World Wide Web (2009) 12:87–105
DOI 10.1007/s11280-008-0057-x

Independence of Containing Patterns Property and Its Application in Tree Pattern Query Rewriting Using Views

Junhu Wang · Jeffrey Xu Yu · Chengfei Liu

Received: 14 May 2008 / Revised: 7 August 2008 /
Accepted: 5 November 2008 / Published online: 4 December 2008

Abstract We show that several classes of tree patterns observe the *independence of containing patterns* property, that is, if a pattern is contained in the union of several patterns, then it is contained in one of them. We apply this property to two related problems on tree pattern rewriting using views. First, given view V and query Q, is it possible for Q to have an equivalent rewriting using V which is the union of two or more tree patterns, but not an equivalent rewriting which is a single pattern? This problem is of both theoretical and practical importance because, if the answer is no, then, to find an equivalent rewriting of a tree pattern using a view, we should use more efficient methods, such as the polynomial time algorithm of Xu and Özsoyoglu (2005), rather than try to find the union of all contained rewritings (which takes exponential time in the worst case) and test its equivalence to Q. Second, given a set S of views, we want to know under what conditions a subset S' of S is redundant in the sense that for *any query* Q, the contained rewritings of Q using the views in S' are contained in those using the views in $S - S'$. Solving this problem can help us to, for example, choose the minimum number of views to be cached, or better design the virtual schema in a mediated data integration system, or avoid repeated calculation in query optimization. For the first problem, we identify several classes of tree patterns for which the equivalent rewriting can be expressed as a single tree pattern. For the second problem, we present necessary and sufficient conditions for S' to be redundant with respect to some classes of tree patterns. For both problems

J. Wang (✉)
Griffith University, Gold Coast, Australia
e-mail: J.Wang@griffith.edu.au

J. X. Yu
Chinese University of Hong Kong, Hong Kong, China
e-mail: yu@se.cuhk.edu.hk

C. Liu
Swinburne University of Technology, Melbourne, Australia
e-mail: cliu@ict.swin.edu.au

we consider extension to cases where there are rewritings using the intersection of multiple views and/or where a schema graph is present.

Keywords XPath · tree pattern · views · containment · rewriting

1 Introduction

Query rewriting using views has many applications including data integration, query optimization, and query caching [5]. A view is an existing query whose answer may or may not have been materialized. Given a new query, the problem is to find another query using only the views that will produce correct answers to the original query. Usually two types of rewritings are sought: *equivalent rewritings* and *contained rewritings*. An equivalent rewriting produces all answers to the original query, while a contained rewriting may produce only part of the answers. Both types of rewritings have been extensively studied in the relational database context, see [5] for an early survey, and [12, 17] for more recent developments.

More recently rewriting XML queries using XML views has attracted attention because of the rising importance of XML data [6, 9, 13, 20]. Since XPath lies in the center of all XML languages, the problem of rewriting XPath queries using XPath views is particularly important. Some major classes of XPath expressions can be represented as tree patterns [1, 10]. Among previous work on rewriting XPath queries using views, Xu et al. [20] studied equivalent rewritings for several different classes of tree patterns, and it gave a polynomial time algorithm for finding equivalent rewritings when the tree patterns do not have *. Mandhani and Suciu [9] presented results on equivalent rewritings of tree patterns when the tree patterns are assumed to be minimized. Lakshmanan et al. [6] studied maximal contained rewritings of tree patterns where both the view and the query involve /, // and [] only (these XPath expressions correspond to tree patterns in $P^{\{/,//,[]\}}$ [10]), both in the absence and presence of non-recursive schema graphs—a restricted form of DTDs. When there are no DTDs, the worst case complexity of finding the maximal contained rewriting is shown to be exponential in the size of the query.

In this paper, we study two related problems on XPath rewritings using views. The first problem is about the form of equivalent rewritings: given view V and query Q, is it possible for Q to have an equivalent rewriting using V which is the union of two or more tree patterns, but not an equivalent rewriting which is a single tree pattern? This problem is of both theoretical and practical importance because, if the answer is no, then, to find an equivalent rewriting using the view, we can use more efficient methods such as the polynomial time algorithm of [20], rather than try to find the union of all contained rewritings [6] and test its equivalence to Q. The second problem is what we call the *redundant views* problem. Given a set S of views, we want to know under what conditions a subset S' of S is redundant in the sense that for *any query* Q, the contained rewritings of Q using the views in S' are contained in those using the views in $S - S'$. Thus the contribution of redundant views to the contained rewritings can be ignored. Solving this problem can help us to, for example, choose the minimum number of views to be cached, or better design the virtual schema in a mediated data integration system, or avoid useless computation

in query optimization. We first study the above problems for the class of tree patterns involving /, // and [], and then extend our results to other classes of tree patterns. We also consider the case where the intersection of views is used in the rewriting, as well as the case a schema graph is present.

Our main contributions are:

- We show that, when there is no DTD, several classes of tree patterns observe the *independence of containing patterns* property, that is, if a tree pattern is contained in the union of multiple tree patterns, then it must be contained in one of them.
- Using the above property, we show that for queries and views in $P^{\{/,//,[]\}}$, if there is no equivalent rewriting in the form of a single tree pattern, then there is no equivalent rewriting in the form of a union of tree patterns. We extend this result to queries in $P^{\{/,[],*\}}$, queries and views in $\widehat{P}^{\{/,//,[],*\}}$, and rewritings using multiple views. We also consider the presence of DTDs.
- When multiple views exist, we provide a necessary and sufficient condition for identifying redundant views. In the case where the intersection of views is also used in the rewriting, we provide a necessary condition and a separate sufficient condition for redundant views.

The rest of the paper is organized as follows. Section 2 provides the terminology and notations. Section 3 shows the independence of containing patterns property of tree patterns. Based on this property, Section 4 presents our result on the form of equivalent rewritings. Section 5 then discusses conditions under which some views are redundant. Section 6 discusses related work. Finally Section 7 concludes the paper.

2 Preliminaries

2.1 DTDs, XML trees, and tree patterns

Let Σ be an infinite set of tags. We adopt the similar notations used in [6], and model a DTD as a connected directed graph G (called a *schema graph*) satisfying the following conditions:

(1) each node is labeled with a distinct label in Σ;
(2) each edge is labeled with one of 1, ?, +, and $*$, which indicate "exactly one", "one or zero", "one or many", and "zero or many", respectively. Here, the default edge label is $*$;
(3) there is a unique node, called the root, which *may* have an incoming degree of zero. All other nodes have incoming degrees greater than 0.

Because a node in a schema graph G has a unique label, we also refer to a node by its label. If the graph is acyclic, then the DTD is said to be *non-recursive*. We will use DTD and schema graph interchangeably.[1] Two example schema graphs are shown in Figure 1(a) and (b), the first one is non-recursive, and the second one is recursive.

An XML *tree* is a node-labeled, unordered tree. Let v be a node in an XML tree t, the label of v is denoted by $label(v)$. Let $N(t)$ (resp. $N(G)$) denote the set of all nodes in

[1] A schema graph cannot model constructs such as $a := (b, b^?)$ and $a := (b \mid c)$ in a DTD.

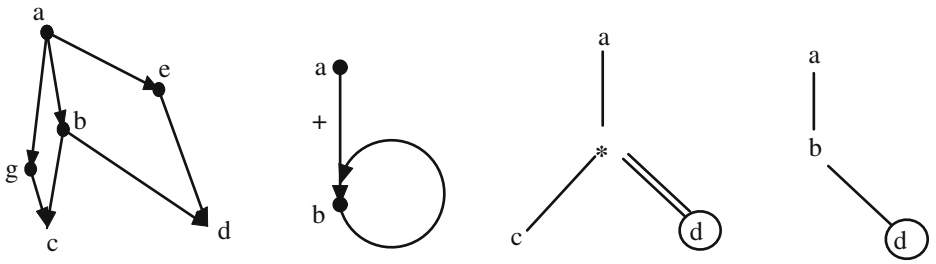


Figure 1 Schema graphs G_1, G_2 and tree patterns P_1, P_2.

XML tree t (resp. schema graph G), and $rt(t)$ (resp. $rt(G)$) denote the root of t (resp. G). A tree t is said to *conform* to schema graph G if

(1) for every node $v \in N(t)$, $label(v) \in \Sigma$,
(2) $label(rt(t)) = label(rt(G))$,
(3) for every edge (u, v) in t, there is a corresponding edge $(label(u), label(v))$ in G, and
(4) for every node $v \in N(t)$, the number of children of v labeled with x is constrained by the label of the edge $(label(v), x)$ given in G.

We denote the set of all XML trees conforming to G by T_G.

We consider a class of XPath expressions given as follows.

$$P ::= \tau \mid * \mid P/P \mid P//P \mid P[P] \mid P[//P]$$

Here, $\tau \in \Sigma$, $*$ is the wildcard representing any tag in Σ, and, /, //, and [] represent the child-axis, descendant-axis, and branching condition, respectively. The class of XPath expressions corresponds to a set of *tree patterns* (TP) known as $P^{\{/,//,[],*\}}$ in [10]. Formally, a *tree pattern (*TP*)* in $P^{\{/,//,[],*\}}$ is a tree such that each edge is labeled with either / or //; each node is labeled with a tag in Σ or the wildcard $*$, and there is a *distinguished node* corresponding to the output of the XPath expression. When there is no confusion, we will use TP and XPath query interchangeably. A tree patten has a tree representation. Figure 1(c) and (d) show two TPs. They correspond to the XPath expressions $a/*[c]//d$ and $a/b/d$, respectively. Here, single lines represent edges labeled with /, called /-edges, and double lines represent edges labeled with //, called //-edges. A branch in the tree representation represents a condition ([]) in an XPath expression, and a circle indicates the distinguished node of P. Below, given a TP P, we use $\mathtt{DN}_P$ to denote the distinguished node. The path from $rt(P)$ to $\mathtt{DN}_P$ is called the *distinguished path*.

The following subsets of $P^{\{/,//,[],*\}}$ are of special interest to us. $P^{\{/,//,[]\}}$ is the set of TPs that do not have *-nodes (i.e., nodes labeled with *), $P^{\{/,[],*\}}$ is the set of TPs that do not have //-edges, and $\widehat{P}^{\{/,//,[],*\}}$ is the set of TPs such that no //-edge is incident on a *-node, and there are no leaf *-nodes. Note that $P^{\{/,//,[]\}}$ is a subset of $\widehat{P}^{\{/,//,[],*\}}$.

Let $N(P)$ (resp. $rt(P)$) denote the set of all nodes in a TP P (resp. the root of P). A *matching* of P in an XML tree t is a mapping δ from $N(P)$ to $N(t)$ which is

(1) *root-preserving*, i.e., $\delta(rt(P)) = rt(t)$,
(2) *label-preserving*, i.e., $\forall v \in N(P)$, $label(v) = label(\delta(v))$ or $label(v) = *$, and

(3) *structure-preserving*, i.e., for every edge (x, y) in P, if it is a /-edge, then $\delta(y)$ is a child of $\delta(x)$; if it is a //-edge, then $\delta(y)$ is a descendant of $\delta(x)$, i.e, there is a path from $\delta(x)$ to $\delta(y)$.

Each matching δ produces a subtree of t rooted at $\delta(\mathtt{DN}_P)$, denoted $sub^t_{\delta(\mathtt{DN}_P)}$, which is also known as an *answer* to the TP. We use $P(t)$ to denote the set of all answers of P on t:

$$P(t) = \left\{sub^t_{\delta(\mathtt{DN}_P)} \mid \delta \text{ is a matching of } P \text{ in } t\right\} \quad (1)$$

If $P(t) = \emptyset$ for every XML tree t, then P is said to be an *empty query*, denoted $P(t) = \emptyset$.

Let T be a set of XML trees. We use $P(T)$ to denote the union of answer sets of Q on the trees in T. That is, $P(T) = \bigcup_{t\in T} P(t)$. In addition, when we discuss TPs in the presence of DTD G, we will implicitly assume every TP P is *satisfiable* under G, that is, there is $t \in T_G$ such that $P(t) \neq \emptyset$.

2.2 Tree pattern containment and containment mapping

A TP P is said to be *contained* in another TP Q, denoted $P \subseteq Q$, if for every XML tree t, $P(t) \subseteq Q(t)$ (Refer to Eq. 1). Given a DTD G and two TPs P and Q, P is said to be *contained in Q under G*, denoted $P \subseteq_G Q$, if for every XML tree $t \in T_G$, $P(t) \subseteq Q(t)$. Tree pattern equivalence is defined as two-way containment as usual. That is, $P = Q$ is defined as $P \subseteq Q$ and $Q \subseteq P$, and $P =_G Q$ means $P \subseteq_G Q$ and $Q \subseteq_G P$.

When there are no DTDs, the containment of some classes of tree patterns can be characterized by the existence of a containment mapping. Recall [1]: a *containment mapping* (CM) from Q to P is a mapping h from $N(Q)$ to $N(P)$ that is label-preserving, root-preserving, structure-preserving (which now means that for every /-edge (x, y) in Q, $(h(x), h(y))$ is a /-edge in P, and for every //-edge (x, y), there is a path from $h(x)$ to $h(y)$), and output-preserving, which means $h(\mathtt{DN}_Q) = \mathtt{DN}_P$. The following lemma is proved in [11].

Lemma 1 *In the following cases, $P \subseteq Q$ iff there is a* CM *from Q to P* [11].

(1) $Q \in P^{\{/,//,[]\}}$,
(2) $P \in P^{\{/,[],*\}}$,
(3) $Q \in P^{\{/,[],*\}}$, *and there are no leaf *-nodes in Q.*

A closer inspection of the proof in [11] shows the following lemma is also true.

Lemma 2 *Suppose $Q \in \widehat{P}^{\{/,//,[],*\}}$. Then for any pattern $P \in P^{\{/,//,[],*\}}$, $P \subseteq Q$ iff there is a* CM *from Q to P.*

2.3 Contained rewriting, maximal contained rewriting and equivalent rewriting

A *view* is a pre-defined TP. Let V be a view and Q be a TP. A *contained rewriting* (CR) of Q using V is a TP Q' such that when evaluated on the subtrees returned by V, Q' gives correct answers to Q. More precisely, (1) for any XML tree t, $Q'(V(t)) \subseteq Q(t)$, and (2) there exists some t such that $Q'(V(t)) \neq \emptyset$.

Let Q' be a CR of Q. We use $Q' \circ V$ to represent the *expansion* of Q', which is the TP obtained by merging $rt(Q')$ and $\mathtt{DN}_V$ as follows: if $label(\mathtt{DN}_V) \neq *$, then

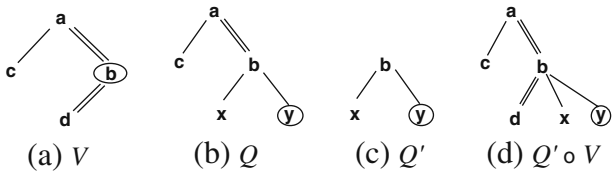


Figure 2 View (**a**), query (**b**), CR (**c**), and expansion (**d**).

the merged node is labeled $label(\mathtt{DN}_V)$, otherwise the merged node is labeled $label(rt(Q'))$ (Note that if $label(\mathtt{DN}_V) \neq *$, $label(rt(Q')) \neq *$ then $label(rt(Q')) = label(\mathtt{DN}_V)$). The distinguished node of $Q' \circ V$ is the distinguished node of Q'. Figure 2 shows a view V, a TP Q, a CR Q' of Q using V, and the expansion $Q' \circ V$ of Q'.

It is easy to see that $(Q' \circ V)(t) = Q'(V(t))$. Thus condition (1) in the definition of CR is equivalent to $Q' \circ V \subseteq Q$, and condition (2) in the definition is equivalent to $Q' \circ V \neq \emptyset$. A CR Q' is said to be an *equivalent rewriting* (ER) if $Q' \circ V \supseteq Q$ also holds. The *maximal contained rewriting* (MCR) of Q using V, denoted MCR(Q, V), is the union of all CRs of Q using V [6]. We use EMCR(Q, V) to denote the union of expansions of all of the CRs in MCR(Q, V). Using the concepts of useful embedding [6] and revised useful embedding [19], it can be easily proved that, when Q is in $P^{\{/,//,[]\}}$ or $\widehat{P}^{\{/,//,[],*\}}$, the MCR of Q using V is the union of a finite number of CRs of Q using V. That is, there are CRs $Q_1 \ldots Q_m$ such that EMCR$(Q, V) = Q_1 \circ V \cup \cdots \cup Q_m \circ V$. For convenience, we define EMCR(Q, V) to be the empty query if Q has no CRs using V.

In the presence of DTDs, a *contained rewriting (resp. equivalent rewriting) of Q using view V under* DTD G is a TP Q' such that (1) for any XML tree $t \in T_G$, $Q'(V(t)) \subseteq Q(t)$ (resp. $Q'(V(t)) = Q(t)$), (2) for some $t \in T_G$, $Q'(V(t)) \neq \emptyset$. The MCR of Q using V under G is the union of all CRs of Q using V under G.

3 Independence of containing patterns property of tree patterns

In this section, we show that some classes of tree patterns observe what we call the *independence of containing patterns (ICP)* property, that is, if a TP is contained in the union of several other TPs, it is contained in one of them.

Theorem 1 *Let P and $P_1, \ldots, P_n$ be tree patterns, in the following cases, if $P \subseteq \bigcup_{i=1}^{n} P_i$, then there exists $i \in [1, n]$ such that $P \subseteq P_i$.*

(1) $P \in P^{\{/,//,[],*\}}$, $P_1, \ldots, P_n \in \widehat{P}^{\{/,//,[],*\}}$, *or*
(2) $P \in P^{\{/,[],*\}}$, $P_1, \ldots, P_n \in P^{\{/,//,[],*\}}$.

To prove the above theorem, we need the concept of boolean tree patterns. A *boolean pattern* [10] is a pattern with no distinguished node. Let P be a boolean pattern. For an XML tree t, the result of evaluating P on t, denoted $P(t)$, is either TRUE or FALSE. $P(t)$ is TRUE if and only if there is a matching of P in t. For two boolean patterns P_1 and P_2, $P_1 \subseteq P_2$ means $P_1(t) =$TRUE implies $P_2(t) =$TRUE for any XML tree t. If P_2 is in $\widehat{P}^{\{/,//,[],*\}}$, then $P_1(t) \subseteq P_2(t)$ iff there is a homomorphism from P_2 to P_1 (Recall: a homomorphism is the same as a containment mapping, except it does

not need to be output-preserving) [10]. In addition, $P_1 \cup P_2$ returns $P_1(t) \vee P_2(t)$ for any t.

We first prove the following lemma.

Lemma 3 *For boolean patterns* $P, P_1, \ldots, P_n \in P^{\{/,//,[],*\}}$, *if* $P \in P^{\{/,[],*\}}$, *or* $P_1, \ldots, P_n \in \widehat{P}^{\{/,//,[],*\}}$, *then* $P \subseteq \bigcup_i P_i$ *implies there exists* $i \in [1, n]$ *such that* $P \subseteq P_i$.

Proof We prove the lemma for the case $n = 2$. The case $n > 2$ is similar.

Using Lemma 1 of [10], we can construct two boolean patterns Q and Q' such that $P \subseteq P_1 \cup P_2$ iff $Q \subseteq Q'$. Q and Q' are as shown in Figure 3, where V is a pattern which is contained in both P_1 and P_2, and V does not have *-nodes or //-edges (see [10] for how to construct V). Furthermore, c can be chosen as a label which does not appear in either P_1 or P_2. Note that (1) if P_1, P_2 are in $\widehat{P}^{\{/,//,[],*\}}$, then $Q' \in \widehat{P}^{\{/,//,[],*\}}$, (2) if $P \in P^{\{/,[],*\}}$, then $Q \in P^{\{/,[],*\}}$. Since $P \subseteq P_1 \cup P_2$ implies $Q \subseteq Q'$, by Lemma 1, we know there is a homomorphism from Q' to Q. Now examine the structure of Q and Q', any homomorphism from Q' to Q must map the nodes u_1 and u_2 either to v_1 and v_2 respectively, or to v_2 and v_3 respectively. In the former case, there will be a homomorphism from P_2 to P; in the latter case there will be a homomorphism from P_1 to P. Therefore, either $P \subseteq P_1$ or $P \subseteq P_2$. $\perp$ □

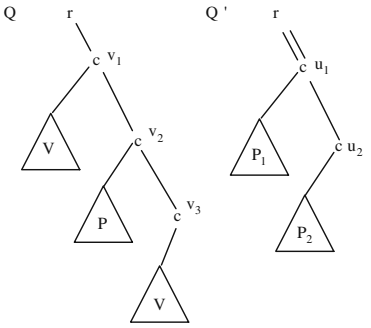
We are now ready to prove Theorem 1.

Proof of Theorem 1 We denote by P', P'_i ($i \in [1, n]$) the boolean patterns obtained from P, P_i by attaching a child node labeled with a *distinct* label z to the distinguished nodes of P and P_i respectively (Since Σ is an infinite set of tags, z exists, refer to Figure 4). Let us denote the new nodes in P and P_i by z_P and z_{P_i} respectively.

We show that $P \subseteq \bigcup_{i\in[1,n]} P_i$ implies $P' \subseteq \bigcup_{i\in[1,n]} P'_i$. Let t be any XML tree. For every matching h of P' in t, there is a matching of P in t which is the one obtained by restricting h to all nodes in P except z_P. Since $P \subseteq \bigcup_{i\in[1,n]} P_i$, there exists an $i \in [1, n]$ such that there is a matching f of P_i in t and $f(\mathtt{DN}_{P_i}) = h(\mathtt{DN}_P)$. This matching f can clearly be extended to P'_i - simply let $f(z_{P_i}) = h(z_P)$. Therefore $P' \subseteq \bigcup_{i\in[1,n]} P'_i$.

By Lemma 3, there exists $i \in [1, n]$ such that $P' \subseteq P'_i$. Therefore, there is a homomorphism from P'_i to P'. This homomorphism implies a containment mapping from P_i to P. Hence $P \subseteq P_i$. $\perp$ □

Figure 3 Q and Q' as in [10].



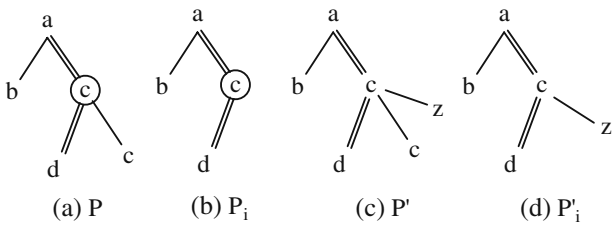


Figure 4 Tree patterns P, P_i and the boolean patterns P' and P'_i.

Theorem 1 does not hold in the presence of DTDs. For example, consider the DTD in Figure 1a. Under the DTD, $a//d \subseteq a/b/d \cup a/e/d$. But $a//d$ is contained in neither $a/b/d$ nor $a/e/d$. Also $a/*$ is contained in the union of a/g, a/b and a/e, but it is not contained in any of them.

4 Equivalent rewritings, single pattern or union of patterns?

Let V be the view, and Q be the query. By definition, $\mathsf{EMCR}(Q, V) \subseteq Q$. In the best case, $\mathsf{EMCR}(Q, V)$ is equivalent to Q (In this case, we say the MCR of Q using V is equivalent to Q). The question arises now whether it is possible for Q to have no ER (which is a single TP) using V, but it has a MCR (which is a union of TPs) using V which is equivalent to Q. In other words, any single CR of Q using V is not equivalent to Q , but the union of all CRs is. We study this problem for the following cases.

4.1 The class $P^{\{/,//,[]\}}$

We first consider the case $V \in P^{\{/,//,[]\}}$, $Q \in P^{\{/,//,[]\}}$ and the rewritings of Q using V are all in $P^{\{/,//,[]\}}$. Using Theorem 1, we can easily prove the following result.

Theorem 2 *Let V, $Q \in P^{\{/,//,[]\}}$ be the view and query respectively. When there are no DTDs, if Q has a* MCR *using V which is equivalent to Q, then it has a single* CR *using V which is equivalent to Q.*

Proof Suppose $\mathsf{EMCR}(Q, V) = Q_1 \circ V \cup \ldots \cup Q_n \circ V$, where each Q_i is a CR of Q using V. Since every $Q_i \circ V$ is in $P^{\{/,//,[]\}}$, if $\mathsf{EMCR}(Q, V) = Q$, then by Theorem 1 (1) there is $i \in [1, n]$ such that $Q = Q_i \circ V$. That is, Q_i is an equivalent rewriting of Q using V. $\perp$ □

Because of Theorem 2, for queries and views in $P^{\{/,//,[]\}}$, if there is no equivalent rewriting of Q using V, then no MCR of Q using V is equivalent to Q. In other words, if we cannot find a single CR of Q using V which is equivalent to Q, then it is impossible to find a union of CRs of Q using V which is equivalent to Q. This suggests that we should always use a more efficient algorithm, such as that in [20] to find an equivalent rewriting of Q using V.

4.2 Other classes of tree patterns

Theorem 2 can be extended to the cases as stated in the next theorem.

Theorem 3 *Let V, Q be the view and query respectively. In the following cases, if Q has a* MCR *using V which is equivalent to Q, then it has a single* CR *using V which is equivalent to Q.*

(1) $Q \in P^{\{/,[],*\}}$ *and there are no leaf *-nodes in Q.*
(2) $V, Q \in \widehat{P}^{\{/,//,[],*\}}$, *and label*$(\mathtt{DN}_V) \neq *$.

Note that Theorem 3 (2) includes the case where $V, Q \in P^{\{/,//,[]\}}$.
Before proving the above theorem we prove the following lemmas first.

Lemma 4 *Let the query Q and view V be in $P^{\{/,//,[]\}}$. For every* CR $Q' \in P^{\{/,//,[],*\}}$ *of Q using V, there is a* CR $Q'' \in P^{\{/,//,[]\}}$ *such that $Q' \circ V \subseteq Q'' \circ V$.*

Proof By definition of CR, $Q' \circ V \subseteq Q$. Since $Q \in P^{\{/,//,[]\}}$, there is a CM δ from Q to $Q' \circ V$. δ partitions $N(Q)$ into two disjoint sets: $N_1 = \{v \in N(Q) \mid \delta(v) \in N(V)\}$, and $N_2 = \{v \in N(Q) \mid \delta(v) \in N(Q') - \{rt(Q')\}\}$. It is clear that for every path p in Q, either every node on p is mapped to V, or the last node on p which is mapped to V, denoted x, and the child of x on p, denoted y, satisfy the following relationship: $\delta(x)$ is on the distinguished path of V, and either (x, y) is a //-edge, or $\delta(x) = \mathtt{DN}_V$. Furthermore $\mathtt{DN}_Q$ is mapped to $\mathtt{DN}_{Q' \circ V} = \mathtt{DN}_{Q'}$.
If we denote Q'' the pattern obtained as follows:

(a) Let $rt(Q'')$ be a node labeled with *label*$(\mathtt{DN}_V)$;
(b) For every path p in Q that is not fully embedded, let the first node on p which is not mapped into V be y, and the node preceding y on p be x, then add the subtree of Q rooted at y under $rt(Q'')$, and connect $rt(Q'')$ and $rt(Q_y)$ with the same type of edge as that of (x, y).

then $Q'' \in P^{\{/,//,[]\}}$, $Q'' \circ V \subseteq Q$, and δ makes a CM from Q'' to Q', therefore, $Q' \subseteq Q''$ and $Q' \circ V \subseteq Q'' \circ V$. $\perp$ □

Lemma 5 *Let the query Q and view V be in $\widehat{P}^{\{/,//,[],*\}}$. Suppose label*$(\mathtt{DN}_V) \neq *$. *For every* CR $Q' \in P^{\{/,//,[],*\}}$ *of Q using V, there is a* CR $Q'' \in \widehat{P}^{\{/,//,[],*\}}$ *such that $Q' \circ V \subseteq Q'' \circ V$.*

The Proof of Lemma 5 is very similar to that of Lemma 4, except that to ensure Q'' is in $\widehat{P}^{\{/,//,[],*\}}$, we need the condition *label*$(\mathtt{DN}_V) \neq *$. In fact, if this condition is not satisfied, then the lemma does not hold. For example, consider the view $V = a/*[/c]$, and the query $Q = a//b$. Clearly $Q' = */*/b$ is a CR of Q using V. There is no CR $Q'' \in \widehat{P}^{\{/,//,[],*\}}$ such that $Q' \circ V \subseteq Q'' \circ V$.

Using the above lemmas, we prove Theorem 3 as follows.

Proof of Theorem 3 Recall (Section 2.3) that, in both of the cases stated in the theorem, there are CRs $Q_1, \ldots, Q_k$ of Q using V such that EMCR(Q, V)=$Q_1 \circ V \cup \ldots \cup Q_k \circ V$. Since EMCR$(Q, V) = Q$, we know $Q \subseteq Q_1 \circ V \cup \ldots \cup Q_k \circ V$. In

case (1), by Theorem 1 (2), there is some i such that $Q \subseteq Q_i \circ V$. That is, Q_i is an equivalent rewriting of Q using V. In case (2), by Lemma 5, there are $Q'_1, \ldots, Q'_k \in \widehat{P}^{\{/,//,[],*\}}$ such that $Q = Q'_1 \circ V \cup \ldots \cup Q'_k \circ V$. Clearly each $Q'_i \circ V$ is a pattern in $\widehat{P}^{\{/,//,[],*\}}$. By Theorem 1 (1), there is $i \in [1, k]$ such that $Q \subseteq Q'_i \circ V$. That is, Q'_i is an equivalent rewriting of Q using V. $\perp$ □

We point out that Theorem 3 (2) does not hold if $label(\mathtt{DN}_V) = *$ and the rewritings are allowed to be in $P^{\{/,//,[],*\}}$. For example, the query $Q = a//b$ has the following two CRs using the view $V = a/*$: $Q_1 = b$ and $Q_2 = *//b$. The union of the expansions of these rewritings is $a/b \cup a/*//b$, which is equivalent to Q. But Q is not equivalent to either a/b or $a/*//b$.

4.3 Rewritings using multiple individual views

Theorem 2 and Theorem 3 can be easily extended to the case when there are multiple views.

Theorem 4 *Let $V_1, \ldots, V_n$ be views and Q be a query. In the following cases, if $Q \subseteq \mathsf{EMCR}(Q, V_1) \cup \cdots \cup \mathsf{EMCR}(Q, V_n)$, then there exists $i \in [1, n]$ such that there is an equivalent rewriting of Q using V_i.*

(1) $Q \in P^{\{/,[],*\}}$ *and Q has no *-leaves.*
(2) $V_1, \ldots, V_n, Q \in \widehat{P}^{\{/,//,[],*\}}$, *and* $label(\mathtt{DN}_{V_i}) \neq *$ (*for* $i \in [1, n]$).

The proof of the above theorem is similar to that of Theorem 3.

4.4 Rewritings using intersections of views

So far we have only considered rewritings using individual views. However, multiple views can be combined to rewrite a query. For instance, when the views have identical root labels and identical labels for distinguished nodes (we say such views are *compatible*), the intersection of these views may be used to rewrite a query, even though there are no rewritings using any individual view. For example, the query $Q = a[b][c]/d$ does not have a CR using either $V_1 = a[b]/d$ or $V_2 = a[c]/d$, but it has a CR using $V_1 \cap V_2$, which is equivalent to Q. We will consider this case now.

We need to formally define CR and ER using $V_1 \cap \cdots \cap V_n$ first. We focus on views and queries in the class $P^{\{/,//,[]\}}$ here.

Definition 1 Let $V_1, \ldots, V_n$ be compatible views in $P^{\{/,//,[]\}}$, and Q be a query in $P^{\{/,//,[]\}}$. Suppose $V_1 \cap \cdots \cap V_n$ is not always empty. A *contained rewriting* (CR) of Q using $V_1 \cap \cdots \cap V_n$ is a TP $Q' \in P^{\{/,//,[]\}}$, such that for any XML tree t, $Q'(V_1(t) \cap \cdots \cap V_n(t)) \subseteq Q(t)$, and there is at least one t such that $V_1(t) \cap \cdots \cap V_n(t) \neq \emptyset$. Q' is said to be an *equivalent rewriting* (ER) if $Q'(V_1(t) \cap \cdots \cap V_n(t)) \supseteq Q(t)$ also holds. The *maximal contained rewriting* (MCR) of Q using the intersection is the union of all CRs of Q using the intersection.

In the presence of DTD G, the CR, MCR and ER are defined similarly, except we consider only XML trees conforming to G.

The next example shows that it is possible for a query to have different CRs using the intersection, thus the union of these CRs produces strictly more answers than any single CR.

Example 1 Consider the views $V_1 = a[x]/b$ and $V_2 = a[y]/b$, and the query $Q = a[x][y][//b/d]//b[c]$. It can be verified that $Q_1 = b[c][d]$, $Q_2 = b[d]//b[c]$, $Q_3 = b[//b/d][c]$ and $Q_4 = b[//b/d]//b[c]$ are all CRs of Q using $V_1 \cap V_2$, and none of them is contained in the others.

However, if the union of all CRs becomes equivalent to Q, then one of the CRs is equivalent to Q. In other words, Theorem 2 can be extended to rewritings using the intersection of compatible views.

Theorem 5 *Let $V_1, \ldots, V_n$ be compatible views in $P^{\{/,//,[]\}}$, and Q be a query in $P^{\{/,//,[]\}}$. If the* MCR *of Q using $V_1 \cap \cdots \cap V_n$ is equivalent to Q, then one of the* CRs *is an* ER *of Q using $V_1 \cap \cdots \cap V_n$.*

To prove the above result, we need an important property of intersection of TPs, as stated in the following lemma.

Lemma 6 [18] *Let $V_1, \ldots, V_n$ be compatible views in $P^{\{/,//,[]\}}$. If $V_1 \cap \cdots \cap V_n$ is not always empty, then there are* TPs *$V'_1, \ldots, V'_k \in P^{\{/,//,[]\}}$ such that $V_1 \cap \cdots \cap V_n$ is equivalent to $V'_1 \cup \cdots \cup V'_k$.*

We call each V'_i in the above lemma a *disjunctive component* of the intersection.

Proof of Theorem 5 By Lemma 6, there are TPs $V'_1, \ldots, V'_k \in P^{\{/,//,[]\}}$ such that $V_1 \cap \cdots \cap V_n$ is equivalent to $V'_1 \cup \cdots \cup V'_k$. Therefore, Q' is a CR of Q using $V_1 \cap \cdots \cap V_n$ if and only if it is a CR of Q using every V'_i for $i = 1, \ldots, k$. Suppose $Q_1, \ldots, Q_m$ are all of the CRs of Q using $V_1 \cap \cdots \cap V_n$. Then the MCR of Q using the intersection is equivalent to Q implies

$$Q \subseteq \bigcup_{i=1}^{m} \bigcup_{j=1}^{k} (Q_i \circ V'_j).$$

Since all TPs involved are in $P^{\{/,//,[]\}}$, by Theorem 1, we know there is an i and a j such that $Q \subseteq Q_i \circ V'_j$. Therefore, the CR Q_i is an equivalent rewriting of Q using $V_1 \cap \cdots \cap V_n$. □

Theorem 5 can also be extended to the case where rewritings using both individual views and using intersections of views are considered together. Let $S = \{V_1, \ldots, V_n\}$ be a set of compatible views. We use MCR(Q, S) (resp. EMCR(Q, S)) to denote the union of all CRs (resp. union of the expansions of all CRs) of Q using a single view in S or using the intersection of a subset of views in S. Similar to the Proof of Theorem 5, we can prove the theorem below.

Theorem 6 *Let $S = \{V_1, \ldots, V_n\}$ be compatible views in $P^{\{/,//,[]\}}$, and Q be a query in $P^{\{/,//,[]\}}$. If* EMCR(Q, S) *is equivalent to Q, then there exists a single* CR*, Q', of Q*

using either a single view, or using the intersection of some of the views in S, such that the expansion of Q' is equivalent to Q.

4.5 The presence of DTDs

Theorem 2 still holds in the presence of a non-recursive schema graph. This is because an equivalent rewriting is also a MCR, and when both Q and V are in $P^{\{/,//,[]\}}$, the MCR of Q using V under a non-recursive schema graph can be expressed as a single TP in $P^{\{/,//,[]\}}$ [6].

Theorem 2 does not hold in the presence of recursive schema graphs, as demonstrated by the next example.

Example 2 Consider the query $Q = a//b$ and the view $V = a/b$ under the recursive DTD in Figure 1(b). Q has two CRs: $Q'_1 = b$ and $Q'_2 = b//b$. The expansions of these CRs are a/b and $a/b//b$ respectively. Under the DTD, Q is equivalent to the union of a/b and $a/b//b$, but it is not equivalent to either one of them. That is, there is no single CR of Q using V which is equivalent to Q, but the union of the two CRs is equivalent to Q.

Theorem 4 does not hold in the presence of non-recursive schema graphs. For example, consider the DTD G shown in Figure 1(a). Under the DTD, if we let $Q = a//d$, $V_1 = a/b/d$ and $V_2 = a/e/d$, then $\mathsf{MCR}(Q, V_1) = V_1$ and $\mathsf{MCR}(Q, V_2) = V_2$, and $\mathsf{MCR}(Q, \{V_1, V_2\}) = V_1 \cup V_2 =_G Q$. But Q is not equivalent to either V_1 or V_2 under G.

Theorem 5 still holds in the presence of non-recursive DTDs. This is because, under a non-recursive DTD G, $V_1 \cap \cdots \cap V_n$ is equivalent to a single TP in $P^{\{/,//,[]\}}$, say V. As given in [6], the MCR of Q using V under G is contained in a single CR of Q using V under G. Therefore, the MCR of Q using $V_1 \cap \cdots \cap V_n$ under G is contained in a single CR, say Q', of Q using $V_1 \cap \cdots \cap V_n$ under G. If the MCR produces all answers to Q, namely Q is contained in the MCR, then Q' is an ER of Q.

4.6 Discussion: answerability of Q using V

In this section, we show, by example, that even if Q has no equivalent rewriting using V according to the definition given in Section 2, it is still possible to answer Q completely using V. The next example demonstrates this point.

Example 3 Consider the query Q and view V shown in Figure 5(a) and (b) respectively. Q has no equivalent rewriting using V. But given any XML tree t, we can find $Q(t)$ using the view as follows. We evaluate the query $Q_1 = x/x[e]/y$ over the subtrees in $V(t)$, and denote the results as $Q_1(V(t))$; we then evaluate $Q_2 = x/y$ over the subtrees in $V(t)$, and obtaining a set denoted as $Q_2(V(t))$. Finally, we take the intersection of $Q_1(V(t))$ and $Q_2(V(t))$. It can be verified that $Q(t) = Q_1(V(t)) \cap Q_2(V(t))$.

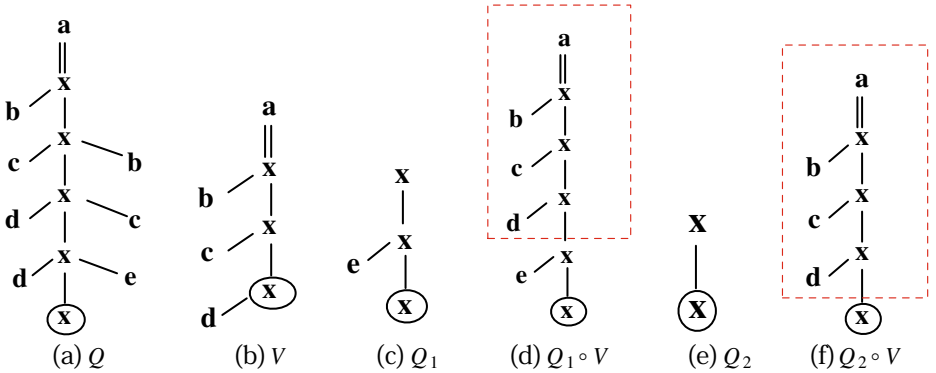


Figure 5 Q has no equivalent rewriting using V according to conventional definition, but Q can be fully answered using V: $Q_1 \circ V \cap Q_2 \circ V = Q$.

5 Redundant views

In this section we assume there are multiple views $V_1, \ldots, V_n$ in $\widehat{P}^{\{/,//,[],*\}}$. We now ask the question when a subset of views $V_{i_1}, \ldots, V_{i_k}$ are redundant in the sense that for every query Q, the MCRs of Q using $V_{i_1}, \ldots, V_{i_k}$ are all contained in the union of the MCRs of Q using the other views. Formally we have

Definition 2 Let $V_1, \ldots, V_n \in \widehat{P}^{\{/,//,[],*\}}$ be views and $k < n$. If for every TP $Q \in \widehat{P}^{\{/,//,[],*\}}$,

$$\bigcup_{i=1}^{k} \mathsf{EMCR}(Q, V_i) \subseteq \bigcup_{j=k+1}^{n} \mathsf{EMCR}(Q, V_j)$$

then we say the views $V_1, \ldots, V_k$ are *redundant*.

Intuitively, when considering CRs the redundant views can be ignored because all answers returned by CRs using the redundant views can be returned by CRs using other views.

One might wonder how the redundant views problem is related to the view containment problem $\cup_{i=1}^{k} V_i \subseteq \cup_{j=k+1}^{n} V_j$. As we show in the next example, the condition $\cup_{i=1}^{k} V_i \subseteq \cup_{j=k+1}^{n} V_j$ is neither sufficient nor necessary for $V_1, \ldots, V_k$ to be redundant.

Example 4 (1) Let $V_1 = a[b]/c$ and $V_2 = a/c$. Clearly $V_1 \subseteq V_2$. But V_1 is not redundant, because the query $Q = a[b]/c/d$ has a CR using V_1, but it does not have a CR using V_2. (2) Now let $V_1 = a/x/x$ and $V_2 = a/x$. It is easy to verify $V_1 \nsubseteq V_2$, but V_1 is redundant.

We now provide the following sufficient and necessary condition for redundant views.

Theorem 7 *Given* $V_1, \ldots, V_n \in P^{\{/,//,[],*\}}$, $V_1, \ldots, V_k$ $(k < n)$ *are redundant iff for every* $i \in [1, k]$, *the union of the expansions of the* CRs *of* V_i *using the views* $V_{k+1}, \ldots, V_n$ *is equivalent to* V_i, *that is,*

$$V_i = \bigcup_{j=k+1}^{n} \mathsf{EMCR}(V_i, V_j).$$

Proof (only if) we only need to consider the query $Q = V_i$ for $i \in [1, k]$. Clearly there is an equivalent rewriting of V_i using itself. Therefore $\bigcup_{j=1}^{k} \mathsf{EMCR}(V_i, V_j) = V_i$ for $i \in [1, k]$. By definition, $V_1, \ldots, V_k$ are redundant implies that

$$\bigcup_{j=1}^{k} \mathsf{EMCR}(V_i, V_j) \subseteq \bigcup_{j=k+1}^{n} \mathsf{EMCR}(V_i, V_j).$$

Thus

$$V_i \subseteq \bigcup_{j=k+1}^{n} \mathsf{EMCR}(V_i, V_j)$$

for all $i \in [1, k]$. Since $\mathsf{EMCR}(V_i, V_j) \subseteq V_i$ for all $j \in [k+1, n]$, we know

$$V_i = \bigcup_{j=k+1}^{n} \mathsf{EMCR}(V_i, V_j).$$

(if) Suppose for every $i \in [1, k]$,

$$V_i = \bigcup_{j=k+1}^{n} \mathsf{EMCR}(V_i, V_j).$$

Suppose $\mathsf{EMCR}(V_i, V_j) = Q_{j,1} \circ V_j \cup \ldots \cup Q_{j,m_j} \circ V_j$. Then

$$V_i = \bigcup_{j=k+1}^{n} \bigcup_{s=1}^{m_j} (Q_{j,s} \circ V_j).$$

For any TP Q, if Q' is a CR of Q using V_i, then $Q' \circ V_i \subseteq Q$. Hence

$$Q' \circ \bigcup_{j=k+1}^{n} \bigcup_{s=1}^{m_j} (Q_{j,s} \circ V_j) \subseteq Q$$

i.e.,

$$\bigcup_{j=k+1}^{n} \bigcup_{s=1}^{m_j} ((Q' \circ Q_{j,s}) \circ V_j) \subseteq Q$$

Thus every $Q' \circ Q_{j,s}$ $(s = 1, \ldots, m_j)$, if not empty, is a CR of Q using V_j. Therefore,

$$Q' \circ V_i \subseteq \bigcup_{j=k+1}^{n} \mathsf{EMCR}(Q, V_j)$$

Since Q' is an arbitrary CR of Q using V_i, we know

$$\bigcup_{i=1}^{k} \mathsf{EMCR}(Q, V_i) \subseteq \bigcup_{j=k+1}^{n} \mathsf{EMCR}(Q, V_j).$$

By definition, $V_1, \ldots, V_k$ are redundant. $\perp$ □

If the views $V_1, \ldots, V_k$ are in $P^{\{/,[],*\}}$ and they have no *-leaves, or if $V_1, \ldots, V_n$ are in $\widehat{P}^{\{/,//,[],*\}}$ and the distinguished nodes of $V_{k+1}, \ldots, V_n$ are not labeled *, then by Theorem 4, $V_i = \bigcup_{j=k+1}^{n} \mathsf{EMCR}(V_i, V_j)$ iff there is $j \in [k+1, n]$ such that V_i has an equivalent rewriting using V_j. This leads to the following corollary.

Corollary 1 *Given views $V_1, \ldots, V_n$, In the following cases, $V_1, \ldots, V_k$ $(k < n)$ are redundant iff for every $i \in [1, k]$, there exists $j \in [k+1, n]$ such that V_i has an equivalent rewriting using V_j.*

(1) *$V_1, \ldots, V_k$ are in $P^{\{/,[],*\}}$ and they have no *-leaves.*
(2) *$V_1, \ldots, V_n$ are in $\widehat{P}^{\{/,//,[],*\}}$, and the distinguished nodes of $V_{k+1}, \ldots, V_n$ are not labeled *.*

Note that case (2) above includes the case where $V_1, \ldots, V_n$ are in $P^{\{/,//,[]\}}$.

Theorem 7 still holds when there is a schema graph, and the proof is similar. However, when a schema graph exists, the condition in Corollary 1 is still sufficient but no longer necessary. The proof of sufficiency is similar to the case when there are no schema graphs. The non-necessity is shown by the DTD in Figure 1(a), and the views $V_1 = a/b$, $V_2 = a/e$, and $V_3 = a//d$. Clearly V_3 is redundant under the DTD, but it does not have an equivalent rewriting using either V_1 or V_2.

A special case of Corollary 1 is when all the views $V_1, \ldots, V_n$ are in $P^{\{/,//,[]\}}$, and they are *compatible*, that is, their distinguished nodes have identical labels. In this case, if V_i is redundant, then by Corollary 1, there is $j \neq i$ such that V_i has an equivalent rewriting using V_j. That is, there is a TP Q' such that $V_i = Q' \circ V_j$. If the distinguished path of V_i does not have repeating labels, then we know the root of Q' must be the same as the distinguished node of Q'. Therefore $V_i \subseteq V_j$. This proves the following corollary.

Corollary 2 *Let $V_1, \ldots, V_n$ be compatible views in $P^{\{/,//,[]\}}$. If $V_1, \ldots, V_k$ $(k < n)$ are redundant, then for every $i \in [1, k]$ such that the distinguished path of V_i has no repeating labels, there exists $j \in [k+1, n]$ such that $V_i \subseteq V_j$.*

The condition that the distinguished path of V_i has no repeating labels is important in the above corollary. If it is not satisfied, the corollary does not hold. This is easily seen in Example 4 (2).

Identifying redundant views Corollary 1 provides a means to find the redundant views. To see whether V_i is redundant we only need to check whether there exists V_j such that V_i has an equivalent rewriting using V_j. To do so we can use the algorithm in [20].

5.1 Redundant views when intersections of views are used for rewriting

We now re-examine the redundant views problem, taking into consideration rewritings using intersections of views as well as individual views.

Let $S = \{V_1, \ldots, V_n\}$ be a set of compatible views. We use $\mathsf{EMCR}(Q, S)$ to denote the union of expansions of all CRs of Q using any single view in S or using the intersection of any subset of views in S. The new meaning of redundant views is as follows.

Definition 3 Let $S = \{V_1, \ldots, V_n\}$ be a set of compatible views and S' be a proper subset of S. We say that S' is *strongly redundant* if for every query Q, $\mathsf{EMCR}(Q, S) \subseteq \mathsf{EMCR}(Q, S - S')$.

The following theorem provides a necessary condition for S' to be strongly redundant.

Theorem 8 *Let $S = \{V_1, \ldots, V_n\}$ be a set of compatible views in $P^{\{/,//,[]\}}$ and S' be a proper subset of S. If S' is strongly redundant, then for every view $V \in S'$, $V = \mathsf{EMCR}(V, S - S')$.*

Proof We prove the theorem by contradiction. Suppose there is $V \in S$ such that $V \neq \mathsf{EMCR}(V, \mathcal{S} - \mathcal{S}')$. That is, $V \notin \mathsf{EMCR}(V, \mathcal{S} - \mathcal{S}')$. Consider the query $Q = V$. Q has an equivalent rewriting using S, whose expansion is V itself. Thus $\mathsf{EMCR}(Q, S) = V$. Therefore $\mathsf{EMCR}(Q, S) \notin \mathsf{EMCR}(Q, \mathcal{S} - \mathcal{S}')$. This contradicts the assumption that S' is strongly redundant. $\perp$ □

However, the condition that $\forall V \in S, V = \mathsf{EMCR}(V, S - S')$ is generally not sufficient for S' to be strongly redundant, as shown in the following example.

Example 5 Let V be the view shown in Figure 5b, V_1 be the pattern shown in Figure 5(d), and V_2 be the pattern shown in Figure 5f. Let $S = \{V, V_1, V_2\}$ and $S' = \{V_1, V_2\}$. As shown in Figure 5, $V_1 = Q_1 \circ V$ and $V_2 = Q_2 \circ V$. However, S' is not strongly redundant because there is Q (as shown in Figure 5(a)) which has an equivalent rewriting using $V_1 \cap V_2$, but no equivalent rewriting using V.

The next theorem provides a sufficient condition for S' to be strongly redundant.

Theorem 9 *Let $S = \{V_1, \ldots, V_n\}$ be a set of compatible views in $P^{\{/,//,[]\}}$ and S' be a proper subset of S. If for every view $V \in S'$, $V = \mathsf{EMCR}(V, S - S')$, and for every intersection I involving views in S' each disjunctive component V' in I satisfies $V' = \mathsf{EMCR}(V', S - S')$, then S' is strongly redundant.*

Proof Let Q be any TP in $P^{\{/,//,[]\}}$. Suppose Q has a CR Q' using the intersection I of some views (a single view is treated as a special intersection). Suppose $I = V'_1 \cup \ldots \cup V'_k$, then Q' is a CR of Q using V'_i for all $i \in [1, k]$. By assumption,

every V_i' satisfies $V_i' = \mathsf{EMCR}(V_i', S - S')$. By Theorem 6, there is intersection I_i of some views in $S - S'$ such that V_i' has an equivalent rewriting using I_i. Let $V_i' = Q_i \circ I_i$. Then $Q' \circ (Q_i \circ I_i) \subseteq Q$ (hence $(Q' \circ Q_i) \circ I_i \subseteq Q$). Thus $Q' \circ Q_i$ is a CR of Q using I_i, and $Q' \circ I = Q' \circ (V_1' \cup \ldots \cup V_k') = \bigcup_{i=1}^{k} (Q' \circ Q_i) \circ I_i$. Therefore, $\mathsf{EMCR}(Q, S) \subseteq \mathsf{EMCR}(Q, S - S')$. By definition, S' is strongly redundant. $\perp$ □

6 Related work

When a tree pattern is viewed as a constraint over XML trees, the independence of containing patterns property can be regarded as a special case of the *independence of negative constraints (INC)* property: given constraints $C, C_1, \ldots, C_n$, C implies $C_1 \vee \cdots \vee C_n$ iff C implies some C_i. The INC property was first studied in [7] and since then found to hold for many classes of constraints. For works on tree pattern query rewriting using views, besides the papers [6, 9, 20] discussed in Section 1, several other papers have dealt with the problem. In particular, [18] considered rewritings using different combinations of multiple views, one of them is intersection. Tajima and Fukui [14] studied the problem of query answerability using views for general `XPath` queries (that may involve negation, and disjunction), that is, given Q and $V_1, \cdots, V_n$, whether there are $Q_1, \cdots, Q_n$ such that $Q_1 \circ V_1 \cup \cdots \cup Q_n \circ V_n = Q$. As shown in Section 4 (Theorem 4), when Q and $V_1, \cdots, V_n$ are restricted to some classes of tree patterns, the problem is significantly simplified because no unions of tree patterns need to be considered, hence the simple algorithm in [20] can be applied. Tang and Zhou [15] defined *correct rewritings* of TPs, using a single view, for tree patterns with multiple output nodes. The rewriting is essentially a mapping from the output nodes of Q to the output nodes of V under which $V \subseteq Q$. When restricted to a single output node for each pattern or view, the mapping is unique, and the existence of a correct rewriting simply means $V \subseteq Q$. Balmin et al. [3] addressed the problem of answering `XPath` queries using a single *materialized* view where, for the view, a combination of node references, typed data values, and full paths may be stored. However, the way in which a query is answered using the view is different: one can follow node references to go to the original document, so the original XML tree cannot be discarded. Tang et al. [16] presented an algorithm for *equivalently answering* `XPath` queries using multiple materialized views based on the assumption that the Dewey codes are stored in the materialized views so that the common ancestors of nodes in different views can be found. The paper also studied the view selection problem, which is to, given a query, find a minimal subset of views to equivalently answer that query. This is apparently related to the redundant views problem since any redundant view should not be selected. However, since the way of utilizing the view in [16] is different from ours, the redundant views are also different. Arion et al. [2] studied a different type of equivalent rewriting using multiple views in the presence of *structural summaries* and integrity constraints: the answer sets of the views are nodes rather than subtrees, and the answers to the new query are obtained by combining answers to the views through a number of algebraic operations. There have also been works on rewriting XQuery queries using views [4, 13, 21]. In relational query rewriting using views, the redundant views problem was studied in [8], which showed similar properties for views which are redundant: a view is redundant if and only if

it has an equivalent rewriting using the other views. We are not aware of any work on the form of equivalent rewritings, neither for XPath rewritings nor for relational rewritings.

7 Conclusion

We showed that some tree patterns observe the independence of containing patterns property. Based on which, we showed that for some classes of tree patterns, the equivalent rewriting using views can be expressed as a single tree pattern rather than the union of multiple tree patterns. We also identified necessary and sufficient conditions for a subset of views to be redundant. In doing this, we considered different scenarios: the absence or presence of DTDs, rewritings using multiple single views, and rewritings using the intersection of views.

Acknowledgements This work is partially supported by grant from the Research Grant Council of the Hong Kong Special Administrative Region, China (CUHK418205), and Griffith University New Researcher's Grant (GUNRG36621).

References

1. Amer-Yahia, S., Cho, S., Lakshmanan, L.V.S., Srivastava, D.: Minimization of tree pattern queries. In: SIGMOD, pp. 497–508. ACM, New York, USA (2001)
2. Arion, A., Benzaken, V., Manolescu, I., Papakonstantinou, Y.: Structured materialized views for XML queries. In: VLDB, pp. 87–98. ACM, New York, USA (2007)
3. Balmin, A., Özcan, F., Beyer, K.S., Cochrane, R., Pirahesh, H.: A framework for using materialized XPath views in XML query processing. In: VLDB, pp. 60–71. Morgan Kaufmann, San Francisco, USA (2004)
4. Deutsch, A., Tannen, V.: Reformulation of XML queries and constraints. In: ICDT, pp. 225–241. Lecture Notes in Computer Science 2572, Springer, Germany (2003)
5. Halevy, A.Y.: Answering queries using views: a survey. VLDB J. **10**(4), 270–294. Springer Berlin, Germany (2001)
6. Lakshmanan, L.V.S., Wang, H., Zhao, Z.J.: Answering tree pattern queries using views. In: VLDB, pp. 571–582. ACM, New York, USA (2006)
7. Lassez, J.-L., McAloon, K.: Independence of negative constraints. In: TAPSOFT, vol. 1, pp. 19–27. Lecture Notes in Computer Science 352, Springer-Verlag, Germany (1989)
8. Li, C., Bawa, M., Ullman, J.D.: Minimizing view sets without losing query-answering power. In: ICDT, pp. 99–113. Lecture Notes in Computer Science 1973, Springer, Germany (2001)
9. Mandhani, B., Suciu, D.: Query caching and view selection for XML databases. In: VLDB, pp. 469–480. ACM, New York, USA (2005)
10. Miklau, G., Suciu, D.: Containment and equivalence for an XPath fragment. In: PODS, pp. 65–76. ACM, New York, USA (2002)
11. Miklau, G., Suciu, D.: Containment and equivalence for a fragment of XPath. J. ACM **51**(1), 2–45. ACM, New York, USA (2004)
12. Nash, A., Segoufin, L., Vianu, V.: Determinacy and rewriting of conjunctive queries using views: a progress report. In: ICDT, pp. 59–73. Lecture Notes in Computer Science 4353, Springer, Germany (2007)
13. Onose, N., Deutsch, A., Papakonstantinou, Y., Curtmola, E.: Rewriting nested XML queries using nested views. In: SIGMOD, pp. 443–454. ACM, New York, USA (2006)
14. Tajima, K., Fukui, Y.: Answering XPath queries over networks by sending minimal views. In: VLDB, pp. 48–59. Morgan Kaufmann, San Francisco, USA (2004)
15. Tang, J., Zhou, S.: A theoretic framework for answering XPath queries using views. In: XSym, pp. 18–33. Lecture Notes in Computer Science 3671, Springer, Germany (2005)
16. Tang, N., Yu, J.X., Özsu, M.T., Choi, B., Wong, K.-F.: Multiple materialized view selection for XPath query rewriting. In: ICDE, pp. 873–882. IEEE, USA (2008)

17. Wang, J., Topor, R.W., Maher, M.J.: Rewriting union queries using views. Constraints **10**(3), 219–251. Springer, The Netherlands (2005)
18. Wang, J., Xu, J.Y.: Tree pattern rewriting using multiple views. In: DEXA, pp. 493–507. Lecture Notes in Computer Science 5181, Springer, Germany (2008)
19. Wang, J., Xu, J.Y., Liu, C.: Contained XPath rewriting using views revisited. In: WISE, pp. 410–425. Lecture Notes in Computer Science 5175, Springer, Germany (2008)
20. Xu, W., Özsoyoglu, Z.M.: Rewriting XPath queries using materialized views. In: VLDB, pp. 121–132. ACM, New York, USA (2005)
21. Yu, C., Popa, L.: Constraint-based XML query rewriting for data integration. In: SIGMOD, pp. 371–382. ACM, New York, USA (2004)