

On Tree Pattern Query Rewriting Using Views^{*}

Junhu Wang^{1**}, Jeffrey Xu Yu², and Chengfei Liu³

¹ Griffith University, Gold Coast, Australia
J.Wang@griffith.edu.au

² Chinese University of Hong Kong, Hong Kong, China
yu@se.cuhk.edu.hk

³ Swinburne University of Technology, Melbourne, Australia
cliu@ict.swin.edu.au

Abstract. We study and present our findings on two closely related problems on XPATH rewriting using views when both the view and the query are tree patterns involving $/, //$ and $[]$. 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 answer a query completely using the views, we should use more efficient methods, such as the polynomial algorithm of [12], to find the equivalent rewriting, rather than try to find the union of all contained rewritings and test its equivalence to Q . Second, given a set \mathcal{V} of views, we want to know under what conditions a subset \mathcal{V}' of the views are redundant in the sense that for *any query* Q , the contained rewritings of Q using the views in \mathcal{V}' are contained in those using the views in $\mathcal{V} - \mathcal{V}'$. 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. We provide necessary and sufficient conditions for the second problem, based on answers to the first problem. When the views produce comparable answers, we extend our findings to include the case where the intersection of views, in addition to the individual views, are used in the 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

^{*} The work described in this paper was supported by grant from the Research Grant Council of the Hong Kong Special Administrative Region, China (CUHK418205).

^{**} Work done while visiting the Chinese University of Hong Kong.

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 [5, 9].

More recently rewriting XML queries using XML views has attracted attention because of the rising importance of XML data [12, 7, 6, 10]. 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, 8]. Among previous work on rewriting XPATH queries using views, Xu et al [12] studied equivalent rewritings for several different classes of tree patterns, while Mandhani and Suciu [7] 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 $P\{/, //, []\}$ [8]), both in the absence and presence of non-recursive schema graphs - a restricted form of DTDs.

In this paper, we study two closely related problems on XPATH rewritings using views, for queries and views in $P\{/, //, []\}$. 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 completely answer a query using the view, we can use more efficient methods such as the polynomial time algorithm of [12] to find the equivalent rewriting, rather than try to find the union of all contained rewritings and test its equivalence to Q . The second problem is what we call the *redundant views* problem. Given a set \mathcal{V} of views, we want to know under what conditions a subset \mathcal{V}' of the views are redundant in the sense that for *any query* Q , the contained rewritings of Q using the views in \mathcal{V}' are contained in those using the views in $\mathcal{V} - \mathcal{V}'$. 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 identify a necessary and sufficient condition for \mathcal{V}' to be redundant, based on our results to the first problem. In the case that all the views in \mathcal{V} produce comparable answers, we show it is possible to use the intersection of views to rewrite a query, when there are no rewritings using any single view. We extend our results on the two problems to include this scenario.

Our main contributions are:

- 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, and this is true even in the presence of non-recursive, non-disjunctive DTDs. But the result no longer holds for larger classes of tree patterns or if there is a recursive DTD.
- When multiple views exist, we provide a necessary and sufficient condition for identifying redundant views.

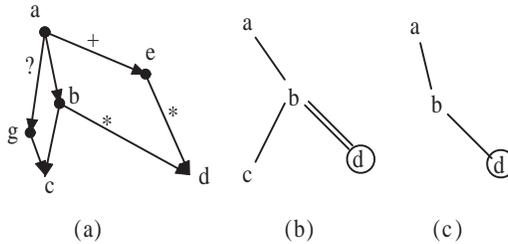


Fig. 1. Example DTD and TPs consistent with the DTD

- We extend tree pattern rewriting using single views to that using intersections of views, and extend the above results to include such rewritings.

The rest of the paper is organized as follows. Section 2 provides the terminology and notations. Section 3 presents our result on the form of equivalent rewritings. Based on this result, Section 4 proves a necessary and sufficient condition under which some views are redundant. Section 5 studies the two problems when rewriting using intersection of views is included. 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*) as follows. (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 1. (3) There is a unique node, called the root, which has an incoming degree of zero. 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. A schema graph (DTD) example is shown in Figure 1 (a).

An XML document is a node-labeled, unordered tree⁴. 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 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

⁴ Order and attributes in XML documents are disregarded in this paper.

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.

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

Here, $\tau \in \Sigma$, 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 [8]. A tree pattern has a tree representation. Figures 1 (b) and (c) show two TPs. They correspond to XPATH expressions $a/b[c]//d$ and $a/b/d$, respectively. Here, single and double lines represent child-axis ($/$), called $/$ -edge, and descendant-axis ($//$), called $//$ -edge, respectively. A circle denotes the output of P (called the distinguished node of P). Below, given a TP P , like [6], we use d_P to denote the distinguished node.

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) **label-preserving**, i.e., $\forall v \in N(P), label(v) = label(\delta(v))$, (2) **root-preserving**, i.e., $\delta(rt(P)) = rt(t)$, 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 $\delta(d_P)$, denoted $sub_{\delta(d_P)}^t$, 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 :

$$P(t) = \{sub_{\delta(d_P)}^t \mid \delta \text{ is a matching of } P \text{ in } t\} \quad (1)$$

Let T be a set of XML trees. We use $P(T)$ to denote the union of answer sets of P 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 consistent with G , that is, there is $t \in T_G$ such that $P(t) \neq \emptyset$.

2.2 TP rewriting using views

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 $P, Q \in P^{\{/, //, []\}}$, $P \subseteq Q$ iff there is a containment mapping from Q to P [1]. Recall: 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 $h(d_Q) = d_P$.

Contained Rewriting and Maximal Contained Rewriting: A *view* is an existing 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, 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 . By

definition the root of Q' must be labeled with $label(d_V)$. We use $V \oplus Q'$ to represent the *expansion* of Q' , which is the TP obtained by merging $rt(Q')$ and d_V (see Figure 2). Clearly $V \oplus Q' \subseteq Q$. If $V \oplus Q' \supseteq Q$ also holds, Q' is called an *equivalent rewriting* (ER) of Q using V [12]. The *maximal contained rewriting* (MCR), 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 the CRs in $MCR(Q, V)$.

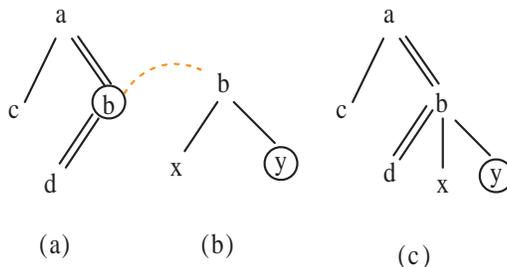


Fig. 2. (a) the view V , (b) the CR Q' , (c) the expansion $V \oplus Q'$

In the presence of DTDs: The concepts of TP containment, CR, and MCR extend naturally to the case where a DTD is available. 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 $t \in T_G$, $P(t) \subseteq Q(t)$. A *contained rewriting* (resp. *equivalent rewriting*) of Q using view V under 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 $t \in T_G$, $Q'(V(t)) \neq \emptyset$. A MCR of Q using V under G is the union of all CRs of Q using V under G .

3 Form of equivalent rewritings

Let $V \in P\{/, //, []\}$ be the view, and $Q \in P\{/, //, []\}$ be the query. By definition, $EMCR(Q, V) \subseteq Q$. In the best case, $EMCR(Q, V)$ may be 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.

The answer to the above question is “no”, and this is even true in the presence of non-recursive DTDs. We prove this result below.

Theorem 1. *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 .*

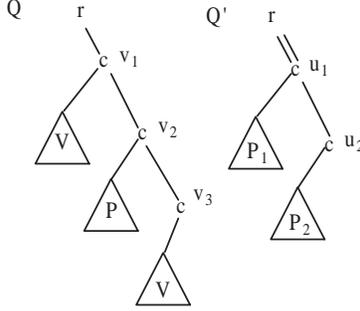


Fig. 3. Q and Q' as in [8]

The proof of Theorem 1 follows directly from an interesting property of TPs in $P^{\{/,//,\emptyset\}}$ stated in the theorem below.

Theorem 2. *For tree patterns $P, P_1, \dots, P_n \in P^{\{/,//,\emptyset\}}$, if $P \subseteq \bigcup_{i=1}^n P_i$, then there exists $i \in [1, n]$ such that $P \subseteq P_i$.*

To prove the above theorem, we need the concept of boolean tree patterns. A *boolean pattern* [8] is a pattern with no output 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 $P_1(t) = \text{TRUE}$ implies $P_2(t) = \text{TRUE}$ for any XML tree t . If P_1 and P_2 are in $P^{\{/,//,\emptyset\}}$, 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) [8]. 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 1. *For boolean patterns $P, P_1, \dots, P_n \in P^{\{/,//,\emptyset\}}$, if $P \subseteq \bigcup_i P_i$, then there exists $i \in [1, n]$ such that $P \subseteq P_i$.*

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

Using Lemma 1 of [8], 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 . Furthermore, V does not have $*$ -nodes (see [8] for how to construct V). Since $P \subseteq P_1 \cup P_2$ implies $Q \subseteq Q'$, and there is no $*$ -node in Q' , as given in [8], 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 later case there will be a homomorphism from P_1 to P . Therefore, either $P \subseteq P_1$ or $P \subseteq P_2$.

We are now ready to prove Theorem 2.

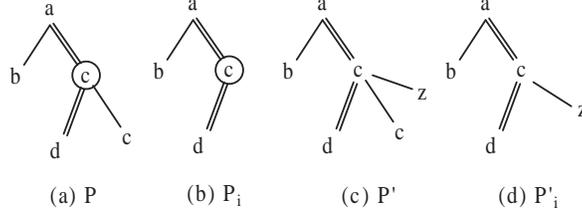


Fig. 4. Tree patterns P , P_i and the boolean patterns P' and P'_i

Proof (proof of Theorem 2).

We denote by P' , P'_i ($i \in [1, n]$) the boolean patterns obtained from P, P_i by attaching a child node labelled 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 some P_i in t and $f(d_{P_i}) = h(d_P)$. 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 1, 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 $P \subseteq P_i$.

Because of Theorem 1, 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, if we want to use the results of a view to **completely** answer query Q (for all XML trees), we should always use a more efficient algorithm, such as Lemma 4.8 of [12], to find an equivalent rewriting of Q using V . If we cannot find such an equivalent rewriting, then it is impossible to answer Q completely using the results of V .

Theorem 1 can be easily extended to the case when there are multiple views.

Theorem 3. *Let $V_1, \dots, V_n \in P^{\{/, //, \square\}}$ be views and $Q \in P^{\{/, //, \square\}}$ be a query. If $Q \subseteq \text{EMCR}(Q, V_1) \cup \dots \cup \text{EMCR}(Q, V_n)$, then there exists $i \in [1, n]$ such that there is an equivalent rewriting of Q using V_i .*

Theorem 1 (but not Theorem 3) is still true in the presence of non-recursive DTDs. This is because an equivalent rewriting is also a MCR, and when both Q and V are in $P^{\{/, //, \square\}}$, the MCR of Q using V under a non-recursive DTD is a single TP [6]. This is a bit surprising in view of the fact that Theorem 2 is not true in the presence of DTDs. For example, consider the DTD in Figure 5 (a). Under the DTD, $a//d \subseteq a/b/d \cup a/c/d$. But $a//d$ is contained in neither $a/b/d$

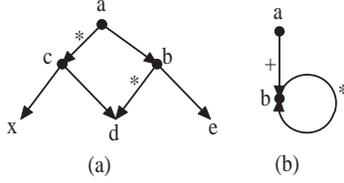


Fig. 5. Example schema graphs

nor $a/c/d$. If we let $Q = a//d$, $V_1 = a/b/d$ and $V_2 = a/c/d$, then this serves as an example to show Theorem 3 is not true in the presence of DTDs.

Theorem 1 does not hold in the presence of recursive DTDs, as demonstrated by the next example.

Example 1. Consider the query $Q = a//b$ and the view $V = a/b$ under the recursive DTD in Figure 5 (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 .

We point out that Theorem 1 is not true if V and the rewritings have the wildcard $*$ (see details in the full version of this paper).

4 Redundant views

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

Definition 1. Let $V_1, \dots, V_n \in P\{/,//,\emptyset\}$ be views and $k < n$. If for every TP $Q \in P\{/,//,\emptyset\}$,

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

then we say the views V_1, \dots, 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 $\bigcup_{i=1}^k V_i \subseteq \bigcup_{j=k+1}^n V_j$. As we show in the next example, the condition $\bigcup_{i=1}^k V_i \subseteq \bigcup_{j=k+1}^n V_j$ is neither sufficient nor necessary for V_1, \dots, V_k to be redundant.

Example 2. (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 $V_1 \not\subseteq V_2$, but V_1 is redundant.

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

Theorem 4. *Given $V_1, \dots, V_n \in P^{\{/, //, \emptyset\}}$, V_1, \dots, 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 .*

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 $\bigcup_{j=1}^k \text{EMCR}(V_i, V_j) = V_i$ for $i \in [1, k]$. By definition, V_1, \dots, V_k are redundant implies that

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

Thus

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

for all $i \in [1, k]$. By Theorem 3, there is an equivalent rewriting of V_i using some V_j ($j \in [k+1, n]$).

(if) Suppose for every $i \in [1, k]$, there exists $j \in [k+1, n]$ such that V_i has an equivalent rewriting using V_j . Let V'_i be the equivalent rewriting of V_i using V_j . By definition V_i is equivalent to $V_j \oplus V'_i$. For any TP $Q \in P^{\{/, //, \emptyset\}}$, if Q' is a CR of Q using V_i , then $V_i \oplus Q' \subseteq Q$. Hence $V_j \oplus (V'_i \oplus Q') = (V_j \oplus V'_i) \oplus Q'$ is contained in Q . Therefore $V'_i \oplus Q'$ is a CR of Q using V_j , and the expansion of this CR is equivalent to the expansion of Q' . Therefore, $\text{EMCR}(Q, V_i) \subseteq \text{EMCR}(Q, V_j)$. This implies

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

When DTDs exist, the condition in Theorem 4 is still sufficient but not necessary. The proof of sufficiency is similar to the case when there are no DTDs. The non-necessity is shown by the DTD in Figure 5 (a), and the views $V_1 = a/b$, $V_2 = a/c$, 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 is when all the views V_1, \dots, V_n are *compatible*, that is, their distinguished nodes have identical labels. In this case, if V_i is redundant, then by Theorem 4, 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 $V_i = V_j \oplus Q'$. If the distinguished path of V_i does not have repeating labels, then we know the root of Q' is the distinguished node of Q' . Therefore $V_i \subseteq V_j$. This proves the following corollary of Theorem 4.

Corollary 1. *Let V_1, \dots, V_n be compatible views. If V_1, \dots, 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 $V_i \subseteq V_j$.*

Note that in Definition 1, and so in Theorem 4, we haven't taken into account the possibility of rewriting a query using multiple views. We are only considering rewritings using single views. 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 in the next section.

Finding redundant views Theorem 4 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 Lemma 4.8 of [12].

5 TP rewriting using intersection of views

As noted in the previous section, given compatible views V_1, \dots, V_n , it is possible for a query Q to have no CRs using any single view but it has a CR using the intersection of these views. In this section we investigate the form of equivalent rewriting problem and redundant views problem, taking into consideration of this possibility. We need to formally define CR and ER using $V_1 \cap \dots \cap V_n$ first.

Definition 2. *Let V_1, \dots, V_n be compatible views in $P^{\{\cdot//\cdot, \cdot\}}$, and Q be a query in $P^{\{\cdot//\cdot, \cdot\}}$. Suppose $V_1 \cap \dots \cap V_n$ is not always empty. A contained rewriting (CR) of Q using $V_1 \cap \dots \cap V_n$ is a TP $Q' \in P^{\{\cdot//\cdot, \cdot\}}$, such that for any XML tree t , $Q'(V_1(t) \cap \dots \cap V_n(t)) \subseteq Q(t)$. Q' is said to be an equivalent rewriting (ER) if $Q'(V_1(t) \cap \dots \cap V_n(t)) \supseteq Q(t)$ 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 3. 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 another.

However, if the union of all CRs becomes equivalent to Q , then one of the CRs is equivalent to Q . In other words, Theorem 1 can be extended to rewritings using the intersection of compatible views.

Theorem 5. *Let V_1, \dots, V_n be compatible views in $P^{\{\cdot//\cdot, \cdot\}}$, and Q be a query in $P^{\{\cdot//\cdot, \cdot\}}$. If the MCR of Q using $V_1 \cap \dots \cap V_n$ is equivalent to Q , that is, it produces all answers to Q when evaluated on $V_1(t) \cap \dots \cap V_n(t)$ for all t , then one of the CRs is an ER of Q using $V_1 \cap \dots \cap V_n$.*

Proof. The proof uses an important property of intersection of TPs: there are TPs $V'_1, \dots, V'_k \in P\{\cdot, \cdot, \cdot, \cdot\}$ such that $V_1 \cap \dots \cap V_n$ is equivalent to $V'_1 \cup \dots \cup V'_k$ (For the proof of this property see the full version of this paper. We omit it because of page limit). Therefore, Q' is a CR of Q using $V_1 \cap \dots \cap V_n$ if and only if it is a CR of Q using every V'_i for $i = 1, \dots, k$. Suppose Q_1, \dots, Q_m are all of the CRs of Q using $V_1 \cap \dots \cap V_n$. Then the MCR of Q using the intersection is equivalent to Q implies

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

Since all TPs involved are in $P\{\cdot, \cdot, \cdot, \cdot\}$, by Theorem 2, we know there is an i and a j such that $Q \subseteq V'_j \oplus Q_i$. Therefore, the CR Q_i is an equivalent rewriting of Q using $V_1 \cap \dots \cap V_n$.

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

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

Theorem 6. *If $\text{MCR}(Q, \mathcal{V})$ 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 \mathcal{V} , such that Q' is equivalent to Q .*

Next we re-examine the redundant views problem, taking into consideration of rewritings using intersections of views as well as individual views. The new meaning of redundant views is as follows.

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

The following theorem can be proved in a way similar to the proof of Theorem 4.

Theorem 7. *Let $\mathcal{V} = \{V_1, \dots, V_n\}$ be a set of compatible views and \mathcal{V}' be a proper subset of \mathcal{V} . \mathcal{V}' is strongly redundant iff for every view $V \in \mathcal{V}'$, $V = \text{MCR}(V, \mathcal{V} - \mathcal{V}')$.*

6 Other related work

Besides the papers on rewriting XPath queries using views [6, 12, 7] discussed in Section 1, our work is closely related to [8], which studies the containment problem of different classes of TPs and provides the basis for TP rewriting. For XQuery rewriting using views, [10] deals with equivalent rewriting of XQuery queries using XQuery views, and [3] translates XQuery rewriting to relational rewriting problems. Query rewriting using views has also been extensively studied in the relational database context, see [5] for a survey and [11, 2, 4, 9] for more recent developments.

7 Conclusion

We presented some interesting results on the form of equivalent rewritings, and identified necessary and sufficient conditions for a subset of views to be redundant, in two scenarios. The first is when the rewritings are using a single view, and second is when the rewriting can use the intersection of compatible views in addition to individual views. These results are of theoretical and practical importance to TP query processing.

References

1. S. Amer-Yahia, S. Cho, L. V. S. Lakshmanan, and D. Srivastava. Minimization of tree pattern queries. In *SIGMOD Conference*, pages 497–508, 2001.
2. S. Cohen, W. Nutt, and Y. Sagiv. Rewriting queries with arbitrary aggregation functions using views. *ACM Trans. Database Syst.*, 31(2), 2006.
3. A. Deutsch and V. Tannen. Reformulation of XML queries and constraints. In *ICDT*, pages 225–241, 2003.
4. G. Gou, M. Kormilitsin, and R. Chirkova. Query evaluation using overlapping views: completeness and efficiency. In *SIGMOD Conference*, pages 37–48, 2006.
5. A. Y. Halevy. Answering queries using views: A survey. *VLDB J.*, 10(4):270–294, 2001.
6. L. V. S. Lakshmanan, H. Wang, and Z. J. Zhao. Answering tree pattern queries using views. In *VLDB*, pages 571–582, 2006.
7. B. Mandhani and D. Suciu. Query caching and view selection for XML databases. In *VLDB*, pages 469–480, 2005.
8. G. Miklau and D. Suciu. Containment and equivalence for an XPath fragment. In *PODS*, pages 65–76, 2002.
9. A. Nash, L. Segoufin, and V. Vianu. Determinacy and rewriting of conjunctive queries using views: A progress report. In *ICDT*, pages 59–73, 2007.
10. N. Onose, A. Deutsch, Y. Papakonstantinou, and E. Curtmola. Rewriting nested XML queries using nested views. In *SIGMOD Conference*, pages 443–454, 2006.
11. J. Wang, R. W. Topor, and M. J. Maher. Rewriting union queries using views. *Constraints*, 10(3):219–251, 2005.
12. W. Xu and Z. M. Özsoyoglu. Rewriting XPath queries using materialized views. In *VLDB*, pages 121–132, 2005.