Section.Emp.S, root.Dept.Section.Project`} then the tree in Figure 1 conforms to P and is complete.

The next function returns all the final nodes of the path instances of a path p in T.

Definition 8. *Let P be a consistent set of paths, let T be an XML tree that conforms to P . The function $N(p)$, where $p \in P$, is the set of nodes defined by $N(p) = \{v | v_1.\cdots.v_n \in Paths(p) \wedge v = v_n\}$.*

For example, in Figure 1, $N($`root.Dept`$) = \{v_1, v_2\}$.

We now need to define a function that returns a node and its ancestors.

Definition 9. *Let P be a consistent set of paths, let T be an XML tree that conforms to P. The function $AAncestor(v)$, where $v \in V \cup \mathbf{N}$, is the set of nodes in T defined by $AAncestor(v) = v \cup Ancestor(v)$.*

For example in Figure 1, $AAncestor(v_3) = \{v_r, v_1, v_3\}$. The next function returns all nodes that are the final nodes of path instances of p and are descendants of v.

Definition 10. *Let P be a consistent set of paths, let T be an XML tree that conforms to P. The function $Nodes(v, p)$, where $v \in V \cup \mathbf{N}$ and $p \in P$, is the set of nodes in T defined by $Nodes(v, p) = \{x | x \in N(p) \wedge v \in AAncestor(x)\}$*

For example, in Figure 1 , $Nodes(v_1,$`root.Dept.Section.Emp`$) = \{v_4, v_5\}$. We also define a partial ordering on the set of nodes as follows.

Definition 11. *The partial ordering $>$ on the set of nodes V in an XML tree T is defined by $v_1 > v_2$ iff $v_2 \in Ancestor(v_1)$.*

3 XMVDs in XML

Before presenting the main definition of the paper, we present an example to illustrate the thinking behind the definition. Consider the relation shown in Figure 2. It satisfies the MVD `Course` $\rightarrow\!\!\rightarrow$ `Teacher | Text`. The XML tree shown in Figure 3 is then a XML representation of the data in Figure 2. The tree has the following property. There exists two path instances of `root.Id.Id.Id.Text`, namely $v_r.v_{13}.v_{17}.v_{21}.v_9$ and $v_r.v_{16}.v_{20}.v_{24}.v_{12}$ such that $val(v_9) \neq val(v_{12})$. Also, these two paths have the property that for the closest `Teacher` node to v_9, namely v_5, and the closest `Teacher` node to v_{12}, namely v_8, then $val(v_5) \neq val(v_8)$ and for the closest `Course` node to both v_9 and v_5, namely v_1, and for the closest

`Course` node to both v_{12} and v_8, namely v_4, we have that $val(v_1) = val(v_4)$. Then the existence of the two path instances $v_r.v_{13}.v_{17}.v_{21}.v_9$ and $v_r.v_{16}.v_{20}.v_{24}.v_{12}$ with these properties and the fact that `Course` $\rightarrow\rightarrow$ `Teacher|Text` is satisfied in the relation in Figure 2 implies that there exists two path instances of `root.Id.Id.Id.Text`, namely $v_r.v_{15}.v_{19}.v_{23}.v_{11}$ and $v_r.v_{14}.v_{18}.v_{22}.v_{10}$, with the following properties. $val(v_{11}) = val(v_9)$ and for the closest `Teacher` node to v_{11}, v_7, $val(v_7) = val(v_8)$ and for the closest `Course` node to v_{11} and v_7, namely v_3, $val(v_3) = val(v_1)$. Also, $val(v_{10}) = val(v_{12})$ and the closest `Teacher` node to v_{10}, v_6, $val(v_6) = val(v_5)$ and for the closest `Course` node to v_{10} and v_6, namely v_2, $val(v_2) = val(v_4)$. This type of constraint is an XMVD. We note however that there are many other ways that the relation in Figure 2 could be represented in an XML tree. For instance we could also represent the relation by Figure 4 and this XML tree also satisfies the XMVD. In comparing the two representations, it is clear that the representation in Figure 4 is a more compact representation than that in Figure 3 and we shall see later that the example in Figure 4 is normalised whereas the example in Figure 3 is not.

Course	Teacher	Text
Algorithms	Fred	Text A
Algorithms	Mary	Text B
Algorithms	Fred	Text B
Algorithms	Mary	Text A

Fig. 2. A flat relation satisfying a MVD.

This leads us to the main definition of our paper. In this paper we consider the simplest case where there are only single paths on the l.h.s. and r.h.s. of the XMVD and all paths end in an attribute or text node.

Definition 12. *Let P be a consistent set of paths and let T be an XML tree that conforms to P and is complete. An XMVD is a statement of the form $p \rightarrow\rightarrow q|r$ where p, q and r are paths in P. T satisfies $p \rightarrow\rightarrow q|r$ if whenever there exists two distinct paths path instances $v_1.\cdots.v_n$ and $w_1.\cdots.w_n$ in $Paths(q)$ such that:*

(i) $val(v_n) \neq val(w_n)$;

(ii) there exists two nodes z_1, z_2, where $z_1 \in Nodes(x_{1_1}, r)$ and $z_2 \in Nodes(y_{1_1}, r)$ such that $val(z_1) \neq val(z_2)$;

(iii) there exists two nodes z_3 and z_4, where $z_3 \in Nodes(x_{1_{1_1}}, p)$ and $z_4 \in Nodes(y_{1_{1_1}}, p)$, such that $val(z_3) = val(z_4)$;

then:

(a) there exists a path $v'_1.\cdots.v'_n$ in $Paths(q)$ such that $val(v'_n) = val(v_n)$ and there exists a node z'_1 in $Nodes(x'_{1_1}, r)$ such that $val(z'_1) = val(z_2)$ and there exists a node z'_3 in $Nodes(x'_{1_{1_1}}, p)$ such that $val(z'_3) = val(z_3)$;

(b) there exists a path $w'_1.\cdots.w'_n$ in $Paths(q)$ such that $val(w'_n) = val(w_n)$ and there exists a node z'_2 in $Nodes(x'_{1_1}, r)$ such that $val(z'_2) = val(z_1)$ and there exists a node z'_4 in $Nodes(x'_{1_{1_1}}, p_l)$ such that $val(z'_4) = val(z_4)$;

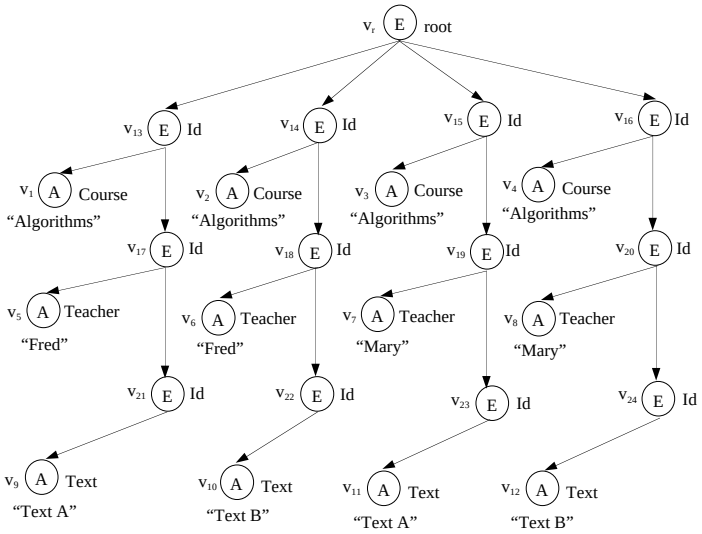

Fig. 3. An XML tree

where $x_{1_1} = \{v|v \in \{v_1, \cdots, v_n\} \wedge v \in N(r \cap q)\}$ *and* $y_{1_1} = \{v|v \in \{w_1, \cdots, w_n\} \wedge v \in N(r \cap q)\}$ *and* $x_{1_{1_1}} = \{v|v \in \{v_1, \cdots, v_n\} \wedge v \in N(p \cap r \cap q)\}$ *and* $y_{1_{1_1}} = \{v|v \in \{w_1, \cdots, w_n\} \wedge v \in N(p \cap r \cap q)\}$

and $x'_{1_1} = \{v|v \in \{v'_1, \cdots, v'_n\} \wedge v \in N(r \cap q)\}$ *and* $y'_{1_1} = \{v|v \in \{w'_1, \cdots, v'_n\} \wedge v \in N(r \cap q)\}$ *and* $x'_{1_{1_1}} = \{v|v \in \{v'_1, \cdots, v'_n\} \wedge v \in N(p \cap r \cap q)\}$ *and* $y'_{1_{1_1}} = \{v|v \in \{w'_1, \cdots, w'_n\} \wedge v \in N(p \cap r \cap q)\}$.

We note that since the path $r \cap q$ is a prefix of q, there exists only one node in $v_1.\cdots.v_n$ that is also in $N(r \cap q)$ and so x_1 is always defined and is a single node. Similarly for $y_1, x_{1_{1_1}}, y_{1_{1_1}}, x'_{1_1}, y'_{1_1}, x'_{1_{1_1}}, y'_{1_{1_1}}$. We now illustrate the definition by some examples.

Example 1. Consider the XML tree shown in Figure 4 and the XMVD `root.Id.Course` $\rightarrow\rightarrow$ `root.Id.Id.Teacher|root.Id.Id.Text`. Let $v_1.\cdots.v_n$ be the path instance $v_r.v_8.v_2.v_4$ and let $w_1.\cdots.w_n$ be the path instance $v_r.v_8.v_2.v_5$. Both path instances are in $Paths($`root.Id.Id.Teacher`$)$ and $val(v_4) \neq val(v_5)$. Moreover, $x_{1_1} = v_8$, $y_{1_1} = v_8$, $x_{1_{1_1}} = v_8$ and $y_{1_{1_1}} = v_8$. So if we let $z_1 = v_6$ and $z_2 = v_7$ then $z_1 \in Nodes(x_{1_1},$ `root.Id.Id.Text`$)$ and

$z_2 \in Nodes(y_{1_1},$ `root.Id.Id.Text`$)$. Also if we let $z_3 = v_1$ and $z_4 = v_1$ then $z_3 \in Nodes(x_{1_{1_1}},$ `root.Id.Course`$)$ and $z_4 \in Nodes(y_{1_{1_1}},$ `root.Id.Course`$)$ then $val(z_3) = val(z_4)$. Hence conditions (i), (ii) and (iii) of the definition of an XMVD are satisfied.

If we let $v'^i_1.\cdots.v'^i_n$ be the path $v_r.v_8.v_2.v_4$ we firstly have that $val(v'^i_n) = val(v^i_n)$ as required. Also, since the path instances are the same we have that $x_{1_1} = x'_{1_1}$ and $x_{1_{1_1}} = x'_{1_{1_1}}$. So if we let $z'_1 = v_7$ then

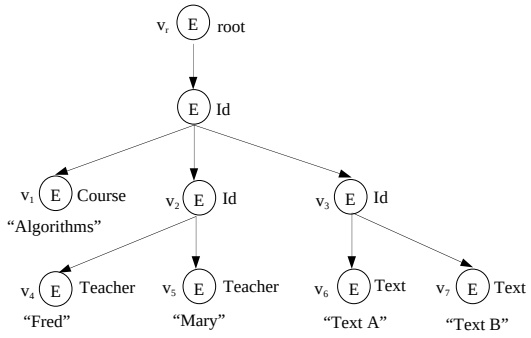

Fig. 4. An XML tree

$z'_1 \in Nodes(x'_{1_1}, \texttt{root.Id.Id.Text})$ and $val(z'_1) = val(z_2)$ and if we let $z'_3 = v_1$ then $z'_3 \in Nodes(x'_{1_{1_l}}, \texttt{root.Id.Course})$ and $val(z'_3) = val(z_3)$. So part (a) of the definition of an XMVD is satisfied. Next if we let $w'^i_1.\cdots.w'^i_n$ be the path $v_r.v_8.v_2.v_5$ then we firstly have that $val(w'^i_n) = val(w^i_n)$ since the paths are the same . Also, since the paths are the same we have that $y_{1_1} = y'_{1_1}$ and $y_{1_{1_1}} = y'_{1_{1_1}}$. So if we let $z'_2 = v_6$ then $z'_2 \in Nodes(y'_{1_1}, \texttt{root.Id.Id.Text})$ and $val(z'_2) = val(z_1)$ and if we let $z'_4 = v_1$ then $z'_4 \in Nodes(x'_{1_{1_l}}, \texttt{root.Id.Course})$ and $val(z'_4) = val(z_4)$. Hence part (b) on the definition of an XMVD is satisfied and so T satisfies $\texttt{root.Id.Course} \rightarrow\rightarrow \texttt{root.Id.Id.Teacher} | \texttt{root.Id.Id.Text}$.

As explained earlier, the tree in Figure 4 also satisfies

$\texttt{root.Id.Course} \rightarrow\rightarrow \texttt{root.Id.Id.Teacher} | \texttt{root.Id.Id.Text}$.

Example 2. Consider the XML tree shown in Figure 5 and the XMVD

$\texttt{root.Project.P\#} \rightarrow\rightarrow \texttt{Root.Project.Person.Name} | \texttt{root.Project.Part.Pid}$.

For the path instances $v_r.v_1.v_5.v_{13}$ and $v_r.v_2.v_8.v_{16}$ in

$Paths(\texttt{Root.Project.Person.Name})$ we have that $val(v_{13}) \neq val(v_{16})$. Moreover, $x_{1_1} = v_1$, $y_{1_1} = v_2$, $x_{1_{1_1}} = v_1$ and $y_{1_{1_1}} = v_2$. So if we let $z_1 = v_{15}$ and $z_2 = v_{18}$ then $z_1 \in Nodes(x_{1_1}, \texttt{root.Project.Part.Pid})$ and $z_2 \in Nodes(y_{1_1},$

$\texttt{root.Project.Part.Pid})$. Also if we let $z_3 = v_4$ and $z_4 = v_7$ then $z_3 \in Nodes(x_{1_{1_1}}, \texttt{root.Project.P\#})$ and $z_4 \in Nodes(y_{1_{1_1}}, \texttt{root.Project.P\#}\ )$ and $val(z_3) = val(z_4)$. Hence conditions (i), (ii) and (iii) of the definition of an XMVD are satisfied. However, for the only other path in

$Paths(\texttt{Root.Project.Person.Name})$, namely $v_r.v_3.v_{11}.v_{19}$ we have that $x'_{1_1} = v_3$ and so $Nodes(x'_{1_1}, \texttt{root.Project.part.Pid}) = v_{21}$ and since $val(v_{21}) \neq val(z_2)$ and so it does not satisfy condition (a) and thus

$\texttt{root.Project.P\#} \rightarrow\rightarrow \texttt{Root.Project.Person.Name} | \texttt{root.Project.part.Pid}$

is violated in T.

Consider then the XMVD $\texttt{root.Project.Person.Name}$

$\rightarrow\rightarrow \texttt{Root.Project.Person.Skill} | \texttt{root.Project.P\#}$ in the same XML tree. For the path instances $v_r.v_1.v_5.v_{14}$ and $v_r.v_3.v_{11}.v_{20}$ in $Paths(\texttt{Root.Project.}$

`Person.Skill`) we have that $val(v_{14}) \neq val(v_{20})$. Moreover, $x_{1_1} = v_1$, $y_{1_1} = v_3$, $x_{1_{1_1}} = v_1$ and $y_{1_{1_1}} = v_3$. So if we let $z_1 = v_4$ and $z_2 = v_{10}$ then $z_1 \in Nodes(x_{1_1}, \texttt{root.Project.P\#})$ and $z_2 \in Nodes(y_{1_1}, \texttt{root.Project.P\#})$. Also if we let $z_3 = v_{13}$ and $z_4 = v_{19}$ then $z_3 \in Nodes(x_{1_{1_1}}, \texttt{root.Project.Person.Name})$ and $z_4 \in Nodes(y_{1_{1_1}}, \texttt{root.Project.Person.Name})$ and $val(z_3) = val(z_4)$. Hence the conditions of (i), (ii) and (iii) of the definition of an XMVD are satisfied. However there does not exist another path instance in $Paths(\texttt{Root.Project.Person.Skill})$ such that *val* of the last node in the path is equal to *val* of node v_{14} and so part (a) of the definition of an XMVD is violated.

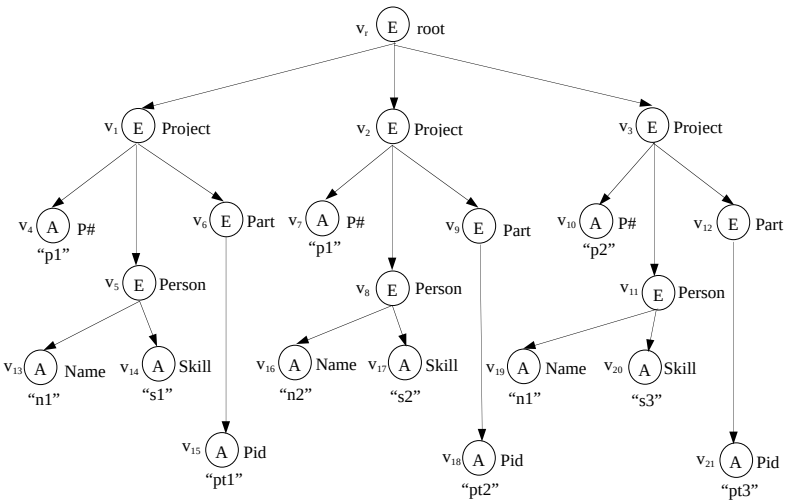

Fig. 5. An XML tree

4 A redundancy Free 4NF for XML documents

In this section we propose a 4NF for XML documents. We also provide a formal justification for the normal form by showing that it ensures the elimination of redundancy in the presence of XMVDs. This approach to justifying the definition of a normal form is an extension of the approach adopted by one of the authors in some other research which investigated the issue of providing justification for the normal forms defined in standard relational databases [10–12, 8].

The approach that we use to justifying our normal form is to formalise the notion of redundancy, the most intuitive approach to justifying normal forms, and then to try to ensure that our normal form ensures there is no redundancy. However, defining redundancy is not quite so straightforward as might first appear. The most obvious approach is, given a relation r and a FD $A \rightarrow B$ and

two tuples t_1 and t_2, to define a value $t_1[B]$ to be redundant if $t_1[B] = t_2[B]$ and $t_1[A] = t_2[A]$. While this definition is fine for FDs in relations, it doesn't generalise in an obvious way to other classes of relational integrity constraints, such as *multi-valued dependencies* (MVDs) or *join dependencies* (JDs) or *inclusion dependencies* (INDs), nor to other data models. The key to finding the appropriate generalisation is based on the observation that if a value $t_1[B]$ is redundant in the sense just defined then every change of $t_1[B]$ to a new value results in the violation of $A \rightarrow B$. One can then define a data value to be redundant if every change of it to a new value results in the violation of the set of constraints (whatever they may be). This is essentially the definition proposed in [12] where it was shown that BCNF is a necessary and sufficient condition for there to be no redundancy in the case of FD constraints and fourth normal form (4NF) is a necessary and sufficient condition for there to be no redundancy in the case of FD and MVD constraints.

The definition we propose is the following which is an extension of the definition given in [12].

Definition 13. *Let T be an XML tree and let v be a node in T. Then the change from v to v', resulting in a new tree T', is said to be a* valid change *if $v \neq v'$ and $val(v) \neq val(v')$.*

We note that the second condition in the definition, $val(v) \neq val(v')$, is automatically satisfied if the first condition is satisfied when $lab(v) \in \mathbf{E}$.

Definition 14. *Let P be a consistent set of paths and let Σ be a set of XMVDs such that every path appearing in an XMVD in Σ is in P. Then Σ is said to* cause redundancy *if there exists a complete XML tree T which conforms to P and satisfies Σ and a node v in T such that* every *valid change from v to v', resulting in a new XML tree T', causes Σ to be violated.*

The essential idea is that if a value is redundant, then it is implied by the other data values and the set of constraints and so any change to the value causes a violation of the constraints. For example, consider Figure 3 and the set Σ of XMVDs

$\{$`root.Id.Course` $\rightarrow\!\!\rightarrow$ `root.Id.Id.Teacher|root.Id.Id.Id.Text`$\}$. Then Σ causes redundancy because the tree shown in Figure 3 satisfies Σ yet every valid change to any of the `Text` nodes (or `Teacher` nodes) results in the violation of Σ.

Next, we define the notion of a key.

Definition 15. *Let P be a consistent set of paths, let T be an XML tree that conforms to P and is complete and let $p \in P$. Then T satisfies the key constraint p if whenever there exists two nodes v_1 and v_2 in $N(p)$ in T such that $val(v_1) = val(v_2)$ then $v_1 = v_2$.*

We note that since node indentifiers in XML trees are unique, it follows that if $Last(p) \in \mathbf{E}$ then p is automatically a key. Next, we define a normal form for XML.

Definition 16. *Let Σ be a set XMVDs and key constraints. Then Σ is in* XML fourth normal form *(4XNF) if for every XMVD $p \rightarrow\rightarrow q|r \in \Sigma$, at least one of the following conditions holds:*

(A) q and r are both keys;

(B) p is a key and $q \cap r = p$;

(C) p is a key and $q \cap r$ is a strict prefix of p;

(D) $q \cap r = root$;

(E) there exists an XMVD $s \rightarrow\rightarrow t|u \in \Sigma$ such that $s \cap p = root$ and $t \geq q \cap r$ and t is a key and $u \geq q \cap r$ and u is a key;

(F) there exists an XMVD $a \rightarrow\rightarrow b|c \in \Sigma$ such that $a \cap p = root$ and b is not a key and c is a key and $b \cap c \cap p \neq root$ and there exists $b \rightarrow\rightarrow d|e$ such that d is a key and e is a key and $d \geq q \cap r$ and $e \geq q \cap r$;

(G) q is not a key and r is not a key and there exists $p \rightarrow\rightarrow q|k$ and there exists $p \rightarrow\rightarrow k|r$ such that $k \geq q \cap r$;

(H) p is a key, q is a key and r is not a key and $q \cap r \neq p$ and $q \cap r$ is not a strict prefix of p and there exists $x \rightarrow\rightarrow q|k$ such that $x < p$ and $k > p \cap q \cap r$ and $k \cap q \cap r \neq p \cap q \cap r$;

(I) p is a key, q is not a key and r is a key and $q \cap r \neq p$ and $q \cap r$ is not a strict prefix of p and $\exists x \rightarrow\rightarrow k|r$ such that $x < p$ and $k > p \cap q \cap r$ and $k \cap q \cap r \neq p \cap q \cap r$.

We now illustrate the definition by an example.

Example 3. Consider the tree T in Figure 3 and assume that the only constraint is the XMVD

`root.Id.Course` $\rightarrow\rightarrow$ `root.Id.Id.Teacher|root.Id.Id.Id.Text`. T satisfies Σ and is complete. However Σ is not in 4XNF since `root.Id.Course` is not a key and `root.Id.Id.Teacher` is not a key and `root.Id.Id.Id.Text` is not a key. Consider then the tree shown in Figure 4 and assume that the only XMVD is `root.Id.Course` $\rightarrow\rightarrow$ `root.Id.Teacher|root.Id.Text`. If `root.Id.Course` is a key, which would be the case if `Course` was specified as type `ID` in the full DTD, then Σ is in 4XNF since `root.Id.Course` is a key and `root.Id.Teacher` $\cap$ `root.Id.Id.Id.Text` = `root.Id` and so (C) of the condition for 4XNF is satisfied since `root.Id` is a strict prefix of `root.Id.Course`.

This leads to the main result of the paper.

Theorem 1. *Let Σ be a set of XMVDS and key constraints. If Σ is in 4XNF then it does not cause redundancy.*

Proof. See Appendix.

5 Conclusions

In this paper we have investigated the issues of XMVDs and 4NF in XML. We proposed a normal form for XML documents in the presence of XMVDs

and justified it by showing that it ensures the elimination of redundancy. This extended the results in [15] which defined a normal form for a restricted class of XMVDs called hierarchical XMVDs.

There are several other issues related to the ones addressed in this paper that warrant further investigation. The first is the need to generalise the main result of this paper. We need to show that 4XNF we proposed is also a necessary condition for the elimination of redundancy. Secondly, we need to investigate the problem of developing an axiom system for reasoning about the implication of XMVDs. In [13] an axiom system for reasoning about the implication of XFDs was developed and the system was shown to be sound for arbitrary XFDs. Later [17], the implication problem for XFDs was shown to be decidable and the axiom system presented in [13] was shown to be complete for unary XFDs and a polynomial time algorithm was developed for determining if a unary XFD is implied by a set of unary XFDs. Similarly, we need to develop an axiom system and algorithm for the implication problem for XMVDs. Thirdly, we need to develop algorithms for converting unnormalised XML documents to normalised ones. In the relational case, the equivalent procedure is performed using a decomposition algorithm based on the projection operator. However, at the moment there has been no commonly agreed upon algebra defined for XML, let alone a projection operator, so the development of procedures for normalising XML documents is likely to be more complex than in the relational case. Fourthly, it is necessary to consider the case where both XFDs and XMVDs exist in a document. It is interesting to note that unlike the situation for the relational case, 4XNF is not a straightforward generalisation of the normal form for XFDs (XNF). This means that, in contrast to the relational case where 4NF implies BCNF in the case where both MVDs and FDs are present, in XML a different normal form from 4XNF is needed when the constraints on an XML document contain both XFDs and XMVDs. The situation is further complicated by the fact that XMVDs and XFDs interact, in the sense that there are XMVDs and XFDs implied by a combined set of XMVDs and XFDs which are not implied by either the XMVDs or XFDs considered alone. This situation parallels that of relational databases [2].

References

1. S. Abiteboul, P. Buneman, and D. Suciu. *Data on the Web.* Morgan Kauffman, 2000.
2. S. Abiteboul, R. Hull, and V. Vianu. *Foundations of databases.* Addison WAesley, 1996.
3. P. Buneman, S. Davidson, W. Fan, and C. Hara. Reasoning about keys for xml. In *International Workshop on Database Programming Languages*, 2001.
4. P. Buneman, S. Davidson, W. Fan, C. Hara, and W. Tan. Keys for xml. *Computer Networks*, 39(5):473–487, 2002.
5. P. Buneman, W. Fan, J. Simeon, and S. Weinstein. Constraints for semistructured data and xml. *ACM SIGMOD Record*, 30(1):45–47, 2001.
6. P. Buneman, W. Fan, and S. Weinstein. Path constraints on structured and semistructured data. In *Proc. ACM PODS Conference*, pages 129 – 138, 1998.

7. W. Fan and J. Simeon. Integrity constraints for xml. In *Proc. ACM PODS Conference*, pages 23–34, 2000.

8. M. Levene and M. W. Vincent. Justification for inclusion dependency normal form. *IEEE Transactions on Knowledge and Data Engineering*, 12:281 –291, 2000.

9. J. Shanmugasundaram, K. Tufte, C. Zhang, G. He, D. J. DeWitt, and J. F. Naughton:. Relational databases for querying xml documents: Limitations and opportunities. In *VLDB Conference*, pages 302 – 314, 1999.

10. M. W. Vincent. A corrected 5nf definition for relational database design. *Theoretical Computer Science*, 185:379 –391, 1997.

11. M. W. Vincent. A new redundancy free normal form for relational database design. In B. Thalheim and L. Libkin, editors, *Database Semantics*, pages 247 –264. Springer Verlag, 1998.

12. M. W. Vincent. Semantic foundations of 4nf in relational database design. *Acta Informatica*, 36:1 –41, 1999.

13. M.W. Vincent and J. Liu. Strong functional dependencies and a redundancy free normal form for xml. Submitted to ACM Transactions on Database Systems, 2002.

14. M.W. Vincent and J. Liu. Functional dependencies for xml. In *Fifth Asian Pacific Web Conference*, 2003.

15. M.W. Vincent and J. Liu. Multivalued dependencies and a 4nf for xml. In *15th International Conference on Advanced Information Systems Engineering (CAISE)*, 2003.

16. M.W. Vincent and J. Liu. Multivalued dependencies in xml. In *20th British National Conference on Databases (BNCOD)*, 2003.

17. M.W. Vincent, J. Liu, and C. Liu. The implication problem for unary functional dependencies in xml. In *submitted for publication*, 2003.

18. J. Widom. Data management for xml - research directions. *IEEE data Engineering Bulletin*, 22(3):44–52, 1999.

6 Appendix

This section is to help the reviewer and will be removed in the final version of the paper

We start with a preliminary lemma.

Lemma 1. *Let P be a consistent set of paths and let Σ be a set of XMVDs such that every path appearing in an XMVD in Σ is in P. Let T be a complete XML tree T which conforms to P and satisfies Σ and let v be a node in T such that* every *valid change from v to v', resulting in a new XML tree T', causes Σ to be violated. Then there exists an XMVD $p \rightarrow\rightarrow q|r$ such that $v \in N(q)$ and there exists path instances $v_1.\cdots.v_n$ (where $v_n = v'$) and $w_1.\cdots.w_n$ in $Paths(q)$ in T and $val(v_n) = val(w_n)$ and there exist two nodes z_1, z_2, where $z_1 \in Nodes(x_{1_1}, r)$ and $z_2 \in Nodes(y_{1_1}, r)$ where $x_{1_1} = \{v|v \in \{v_1, \cdots, v_n\} \wedge v \in N(r \cap q)\}$ and $y_{1_1} = \{v|v \in \{w_1, \cdots, v_n\} \wedge v \in N(r \cap q)\}$, such that $val(z_1) \neq val(z_2)$.*

Proof.

We firstly claim that $v \in N(q)$ or $v \in N(r)$. Suppose that this is not the case and that $v \notin N(q)$ and $v \notin N(r)$ and $v \notin N(p)$. Then obviously T' satisfies Σ and so Σ does not cause redundancy which is a contradiction. Suppose then that v in $N(p)$. If $lab(v) \in \mathbf{E}$ then by definition the new node v' is distinct in T' and so by definition of an XMVD T' satisfies Σ and so Σ does not cause redundancy which is a contradiction. If instead $lab(v) \in \mathbf{A}$ then if we choose a change such that $val(v')$ does not appear anywhere else in T, the XMVD $p \rightarrow\rightarrow q|r$ is not violated after the change and so T' satisfies Σ and so Σ does not cause redundancy which is a contradiction. Hence $v \in N(q)$ or $v \in N(r)$.

Next, consider the claim that there exists an XMVD $p \rightarrow\rightarrow q|r$ such that $v \in N(q)$ and there exists at least two path instances $v_1.\cdots.v_n$ and $w_1.\cdots.w_n$ in $Paths(q)$ in T such that $v_n = v$ and $val(v_n) = val(w_n)$. Suppose that this is not the case. If there is only one path instance in $Paths(q)$ in T, then there will be only one path instance in $Paths(q)$ in T' and so by (i) of a definition of an XMVD $p \rightarrow\rightarrow q|r$ will be satisfied in T' which is a contradiction. Hence we conclude that there is more than one instance in $Paths(q)$ in T. Suppose then that all the the path instances in $Paths(q)$ in T have different $val's$. Let $v_1.\cdots.v_n$ be the path instance in $Paths(q)$ ending in v, i.e. $v =$, and let $w_1.\cdots.w_n$ be an arbitrary path in $Paths(q)$. Let where $x_{1_1} = \{v|v \in \{v_1, \cdots, v_n\} \wedge v \in N(r \cap q)\}$ and $y_{1_1} = \{v|v \in \{w_1, \cdots, v_n\} \wedge v \in N(r \cap q)\}$. Then we claim that there exists a path $w_1.\cdots.w_n$ such that $Nodes(y_{1_1}, r)$ contains a node whose val is distinct from a node in $Nodes(y_{1_1}, r)$. If this is not the case, then since only the val of v is changed, all the nodes in $N(r)$ in T' will have the same val and so condition (ii) of the definition of an XMVD is not satisfied and so the fact that $p \rightarrow\rightarrow q|r$ is violated in T' will be contradicted. So conditions (i), (ii) and (iii) of the definition of an XMVD are satisfied and so by (a) of the definition of an XMVD there exists a path $v'_1.\cdots.v'_n$ in $Paths(q)$ such that $val(v'_n) = val(v_n)$. Now $v'_1.\cdots.v'_n$ must be distinct from $v_1.\cdots.v_n$ or else the fact that $p \rightarrow\rightarrow q|r$ is violated in T' will be contradicted. Hence there are two distinct path instances

in $Paths(q)$ in T which end in nodes having different $val's$ and so the result is proven.

□

Proof of Theorem 1

Assume that (A) holds, i.e. q and r are both keys, and suppose to the contrary that Σ causes redundancy. Then by definition there exists an XML tree T which satisfies Σ and a node v in T such that *every* valid change from v to v', resulting in a new XML tree T', causes some XMVD $p \rightarrow\rightarrow q|r \in \Sigma$ to be violated. We firstly claim that $v \in N(q)$ or $v \in N(r)$. Suppose that this is not the case and that $v \notin N(q)$ and $v \notin N(r)$ and $v \notin N(p)$. Then obviously T' satisfies Σ and so Σ does not cause redundancy which is a contradiction. Suppose then that v in $N(p)$. If $lab(v) \in \mathbf{E}$ then by definition the new node v' is distinct in T' and so by definition of an XMVD T' satisfies Σ and so Σ does not cause redundancy which is a contradiction. If instead $lab(v) \in \mathbf{A}$ then if we choose a change such that $val(v')$ does not appear anywhere else in T, the XMVD $p \rightarrow\rightarrow q|r$ is not violated after the change and so T' satisfies Σ and so Σ does not cause redundancy which is a contradiction. Hence $v \in N(q)$ or $v \in N(r)$. We suppose firstly that $v \in N(q)$.

So since $p \rightarrow\rightarrow q|r$ in T' is violated, there exist path instances $v_1.\cdots.v_n$ (where $v_n = v'$) and $w_1.\cdots.w_n$ in $Paths(q)$ in T' such that:

(i) $val(v_n) \neq val(w_n)$;

(ii) there exists two nodes z_1, z_2, where $z_1 \in Nodes(x_{1_1}, r)$ and $z_2 \in Nodes(y_{1_1}, r)$ such that $val(z_1) \neq val(z_2)$;

(iii) there exists two nodes z_3 and z_4, where $z_3 \in Nodes(x_{1_{1_1}}, p)$ and $z_4 \in Nodes(y_{1_{1_1}}, p)$, such that $val(z_3) = val(z_4)$.

and:

(a.1) there does not exist a path $v'_1.\cdots.v'_n$ in $Paths(q)$ such that $val(v'_n) = val(v_n)$ or there does not exist a node z'_1 in $Nodes(x'_{1_1}, r)$ such that $val(z'_1) = val(z_2)$ or there does not exist a node z'_3 in $Nodes(x'_{1_{1_1}}, p)$ such that $val(z'_3) = val(z_3)$;

or

(b.1) there does not exists a path $w'_1.\cdots.w'_n$ in $Paths(q)$ such that $val(w'_n) = val(w_n)$ or there does not exist a node z'_2 in $Nodes(x'_{1_1}, r)$ such that $val(z'_2) = val(z_1)$ or there does not exist a node z'_4 in $Nodes(x'_{1_{1_1}}, p_l)$ such that $val(z'_4) = val(z_4)$;

where

$x_{1_1} = \{v|v \in \{v_1, \cdots, v_n\} \wedge v \in N(r \cap q)\}$ and $y_{1_1} = \{v|v \in \{w_1, \cdots, v_n\} \wedge v \in N(r \cap q)\}$ and $x_{1_{1_1}} = \{v|v \in \{v_1, \cdots, v_n\} \wedge v \in N(p \cap r \cap q)\}$ and $y_{1_{1_1}} = \{v|v \in \{w_1, \cdots, w_n\} \wedge v \in N(p \cap r \cap q)\}$

and $x'_{1_1} = \{v|v \in \{v'_1, \cdots, v'_n\} \wedge v \in N(r \cap q)\}$ and $y'_{1_1} = \{v|v \in \{w'_1, \cdots, v'_n\} \wedge v \in N(r \cap q)\}$ and $x'_{1_{1_1}} = \{v|v \in \{v'_1, \cdots, v'_n\} \wedge v \in N(p \cap r \cap q)\}$ and $y'_{1_{1_1}} = \{v|v \in \{w'_1, \cdots, w'_n\} \wedge v \in N(p \cap r \cap q)\}$.

Next, because $v_1.\cdots.v_n$ and $w_1.\cdots.w_n$ satisfy (ii), it follows that since only node v_n is changed in T, then $z_1, z_2, z_3, z_4, x_{1_1}, x'_{1_1}, y_{1_1}, y'_{1_1}, x_{1_{1_1}}, x'_{1_{1_1}}, y_{1_{1_1}}$ and

$y'_{1_{1_1}}$ are the same in T and T'. So by (ii) there exist two nodes z_1, z_2, where $z_1 \in Nodes(x_{1_1}, r)$ and $z_2 \in Nodes(y_{1_1}, r)$ in T such that $val(z_1) \neq val(z_2)$. Also, since q is a key it follows that $val(v) \neq val(w_n)$ in T. Consider then the path instances $v_1.\cdots.v$ and $w_1.\cdots.w_n$ in $Paths(q)$ in T. As we have already noted, $val(v) \neq val(w_n)$ so (i) of the definition of an XMVD is satisfied in T. Then since only node v is changed, if we let z_1, z_2 be as defined we have that $z_1 \in Nodes(x_{1_1}, r)$ and $z_2 \in Nodes(y_{1_1}, r)$ and $val(z_1) \neq val(z_2)$ and so (ii) of the definition of an XMVD is satisfied. Similarly, if we let z_3 and z_4 be as defined, then $z_3 \in Nodes(x_{1_{1_1}}, p)$ and $z_4 \in Nodes(y_{1_{1_1}}, p)$ and $val(z_3) = val(z_4)$ and so (iii) of the definition of an XMVD is satisfied in T. Hence by definition of XMVD satisfaction and since $p \rightarrow\rightarrow q|r$ is satisfies in T:

(a) there exists a path $v'_1.\cdots.v'_n$ in $Paths(q)$ in T such that $val(v'_n) = val(v_n)$ and there exists a node z''_1 in $Nodes(s'_{1_1}, r)$ such that $val(z''_1) = val(z_2)$ and there exists a node z''_3 in $Nodes(s'_{1_{1_1}}, p)$ such that $val(z''_3) = val(z_3)$ where $s'_{1_1} = \{v | v \in \{v'_1, \cdots, v'_n\} \wedge v \in N(r \cap q)\}$ and $s'_{1_{1_1}} = \{v | v \in \{v'_1, \cdots, v'_n\} \wedge v \in N(p \cap r \cap q)\}$;

and

(b) there exists a path $w'_1.\cdots.w'_n$ in $Paths(q)$ in T such that $val(w'_n) = val(w_n)$ and there exists a node z''_2 in $Nodes(t'_{1_1}, r)$ such that $val(z''_2) = val(z_1)$ and there exists a node z''_4 in $Nodes(t'_{1_{1_1}}, p_l)$ such that $val(z''_4) = val(z_4)$ where $t'_{1_1} = \{v | v \in \{v'_1, \cdots, v'_n\} \wedge v \in N(r \cap q)\}$ and $t'_{1_{1_1}} = \{v | v \in \{v'_1, \cdots, v'_n\} \wedge v \in N(p \cap r \cap q)\}$.

We claim that $v'_1.\cdots.v'_n$ must be distinct from $v_1.\cdots.v$. Suppose that it is not. Consider then the result of the change to v. If we let the path $v''_1.\cdots.v''_n$ in T' be the same as the path instance $v_1.\cdots.v_n$, and $w''_1.\cdots.w''_n$ be the same as $w_1.\cdots.w_n$ then, as we have seen, $v''_1.\cdots.v''_n$ and $w''_1.\cdots.w''_n$ satisfy (i), (ii) and (iii) of the definition of an XMVD. However, if we let $z'_1 = z''_1$ and $z'_3 = z''_3$ then z'_1 in $Nodes(s'_{1_1}, r)$ and $val(z'_1) = val(z_2)$ and z'_3 in $Nodes(s'_{1_{1_1}}, p)$ and $val(z'_3) = val(z_3)$ so (a) of the definition of an XMVD is satisfied. Similarly, if we let $z'_2 = z''_2$ and $z'_4 = z''_4$ then z'_2 in $Nodes(t'_{1_1}, r)$ and $val(z'_2) = val(z_1)$ and z'_4 in $Nodes(t'_{1_{1_1}}, p)$ and $val(z'_4) = val(z_4)$ so (b) of the definition of an XMVD is satisfied. This contradicts the earlier fact that either (a.1) or (b.1) is satisfied since $p \rightarrow\rightarrow q|r$ is violated in T' and so we conclude that $v'_1.\cdots.v'_n$ must be distinct from $v_1.\cdots.v$. However, by (a) above $val(v'_n) = val(v_n) = val(v)$ which contradicts the fact that q is a key.

Similarly, if $v \in N(r)$ then using the same arguments we contradict the fact that r is a key and so we conclude that Σ does not cause redundancy if (A) of the definition of 4XNF holds.

Assume next that (B) holds, i.e. p is a key and $q \cap r = p$, and suppose to the contrary that Σ causes redundancy. Then, as before, there exists an XML tree T which satisfies Σ and a node v in T such that *every* valid change from v to v', resulting in a new XML tree T', causes some XMVD $p \rightarrow\rightarrow q|r \in \Sigma$ to be violated. Using the same arguments as in (A), it follows that $v \in N(q)$ or $v \in N(r)$. We suppose firstly that $v \in N(q)$. So since $p \rightarrow\rightarrow q|r$ in T' is violated,

there exist path instances $v_1.\cdots.v_n$ (where $v_n = v'$) and $w_1.\cdots.w_n$ in $Paths(q)$ in T' such that:

(i) $val(v_n) \neq val(w_n)$;

(ii) there exists two nodes z_1, z_2, where $z_1 \in Nodes(x_{1_1}, r)$ and

$z_2 \in Nodes(y_{1_1}, r)$ such that $val(z_1) \neq val(z_2)$;

(iii) there exists two nodes z_3 and z_4, where $z_3 \in Nodes(x_{1_{1_1}}, p)$ and $z_4 \in Nodes(y_{1_{1_1}}, p)$, such that $val(z_3) = val(z_4)$.

and:

(a.1) there does not exist a path $v'_1.\cdots.v'_n$ in $Paths(q)$ such that $val(v'_n) = val(v_n)$ or there does not exist a node z'_1 in $Nodes(x'_{1_1}, r)$ such that $val(z'_1) = val(z_2)$ or there does not exist a node z'_3 in $Nodes(x'_{1_{1_1}}, p)$ such that $val(z'_3) = val(z_3)$;

or

(b.1) there does not exists a path $w'_1.\cdots.w'_n$ in $Paths(q)$ such that $val(w'_n) = val(w_n)$ or there does not exist a node z'_2 in $Nodes(x'_{1_1}, r)$ such that $val(z'_2) = val(z_1)$ or there does not exist a node z'_4 in $Nodes(x'_{1_{1_1}}, p_l)$ such that $val(z'_4) = val(z_4)$;

Consider then the paths $v_1.\cdots.v$ and $w_1.\cdots.w_n$ in $Paths(q)$ in T. Since p is a key and $q \cap r = p$, then $x_{1_1} = x'_{1_1} = y_{1_1} = x'_{1_1}$ and $x_{1_{1_1}} = x'_{1_{1_1}} = y_{1_{1_1}} = y'_{1_{1_1}}$. Hence if T' violates $p \rightarrow\rightarrow q|r$ then so will T which is a contradiction. The same argument applies if $v \in N(r)$ and so we conclude that Σ does not cause redundancy.

Assume next that (C) holds, i.e. p is a key and $q \cap r$ is a strict prefix of p and suppose to the contrary that Σ causes redundancy. Then, as before, there exists an XML tree T which satisfies Σ and a node v in T such that *every* valid change from v to v', resulting in a new XML tree T', causes some XMVD $p \rightarrow\rightarrow q|r \in \Sigma$ to be violated. Using the same arguments as in (A), it follows that $v \in N(q)$ or $v \in N(r)$. We suppose firstly that $v \in N(q)$. So since $p \rightarrow\rightarrow q|r$ in T' is violated, there exist path instances $v_1.\cdots.v_n$ (where $v_n = v'$) and $w_1.\cdots.w_n$ in $Paths(q)$ in T' such that:

(i) $val(v_n) \neq val(w_n)$;

(ii) there exists two nodes z_1, z_2, where $z_1 \in Nodes(x_{1_1}, r)$ and

$z_2 \in Nodes(y_{1_1}, r)$ such that $val(z_1) \neq val(z_2)$;

(iii) there exists two nodes z_3 and z_4, where $z_3 \in Nodes(x_{1_{1_1}}, p)$ and $z_4 \in Nodes(y_{1_{1_1}}, p)$, such that $val(z_3) = val(z_4)$.

and:

(a.1) there does not exist a path $v'_1.\cdots.v'_n$ in $Paths(q)$ such that $val(v'_n) = val(v_n)$ or there does not exist a node z'_1 in $Nodes(x'_{1_1}, r)$ such that $val(z'_1) = val(z_2)$ or there does not exist a node z'_3 in $Nodes(x'_{1_{1_1}}, p)$ such that $val(z'_3) = val(z_3)$;

or

(b.1) there does not exists a path $w'_1.\cdots.w'_n$ in $Paths(q)$ such that $val(w'_n) = val(w_n)$ or there does not exist a node z'_2 in $Nodes(x'_{1_1}, r)$ such that $val(z'_2) =$

$val(z_1)$ or there does not exist a node z'_4 in $Nodes(x'_{1_1}, p_l)$ such that $val(z'_4) = val(z_4)$.

We firstly note that because $q \cap r$ is a strict prefix of p, then $q \cap r = p \cap q \cap r$ and $z_3 = z_4$ since p is a key, this implies that $x_{1_1} = y_{1_1} = x_{1_{1_1}} = y_{1_{1_1}}$. Let the path instance $v'_1.\cdots.v'_n$ in T' be the path instance $v_1.\cdots.v_n$ and let $z'_1 = z_2$. Then we have that $x'_{1_1} = x_{1_1}$ and so z'_1 in $Nodes(x'_{1_1}, r)$ because $z_2 \in Nodes(y_{1_1}, r)$ and $x_{1_1} = y_{1_1}$. Thus condition (a) of the definition of an XMVD holds. Using similar arguments, if we let $w'_1.\cdots.w'_n$ in T' be the path instance $w_1.\cdots.w_n$ and let $z'_2 = z_1$ then part (b) of the definition of an XMVD holds. This contradicts the fact that either (a.1) or (b,1) holds and so we conclude that Σ does not cause redundancy.

Assume next that (D) holds, i.e. $q \cap r = root$, and suppose to the contrary that Σ causes redundancy. Then, as before, there exists an XML tree T which satisfies Σ and a node v in T such that *every* valid change from v to v', resulting in a new XML tree T', causes some XMVD $p \rightarrow\rightarrow q|r \in \Sigma$ to be violated. Using the same arguments as in (A), it follows that $v \in N(q)$ or $v \in N(r)$. We suppose firstly that $v \in N(q)$. So since $p \rightarrow\rightarrow q|r$ in T' is violated, there exist path instances $v_1.\cdots.v_n$ (where $v_n = v'$) and $w_1.\cdots.w_n$ in $Paths(q)$ in T' such that: (i), (ii), (iii) and (a.1) and (a.2). Then because $q \cap r = root$ then $x_{1_1} = y_{1_1} = x_{1_{1_1}} = y_{1_{1_1}}$. So if we let $v'_1.\cdots.v'_n$ in T' be the path instance $v_1.\cdots.v_n$ and let $z'_1 = z_2$ and $w'_1.\cdots.w'_n$ in T' be the path instance $w_1.\cdots.w_n$ and let $z'_2 = z_1$ then we derive a contradiction as in (C) and so we conclude that Σ does not cause redundancy.

Assume next that (E) holds, i.e. there exists an XMVD $s \rightarrow\rightarrow t|u \in \Sigma$ such that ($s \cap p = root$) and $t \geq q \cap r$ and t is a key and $u \geq q \cap r$ and u is a key.

Suppose to the contrary that Σ causes redundancy. Then, as before, there exists an XML tree T which satisfies Σ and a node v in T such that *every* valid change from v to v', resulting in a new XML tree T', causes some XMVD $p \rightarrow\rightarrow q|r \in \Sigma$ to be violated. Using the same arguments as in (A), it follows that $v \in N(q)$ or $v \in N(r)$. We suppose firstly that $v \in N(q)$. So since $p \rightarrow\rightarrow q|r$ in T' is violated, there exist path instances $v_1.\cdots.v_n$ (where $v_n = v'$) and $w_1.\cdots.w_n$ in $Paths(q)$ in T' such that: (i), (ii), (iii) and (a.1) and (a.2).

Now because of (a.1) we must have that $x_{1_1} \neq y_{1_1}$ in T' or else by choosing $v'_1.\cdots.v'_n$ to be the path instance $v_1.\cdots.v_n$ and $z'_1 = z_2$ we contradict (a.1). So it then follows that since only node v is changed, $x_{1_1} \neq y_{1_1}$ in T. We assume firstly that p is a prefix of $q \cap r$. By Lemma 1 the only possible cases that could give rise to redundancy are shown in Figure 6.

Consider case **(a)** of Figure 6. We claim that this case cannot arise. To verify this, in case **(a)** it follows that $x_{1_1} = y_{1_1}$ and $x_{1_{1_1}} = y_{1_{1_1}}$. Consider then the satisfaction of $p \rightarrow\rightarrow q|r$ in T'. Then, as already noted, the path instances $v_1.\cdots.v_n$ (where $v_n =$) $w_1.\cdots.w_n$ in $Paths(q)$ in T' satisfy (i), (ii) and (iii) above. Let the path instance in $v'_1.\cdots.v'_n$ in $Paths(q)$ be instances $v_1.\cdots.v_n$ and let $z'_1 = z_2$ and let $z'_3 = z_3$. Then since $x_{1_1} = y_{1_1}$ it follows

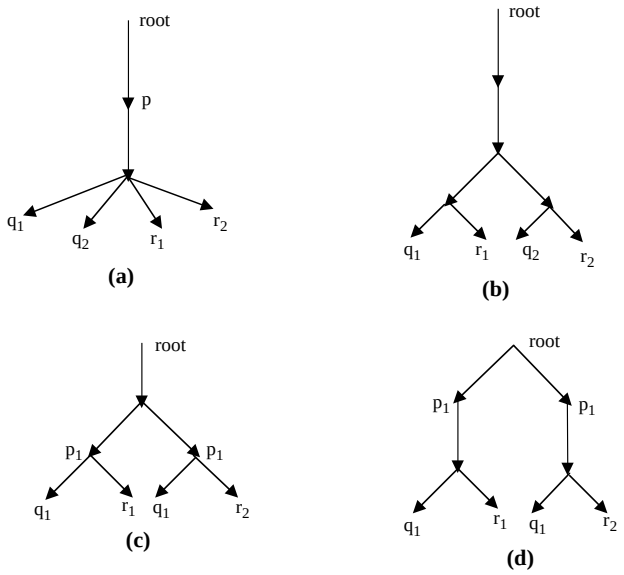

Fig. 6.

that z'_1 in $Nodes(x'_{1_1}, r)$ and $val(z'_1) = val(z_2)$ and z'_3 in $Nodes(x'_{1_{1_1}}, p)$ and $val(z'_3) = val(z_3)$ and so part (a) of the definition of an XMVD is satisfied. Next, let the path instance $w'_1.\cdots.w'_n$ in $Paths(q)$ be $w_1.\cdots.w_n$ and let $z'_2 = z_1$ and let $z'_4 = z_4$. Then since $x_{1_1} = y_{1_1}$ it follows that z'_2 in $Nodes(x'_{1_1}, r)$ and $val(z'_2) = val(z_1)$ and z'_4 in $Nodes(x'_{1_{1_1}}, p)$ and $val(z'_4) = val(z_4)$ and so part (b) of the definition of an XMVD is satisfied. However, this contradicts the assumption that $v_1.\cdots.v_n$ (where $v_n =) \ w_1.\cdots.w_n$ satisfies (i), (ii) and (iii) but violates either (a) or (b) and so we conclude that case (a) in Figure ?? cannot arise.

Consider next the case **(b)** shown in Figure 6. Suppose firstly that $q_1 = q_2$. We claim that this case cannot arise either. To see this, because of (E) the situation is as shown in Figure 7.

Let us denote by v_{t_1} any node in $Nodes(x^1_{1_1}, t)$ in T', where $x^1_{1_1} = \{v | v \in \{v_1, \cdots, v_n\} \wedge v \in N(t \cap q)\}$, and let v_{t_2} any node in $Nodes(x^2_{1_1}, t)$ in in T', where $x^2_{1_1} = \{v | v \in \{w_1.\cdots.wn\} \wedge v \in N(t \cap q)\}$. Then we note that since $t \geq q \cap r$ and t is a key it follows that $val(v_{t_1}) \neq val(v_{t_2})$. Similarly, let us denote by v_{u_1} any node in $Nodes(x^3_{1_1}, t)$ in T', where $x^3_{1_1} = \{v | v \in \{v_1, \cdots, v_n\} \wedge v \in N(u \cap q)\}$, and let v_{u_2} any node in $Nodes(x^4_{1_1}, t)$ in T', where $x^4_{1_1} = \{v | v \in \{w_1.\cdots.wn\} \wedge v \in N(u \cap q)\}$. Then we note that since $u \geq q \cap r$ and u is a key it follows that $val(v_{u_1}) \neq val(v_{u_2})$. Consider then the path instances in $N(t)$ that ends in v_{t_1}, call it p_{t_1}, and the path instances in $N(t)$ that ends in v_{t_2}, call it p_{t_2}. As we have already shown, $val(v_{t_1}) \neq val(v_{t_2})$ and so p_{t_1} and p_{t_2} satisfy condition (i) of the definition of an XMVD. They also satisfy (ii) of the definition of an XMVD if

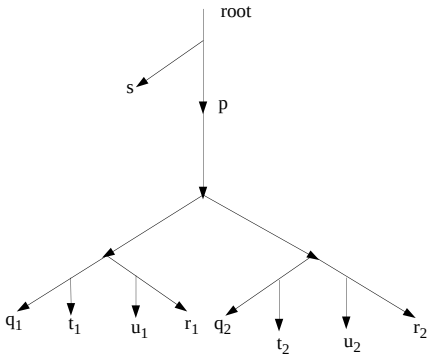

Fig. 7.

we let $z_1 = v_{u_1}$ and $z_2 = v_{u_2}$ because, as we have seen, $val(v_{u_1}) \neq val(v_{u_2})$. The path instances p_{t_1} and p_{t_2} also satisfy condition (iii) of the definition of an XMVD since $s \cap p = root \vee p \geq s$ and so, as can be seen from 7, we have that $x^1_{1_{1_1}} = y^1_{1_{1_1}}, p)$, where $x^1_{1_{1_1}} = \{v | v \in p_{t_1} \wedge v \in N(s \cap t \cap u)\}$ and $y^1_{1_{1_1}} = \{v | v \in p_{t_2} \wedge v \in N(s \cap t \cap u)\}$ and so $val(z_3) = val(z_4)$. Thus all the conditions of an XMVD are satisfied. So since only node v in T is changed, if p_{t_1} and p_{t_1} satisfy conditions (i), (ii) and (iii) of an XMVD in T' then they will will be the same in T and so p_{t_1} and p_{t_1} satisfy conditions (i), (ii) and (iii) of an XMVD in T. Hence, from (a) of the definition of an XMVD, there must exist a path $v'_1.\cdots.v'_n$ in $Paths(t)$ in T such that $val(v'_n) = val(v_{t_1})$ and there exists a node z'_1 in $Nodes(x'^1_{1_1}, u)$ such that $val(z'_1) = val(z_2)$ and there exists a node z'_3 in $Nodes(x'_{1_{1_1}}, s)$ such that $val(z'_3) = val(z_3)$, where $x'_{1_1} = \{v | v \in \{v'_1, \cdots, v'_n\} \wedge v \in N(t \cap u)\}$ and $x'_{1_{1_1}} = \{v | v \in \{v'_1, \cdots, v'_n\} \wedge v \in N(p \cap r \cap q)\}$. However, we cannot have that $v'_1.\cdots.v'_n = p_{t_1}$ or else it would imply that there was a node in $Nodes(v_{t_1}, t \cap u)$ which had the same val as v_{t_2} which contradicts the fact that t is a key. Also, we cannot have that $v'_1.\cdots.v'_n = p_{t_2}$ since we require that $val(v'_n) = val(v_{t_1})$. Thus $v'_1.\cdots.v'_n$ must be distinct from both p_{t_1} and p_{t_2} and so z'_1 is distinct from z_2 (which is v_{u_2}) but $val(z'_1) = val(v_{u_2})$ which contradicts the fact that u is a key. Hence if $q_1 = q_2$ then case (b) cannot arise.

Suppose then that case (b) arises and $q_1 \neq q_2$. The same arguments just used show that this situation cannot arise.

Cases **(c)** and **(d)** in Figure 6 are treated using the same argument as in **(b)**.

Consider next the case where $q \cap r < p$. By Lemma 1 the only possible situation where redundancy could arise is shown in Figure 8

Similar arguments to the one given for the previous case show that this situation cannot arise.

Finally, consider the case where $q \cap r$ and p are incomparable. By Lemma 1 the only possible situations where redundancy could arise is shown in Figure 9

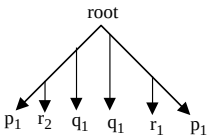

Fig. 8.

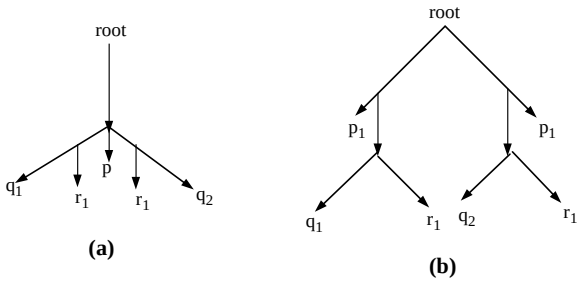

Fig. 9.

Similar arguments to the one given for the previous case show that these situations cannot arise.

Consider (F), there exists an XMVD $a \rightarrow\rightarrow b|c \in \Sigma$ such that $a \cap p = root$ and b is not a key and c is a key and $b \cap c \cap p \neq root$ and $\exists b \rightarrow\rightarrow d|e$ such that d is a key and e is a key and $d \geq q \cap r$ and $e \geq q \cap r$. By Lemma 1, the only possible cases where redundancy could arise are shown in Figure 10.

Consider case **(a)** in Figure 10. Using the same arguments as in case (E) shows that this case cannot arise. Consider then case **(b)** in Figure 10. If not, then using the same arguments as in case (E), this implies that there is another distinct path instance for the path c which has the same val as the final node in the path instance c_1 shown in Figure 10. This contradicts the fact that c is a key and so we conclude that the val of the two path instances of b shown in Figure 10 must be the same. However, if this is the case then using the same arguments asin case (E) we derive that there is another distinct path instance for the path d which has the same val as the final node in the path instance d_1. This contradicts the fact that d is a key and so case **(b)** in Figure 10 cannot arise..

Suppose now that (G) is true, i.e. q is not a key and r is not a key and exists $p \rightarrow\rightarrow q|k$ and exists $p \rightarrow\rightarrow k|r$ and $k \geq q \cap r$. Suppose to the contrary that Σ causes redundancy.

The first subcase we consider is where $q \cap r > p$. By Lemma 1 four possible situations can arise as shown in Figure 11.

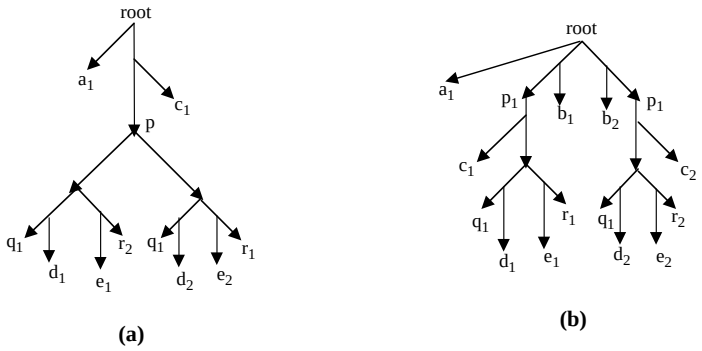

Fig. 10.

As shown previously, case **(a)** in Figure 11 cannot arise. Consider case **(b)**. We claim that this case cannot arise also.

To see this, the situation is as shown in Figure 12.

Let us denote by v_{t_1} any node in $Nodes(x^1_{1_1}, k)$ in T', where $x^1_{1_1} = \{v | v \in \{v_1, \cdots, v_n\} \wedge v \in N(k \cap q)\}$, and let v_{t_2} any node in $Nodes(x^2_{1_1}, k)$ in in T', where $x^2_{1_1} = \{v | v \in \{w_1. \cdots .wn\} \wedge v \in N(k \cap q)\}$. Then we note that since $k \geq q \cap r$ and k is a key it follows that $val(v_{t_1}) \neq val(v_{t_2})$. Consider then the path instances in $N(k)$ that ends in v_{t_1}, call it p_{t_1}, and the path instances in $N(k)$ that ends in v_{t_2}, call it p_{t_2}. As we have already shown, $val(v_{t_1}) \neq val(v_{t_2})$ and so p_{t_1} and p_{t_2} satisfy condition (i) of the definition of an XMVD. They also satisfy (ii) of the definition of an XMVD if we let $z_1 = v_n$ and $z_2 = w_n$ because, as we have seen, $val(v_n) \neq val(w_n)$. The path instances p_{t_1} and p_{t_2} also satisfy condition (iii) of the definition of an XMVD since as can be seen from Figure 12, we have that $x^1_{1_{1_1}} = y^1_{1_{1_1}}$,, where $x^1_{1_{1_1}} = \{v | v \in p_{t_1} \wedge v \in N(p \cap k \cap q)\}$ and $y^1_{1_{1_1}} = \{v | v \in p_{t_2} \wedge v \in N(p \cap k \cap q)\}$ and so $val(z_3) = val(z_4)$. Thus all the conditions of an XMVD are satisfied. So since only node v in T is changed, if p_{t_1} and p_{t_1} satisfy conditions (i), (ii) and (iii) of an XMVD in T' then they will will be the same in T and so p_{t_1} and p_{t_1} satisfy conditions (i), (ii) and (iii) of an XMVD in T. Hence, from (a) of the definition of an XMVD, there must exist a path $v'_1. \cdots .v'_n$ in $Paths(t)$ in T such that $val(v'_n) = val(v_{t_1})$. However, we cannot have that $v'_1. \cdots .v'_n = p_{t_1}$ or else it would imply that there was a node in $Nodes(v_{t_1}, k \cap q)$ which had the same val as v_{t_2} which contradicts the fact that k is a key. Also, we cannot have that $v'_1. \cdots .v'_n = p_{t_2}$ since we require that $val(v'_n) = val(v_{t_1})$. Thus $v'_1. \cdots .v'_n$ must be distinct from both p_{t_1} and p_{t_2} and so z'_1 is distinct from z_2 (which is v_{u_2}) but $val(z'_1) = val(v_{u_2})$ which contradicts the fact that k is a key. So case **(b)** cannot arise. Similarly we can show that cases **(c)** and **(d)** in Figure 6 cannot arise.

The next subcase we consider is where $q \cap r < p$. The only situation that could possibly give rise to redundancy is shown in Figure 13. However, using

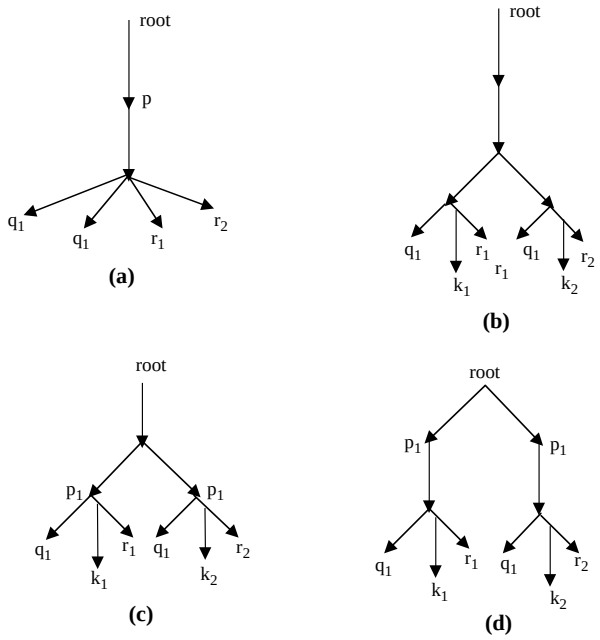

Fig. 11.

similar arguments to the previous case, this situation cannot arise as it would imply the contradiction that k is not a key.

The final subcase we consider is where p and $q \cap r$ are incomparable. In this case the only two situations where redundancy could possibly arise are shown in **(a)** and **(b)** of Figure. 14.

However using the same reasoning as previously we conclude that these cases cannot arise or else they would contradict the fact that k is a key.

Consider now Case (H), i.e. p is a key, q is a key and r is not a key and $q \cap r \neq p$ and $q \cap r$ is not a strict prefix of p and $\exists x \twoheadrightarrow q|k$ such that $x < p$ and $k > p \cap q \cap r$ and $k \cap q \cap r \neq p \cap q \cap r$.

The only situation where redundancy could possibly arise is shown in Figure 15.

The same argument as previously shows that this situation cannot arise or else it contradicts the fact that K is a key.

Finally, for case (I), i.e. p is a key, q isnot a key and r is a key and $q \cap r \neq p$ and $q \cap r$ is not a strict prefix of p and $\exists x \twoheadrightarrow k|r$ such that $x < p$ and $k > p \cap q \cap r$ and $k \cap q \cap r \neq p \cap q \cap r$, by symmetry the same argument as used in (H) applies. □

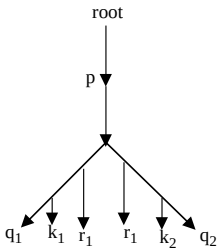

Fig. 12.

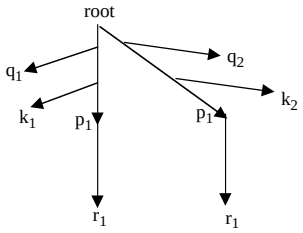

Fig. 13.

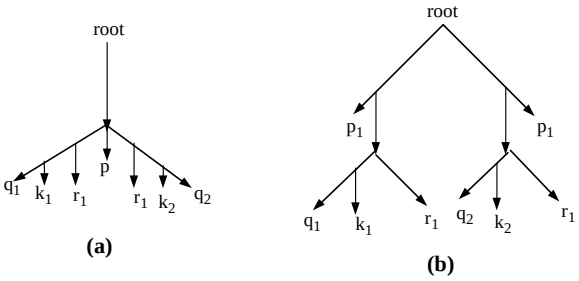

Fig. 14.

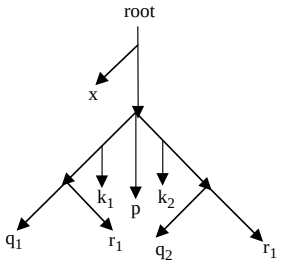

Fig. 15.