Consistency of Temporal XML Documents

Marcela Campo[1] and Alejandro Vaisman[2]

[1] Universidad de Buenos Aires
mcampo@dc.uba.ar
[2] Universidad de Chile
avaisman@dcc.uchile.cl

Abstract. Different models have been recently proposed for representing temporal data, tracking historical information, and recovering the state of the document as of any given time, in XML documents. After presenting an abstract model for temporal XML, we discuss the problem of the validation of the temporal constraints imposed by this model. We first review the problem of checking and fixing isolated temporal inconsistencies. Then, we move on to study validation of a document when many temporal inconsistencies of different kinds are present. We study the conditions that allow to treat each inconsistency isolated from the rest, and give the corresponding proofs. These properties are intended to be the basis of efficient algorithms for checking temporal consistency in XML.

1 Introduction

The problem of validating an XML document with respect to a set of integrity constraints after an update occurs, has recently attracted the attention of the database community. Many incremental validation techniques have been proposed [2,3,10,14]. In the temporal XML setting, although several models exist for representing, querying and updating temporal information [1,5,7,8], little attention has been given to the problem of validating the temporal constraints imposed by these models. In Temporal XML documents, the updates must take as input (and return) a valid XML document, not only with respect to the usual set of integrity constraints, but also with respect to the temporal constraints defined by the model at hand. Further, more often than not, we will not be working with documents built from scratch using update operations, but with a pre-existent temporal XML document; thus, we will need to efficiently check if this document verifies a set of temporal constraints, and, if not, provide the user with tools for fixing the inconsistencies, if needed.

In this work we address the problem of validating a set of temporal constraints in a temporal XML document. Although our proposal is based in the data model introduced in [12] (discussed in more detail in Section 3), it could be extended to other data models for temporal XML. After presenting and discussing the data model, we characterize temporal inconsistencies in temporal XML documents. We then introduce the problem of checking inconsistencies in a document, and fixing individual inconsistencies. Then, we move on to a more realistic scenario, where many inconsistencies could appear concurrently, and study the conditions under which these inconsistencies could be teated isolated from each other. These properties could then be embedded in efficient algorithms for fixing inconsistencies in temporal XML documents. To the best of our knowledge, this is the first contribution in this topic.

The remainder of the paper is organized as follows: in Section 2 we review previous efforts in temporal semistructured/XML data. In Section 3 we introduce the temporal data model. Section 4 presents the main kinds of inconsistencies that may appear in a temporal XML document, and discusses how they can be fixed. Section 5 addresses documents where more than one consistency condition is violated. We conclude in Section 6.

2 Related Work

Some proposals have been recently presented addressing incremental validation of XML documents. Kane *et al* [10] model XML constraints as rules, and present a constraint checking mechanism for update operations, aimed at ensuring that the result of an update leaves the XML document in a consistent state. Basically, this is performed by rewriting an update query into a so-called *safe update* query. Incremental validation of XML documents has also been studied in [2,3,14].

Chawathe *et al* proposed a historical model for *semistructured data* [4], that extends the Object Exchange Model (OEM) with the ability to represent updates and to keep track of them by means of "deltas". Along the same lines, Oliboni *et al* [13] proposed a graphical data model and query language for semistructured data. Both works assume that the documents are consistent with respect to the temporal constraints the models impose. Several data models for *Temporal XML* have been proposed. All of them lack of a mechanism for checking the underlying temporal constraints. In contrast, in this paper we study different ways of tackling (temporal) consistency in temporal XML documents. Amagasa *et al* [1] introduced a temporal data model based in XPath, but not a model for updates, nor a query language taking advantage of the temporal model. Dyreson [7] proposed an extension of XPath with support for transaction time by means of the addition of several temporal axes for specifying temporal directions. Chien *et al* [5] proposed update and versioning schemes for XML, through a scheme where version management is performed by keeping references to the maximal unchanged subtree in the previous version. A similar approach was also followed by Marian *et al* [11]. Gao *et al* [8] introduced τXQuery, an extension to XQuery supporting valid time while maintaining the data model unchanged. Queries are translated into XQuery, and evaluated by an XQuery engine. Finally, Wang *et al* have also proposed solutions based in versioning [16]. In this paper we will work over a data model first introduced in [12].

3 Temporal XML Documents

We will introduce the model through an example, depicted in Figure 1. This is an abstract representation of a temporal XML document for a portion of a

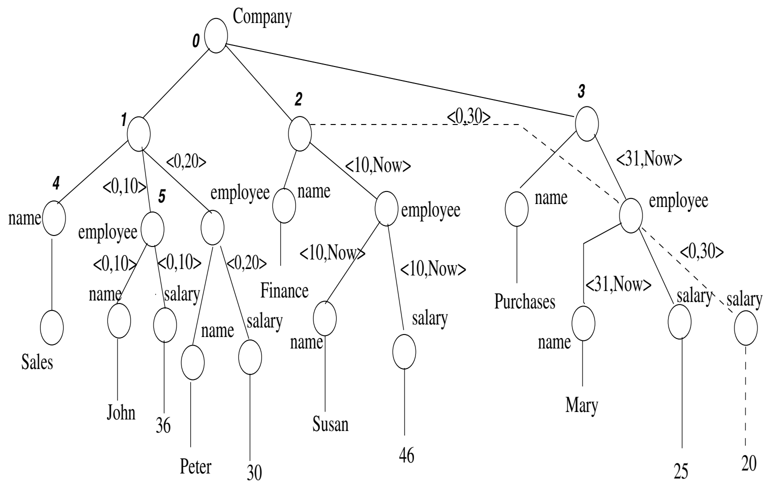

Fig. 1. Example database

company involving departments and their employees. The database also records salaries and, probably, other properties of the employees. Note that in this model, employee's nodes are not duplicated throughout time. For example, can see that John and Peter worked for the Sales department in the intervals [0,10] and [0,20] respectively, while Susan has been working for the Finance department since instant "10". When an edge has no temporal label, its validity interval is assumed to be [0,Now] (i.e. the complete lifespan of the node). Thus, the abstract representation of the temporal document presented in Figure 1 contains the whole history of the company. We can then query the state of the database at a certain point in time, or pose temporal queries like "employees who worked for the Sales department continuously since the year 2000." In [12] the authors provided indexing schemes allowing efficient query evaluation techniques.

More formally, an XML document is a directed labeled graph, where we distinguish several classes of nodes: (a) a distinguished node r, the *root* of the document, such that r has no incoming edges, and every node in the graph is reachable from r; (b) *Value nodes*: nodes representing values (text or numeric); they have no outgoing edges, and have exactly one incoming edge, from attribute or element nodes (or from the root); (c) *Attribute nodes*: labeled with the name of an attribute, plus possibly one of the 'ID' or 'REF' annotations; (d) *Element nodes:* labeled with an element tag, and containing outgoing links to attribute nodes, value nodes, and other element nodes. Each node is uniquely identified by an integer, the *node number,* and is described by a string, the *node label.* Edges in the document graph are constrained to be either *containment edges* or *reference edges.* A containment edge $e_c(n_i, n_j)$ joins two nodes n_i and n_j such that n_i is either r or an element node, and n_j is an attribute node, a value node or another element node; a reference edge $e_r(n_i, n_j)$ links an attribute node n_i of type REF, with an element node n_j. We add the time dimension to document graphs labeling edges with intervals. We will consider time as a discrete, linearly ordered domain. An ordered pair $[a, b]$ of time points, with $a \leq b$, denotes the closed interval from a to b. As usual in temporal databases, the current time point will be represented with the distinguished word '*Now*'. The document's creation instant will be indistinctly denoted by the instant "0′′.

A *temporal label* over a a containment edge $e_c(n_i, n_j)$, is an interval T_{e_c} representing the time period when the element represented by n_j was contained in the element represented by n_i. Our model supports *transaction time* of the containment relation. Although we do not deal with *valid time*, it could be addressed in an analogous way. Analogously, for reference edges, T_{e_r} represents the *transaction* time of the reference edge $e_r(n_i, n_j)$. We note that the full model supports other kinds of nodes, like *versioned* and *attribute* nodes, that we will not consider here. We will use $T_e.TO$ and $T_e.FROM$ to refer to the endpoints of the interval T_e. Two temporal labels T_{e_i} and T_{e_j} are *consecutive* if $T_{e_j}.FROM{=}T_{e_i}.TO + 1$. The *lifespan* of a node is the union of the temporal labels of all the containment edges incoming to the node. The lifespan of the root is the interval $[t_0, Now]$.

Definition 1 (Temporal XML Document). *A* Temporal XML Document *is a document graph, augmented with temporal labels and versioned nodes, that satisfies the following conditions: (1) The union of the temporal labels of the containment edges outgoing from a node is contained in the lifespan of the node. (2) The temporal labels of the containment edges incoming to a node are consecutive. (3) For any time instant t, the sub-graph composed by all containment edges e_c such that $t \in T_{e_c}$ is a tree with root* r, *called the* snapshot *of $\mathcal{D}$ at time t, denoted $\mathcal{D}(t)$. (4) For any containment edge $e_c(n_i, n_j, T_{e_c})$, if n_j is a node of type ID, the time label of e_c is the same as the lifespan of n_i; moreover, if there are two elements in the document with the same value for an ID attribute, both elements are the same. In other words, the ID of a node remains constant for all the snapshots of the document. (5) For any containment edge $e_c(n_i, n_j, T_{e_c})$, if n_j is an attribute of type REF, such that there exists a reference edge $e_r(n_j, n_k, T_r)$, then $T_{e_c} = T_{e_r}$ holds. (6) Given a reference edge $e_r(n_i, n_j, T_{e_r})$, $T_{e_r} \subseteq l_{n_j}$ holds.*

Note that the second condition in Definition 1 implies that we will be working with single intervals instead of temporal elements. This assumption simplifies the presentation and makes the implementations more efficient, although it imposes some constraints on the model. Our definitions and theorems can be, however, extended to the case of temporal elements, overriding the former limitation. Discussion on this topic, and a more detailed description of the model, can be found in [12]. We will also need the following definition:

Definition 2 (Continuous Path and Maximal Continuous Path). *A* continuous path *with interval T from node n_1 to node n_k in a temporal document graph is a sequence $(n_1, \ldots, n_k, T)$ of k nodes and an interval T such that there is a sequence of containment edges of the form $e_1(n_1, n_2, T_1)$, $e_2(n_2, n_3, T_2)$, $\ldots$, $e_k(n_{k-1}, n_k, T_k)$, such that $T = \bigcap_{i=1,k} T_i$. We say there is a maximal* *continuous path (mcp) with interval T from node n_1 to node n_k if T is the union of a maximal set of consecutive intervals T_i such that there is a continuous path from n_1 to n_k with interval T_i.*

4 Consistency in Temporal XML

In this section we will summarize previous results on the problem of checking consistency, and fixing isolated inconsistencies. Details can be found in [15]. In addition, we will give the definitions and concepts needed for studying the general problem (i.e., multiple inconsistencies), that we discuss in the next section.

Definition 3 (Inconsistencies in Temporal XML). *The constraints stated in Definition 1 are violated if: (i) there is an outgoing containment edge whose temporal label is outside the node's lifespan; (ii) the temporal labels of the containment edges incoming to a node are not consecutive. Here, the inconsistency may be due to a gap or an overlapping of the temporal labels of the edges incoming to a node; (iii) there is a cycle in some document's snapshot; (iv) there exist more than one node with the same value for the ID attribute. We will denote these types of inconsistencies as inconsistencies of type i, type ii, type iii, and type iv. An* Interval of Inconsistency, *denoted I_I is the closed interval where consistency conditions in Definition 3 are not satisfied.*

As we will only consider temporal issues in this paper, we will only study inconsistencies of types *i* through *iii*. Also, we will work with documents containing no IDREF or IDREFS attributes.

Example 1. Figures 2 (a) to (c) show examples of inconsistencies of types *i* through *iii*, and their intervals. In Figure 2(a) $I_I = [T_4, Now]$; in Figure 2 (b) $I_I = [T_2, T_4]$; in Figure 2 (c) there is a cycle in every snapshot within the interval $I_I = [T_4, T_6]$.

Checking Consistency. Inconsistencies of types *i* and *ii* are checked using the function $lifespan(n)$, that, given a node n computes its lifespan. For inconsistencies of type i, the algorithm checks, for each edge e, if T_e is in $lifespan(n)$. If there is an inconsistency of type ii, $lifespan(n)$ returns $null$. It can be shown that the lifespan of a node can be computed with an order $O(deg_{in}(n)* \; log(deg_{in}(n)))$, where $deg_{in}(n)$ is the number of edges incident to n. In the worst case (where all edges in the graph are incident to the node), the order of the algorithm is $O(|E| * log(|E|)$; in the average case (all nodes have the same number of incoming edges, i.e. $\frac{|E|}{|V|}$), this reduces to $O(\frac{|E|}{|V|} * log(\frac{|E|}{|V|})$. In the best case (when each node has only one incoming edge) the lifespan is computed in constant time.

Inconsistencies of type *iii* are checked (with order $O(|E| + |V|)$) using the following proposition.

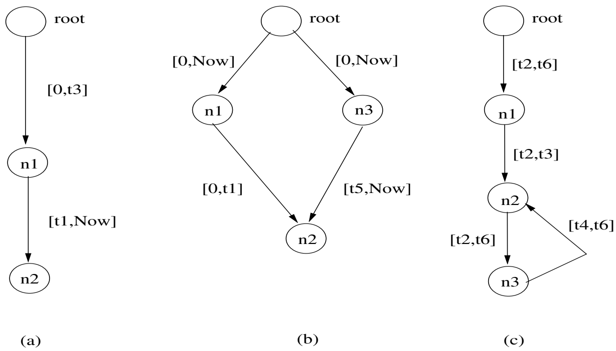

Fig. 2. (a) Inconsistency of type *i*; (b) Inconsistency of type *ii*; (c) Inconsistency of type *iii*

Proposition 1. *Let D be a Temporal XML document where every node has at most one incoming containment edge in every time instant* t; *if there is a cycle in some interval* I_I *in* D, *then, there exists a node* n_i *such that* $T_{mcp}(n_i) \neq lifespan(n_i)$, *where* $T_{mcp(n_i)}$ *is the temporal interval of the mcp between the root and node* n_i.

Definition 4 (Deleting edges). *Let D be a Temporal XML document, and let e be a containment edge* $e(n_i, n_j, T_e)$. *We define three different kinds of deletion of containment edges: (1)* Physical Delete *of an edge e is the deletion of e during all the edge's lifespan. (2)* Delete e in an instant $t \in Te$, with three variants*: (a)* Physical delete e*, if* $T_e.FROM = T_e.TO = t$; *(b)* make $T_e.TO = t - 1$, *if* $T_e.TO = t \wedge T_e.FROM < t$; *(c)* make $T_e.TO = t{+}1$, *if* $T_e.TO = t \wedge T_e.FROM > t$; *(d) Create a duplicate of* n_j *at instant t, and delete e in t (see below) if* $T_e.FROM < t < T_e.TO$. *(3)* Delete e in an Interval I *is the deletion of the edge e for each instant* $t, t \in I \cap T_e$.

Duplication of a node n at instant t_d is performed as follows: (1) create a new node n_c, and, for all edges $e_j(n, n_i, T_{e_j})$ outgoing from n, create a new edge $e_k(n_c, n_i, T_{e_j})$; (2) delete (following Definition 4), all edges outgoing from n, for all instant $t \geq t_d$; (3) delete all edges outgoing from n_c, for all instant $t < t_d$; (4) for each edge $e_i(n_i, n, T_{e_i})$ incident to n such that $T_{e_i}.TO \geq t$ create a new edge $e_n(n_i, n_c, T_{n_i})$ with $T_{n_i}.FROM = t$ if $t \in T_{n_i}$, and $T_{n_i} = T_{e_i}$ otherwise; (5) finally, delete all edges e_i in the interval $[t, T_{e_i}.TO]$ if $t \in T_{e_i}$. It can be shown node duplication can be performed in $O(deg_{out}(n) + deg_{in}(n)) \approx O(|E|)$ time.

The first kind of deletion in Definition 4 is a physical deletion, that is, the whole edge disappears. The second kind of deletion has different flavors. If the edge is deleted in an instant that corresponds to a boundary of it interval of validity (T_e), this boundary is incremented (decremented) in one time unit. Finally, if the edge is deleted in an instant inside T_e, the target node of the edge is split into two, as explained above. Deletion during an interval is just a straightforward generalization of the above.

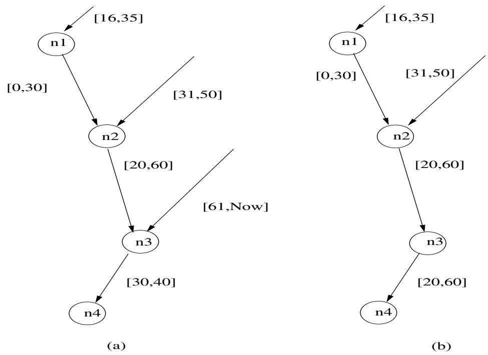

Fig. 3. Inconsistency of type i

Definition 5 (Temporal Label Expansion and Reduction). *Given a containment edge* $e(n_i, n_j, T_e)$*, an expansion of* T_e *to an instant* t *is performed making* $T_e.TO = t$*, if* $t > T_e.TO$*, and* $T_e.FROM = t$*, if* $t < T_e.FROM$*.*

Analogously, reducing the temporal label T_e *to an interval* $T' \subset T_e$ *implies deleting* e *in the intervals* $[T_e.FROM, T'.FROM - 1], [T'.TO + 1, T_e.TO]$.

Given two intervals T_1 and T_1, if $T_1.TO > T_2.TO$ we will say that T_1 is *greater than* T_2, denoted $T_1 \succ T_2$. Analogously, if $T_1.TO < T_2.TO$, we say that T_1 *precedes* T_2, denoted $T_1 \prec T_2$.

Definition 6 (Youngest (Oldest) Incoming Edge). *We will denote youngest edge incoming to a node* n, $y_e(n)$*, an edge whose temporal label is the largest (according to the definition above) among all the temporal labels of the edges incoming to* n*. Analogously we define the oldest edge incoming to a node* n, $o_e(n)$*, as an edge whose temporal label is less than the labels of all the other edges incoming to* n*.*

Fixing Inconsistencies of Type *i*. We study two ways of fixing the problem: (a) correction by expansion; and (b) correction by reduction. *Correction by expansion* expands the lifespan of the inconsistent node until it covers the violating interval; for this task, if $I_I \succ lifespan(n)$, $y_e(n)$ (i.e., the youngest incoming edge) is chosen for expansion; if $lifespan(n) \succ I_I$, $o_e(n)$ is chosen. The problem with this solution is twofold: on the one hand, we do not really know if the containment relation actually existed in the new interval. An expert user will be needed to define this. On the other hand, the expansion may introduce a cycle (i.e., an inconsistency of type iii). In this case, expansion will not be a possible solution. *Correction by reduction* shrinks the temporal label of the inconsistent edge, in order to close I_I. The main idea here is to modify the temporal label of the inconsistent edge, in order that it lies within the lifespan of the starting node of such edge. Although no cycle can be introduced by this solution, new inconsistencies of type i may appear in the ending node of the modified edge, if this node has outgoing edges that cover the interval that has to be reduced; moreover, inconsistencies of type ii may also be introduced if the deleted interval was not in one of the lifespan's extremes. Finally, note that reduction can be propagated downward in cascade.

Example 2. Figure 3(a) shows an inconsistency of type i at node n_2, where $I_I = [51, 60]$. A *correction by expansion* will expand the youngest edge incoming to n_2, resulting in a new label $[31, 60]$. Note that an expansion may recursively propagate the inconsistency upward in the path, until a consistent state is reached. In the same example, the *correction by reduction* approach would generate new inconsistencies of type i and ii. Reducing to [20,50] the interval of the edge (n_2, n_3) in Figure 3 (a), introduces a gap in node n_3. In the case of Figure 3 (b), the same correction will make the temporal label of the edge (n_3, n_4) lie outside the lifespan of node n_3.

Fixing Inconsistencies of Type *ii*. In this case we have two possibilities: (a) there is an overlapping of some of the temporal labels incoming to a node; (b) the union of the temporal labels of the edges incoming to a node presents a gap.

For fixing overlapping it suffices just to delete one of the violating edges in the interval of inconsistency. Closing the gaps has more than one possible solution: (a) physically delete all incoming edges occurring after the gap (i.e., with temporal labels starting after the gap); (b) expand the temporal labels of the edges, in order to close the gap (this could be performed expanding the temporal labels of one or more of the edges involved); (c) duplicate the violating node in a way such that the resulting incoming and outgoing edges have consistent temporal labels. The first two options may introduce new inconsistencies of type i (for example, if the violating node is n, there is an edge $e(n_i, n, T_e)$, and T_e is expanded to T'_e, the latter label may be outside the lifespan of n_i). The third option requires the node created to be semantically equivalent and syntactically consistent. Fixing inconsistencies of type ii can be done in $O(|E|)^2$ time [15].

Fixing Inconsistencies of Type *iii*. Inconsistencies of type iii involve cycles occurring in some interval(s) of the document's lifespan. In this case, again, we have more than one possible way of fixing the inconsistency, basically consisting in deleting (according to Definition 4) edges within the cycle. We may (a) delete all containment edges involved in a cycle during the inconsistency interval I_I (i.e., the interval when the cycle occurs); or (b) delete (within the interval of inconsistency) one of the edges in the cycle. Given that this would introduce an inconsistency of type i, this solution is only possible if there is at least one node n in the cycle with more than one incoming containment edge $e_c(n_i, n, T_e)$, such that T_e lies outside I_I. Thus, besides deleting the edge, T_e must be expanded in order to prevent introducing a new inconsistence.

5 Interaction Between Inconsistencies

So far we have studied document inconsistencies isolated from each other. In a real-world scenario, it is likely that more than one inconsistency appears in a document. In this section we tackle this problem. First, we need some definitions.

Definition 7 (Expansion paths). *We denote* youngest parent of a node n, *the origin node of* $y_e(n)$. *The* oldest parent of a node *is the origin node of* $o_e(n)$. *A* path of oldest (youngest) parents *between two nodes* n_i, n_j *is a path where each node is the youngest (oldest) parent of the next node in the path. We will denote these paths* expansion paths.

Definition 8 (Area of Influence). *We will call* Area of Influence *of an inconsistency* I, *denoted* $A_{inf}(I)$, *the union of all the nodes affected by the possible solutions to the inconsistencies studied in Section 4. An* affected node *is a node changed as a consequence of fixing an inconsistency, i.e., a node such that (a) an incoming or outgoing node was deleted; (b) a temporal label of an incoming or outgoing edge was expanded or reduced.*

For an inconsistency I of type i, the *Area of Influence of* I is the set of nodes composed of: the inconsistent node n, all the nodes in the expansion paths of n, and all the nodes n_i in the document such that there is a continuous path from n to n_i during I_{I_i} (the interval of inconsistency of I).

The *Area of Influence of an Inconsistency of type ii* is only composed of the inconsistent node n. Given that the solution for this kind of inconsistency is the duplication of the node, only the edges are affected (and a new node will be created).

The *Area of Influence of an Inconsistency of Type iii* during an Interval of Inconsistency I_I is composed of: (a) all the nodes n_i in the cycle (*corresponds to the solution of deleting all nodes in the cycle*); (b) all nodes n_j such that there is a continuous path from n_i to n_j with interval $T \supseteq I_{I_{iii}}$ (*a consequence of the above*); (c) all the nodes in the expansion path of each node in the cycle, with temporal label less than I_I; (corresponds to the solution of deleting only one edge in the cycle).

Example 3. Figure 4 (a) shows an example of an inconsistency of type i over node n_3. The possible solutions are, as we have seen before, correction by reduction or by expansion. The former affects all nodes belonging to a path with origin in n_3, in the interval $[t3, t10]$ (i.e., n_4 and n_6). Expansion would affect all nodes in the path of youngest parents of n_3, i.e, n_2 and n_1. Then, the area of influence is the set: $\{n_1, n_2, n_4, n_6, n_3\}$.

Figure 4 (b) depicts the area of influence of a cycle between nodes n_2, n_3, n_4 and n_5. If all of them are deleted, node n_7 will also be affected, because it is reached from n_3 within the cycle's interval. In fact, all nodes, except n_2 will be physically deleted. Deleting only the ending node of one inconsistent edge in the cycle implies deleting node n_2 (we delete $e(n_5, n_2, T_{52})$), expanding the intervals of the path of youngest parents of n_2, i.e., n_1 is also affected.

If more than one inconsistency appears in a temporal XML document, the order in which we solve them will have an impact on the document that we will finally obtain. We would like to identify, at low cost, sets of inconsistencies that do not interfere with each other. In this case, we would be able to fix them in any order, and the result will be the same. The notion of area of influence allows us to identify such sets.

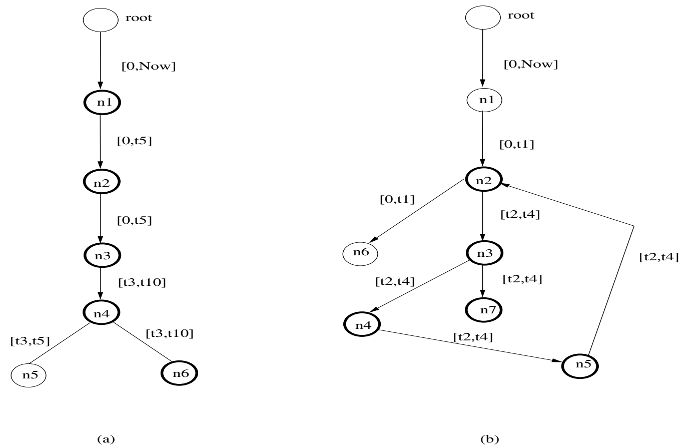

Fig. 4. Inconsistencies of Types i and iii - Area of influence

Intuitively, given any pair of inconsistencies (of any type), I_1 and I_2, we say that I_1 and I_2 *interfere* with each other if their areas of influence have non-empty intersection. Conversely, if $A_{inf}(I_1) \cap A_{inf}(I_2) \neq \phi$, we say that I_1 and I_2 are *isolated* from each other. Given a set of n inconsistencies $I_1, ..., I_n$, we denote the set composed of the nodes in $A_{inf}(I_1) \cap A_{inf}(I_2) ... \cap A_{inf}(I_n)$, the *Area of Interference* of $I_1, ..., I_n$, denoted $A_i(I_1, I_2, ... I_n)$

Definition 9 (Classification of Interferences). *Given a set of inconsistencies $\mathcal{I} = \{I_1, ..., I_n\}$ such that $A_{inf}(I_1) \cap A_{inf}(I_2) ... \cap A_{inf}(I_n) \neq \phi$ we denote their interference* Irrelevant *if we can fix them in any order and obtain the same result (i.e., everything happens as if they were isolated). On the contrary, if this property does not hold, we denote the interference* relevant.

Example 4. Figure 5 (a) shows two irrelevant inconsistencies of type ii over the same node. In both cases, the solution will be node duplication. However, it is easy to see that the result will be the same, no matter which one we address in the first place. Figure 5 (b) shows a cycle interfering with an inconsistency of type i. The cycle cannot be corrected by expansion because it involves nodes in the potential path of youngest parents of the inconsistency of type i.

In what follows, we will study the conditions that state when an interference is irrelevant. Detecting irrelevant interferences through the propositions below, constitutes the basis of an efficient solution to the problem of fixing a document with multiple temporal inconsistencies. We will not address relevant interferences in this paper.

Irrelevant Interferences. In the propositions below, we will be using a simple metric, namely the number of changes needed to fix an inconsistency, where a change could be: (a) the expansion of an interval; (b) the reduction of an interval;

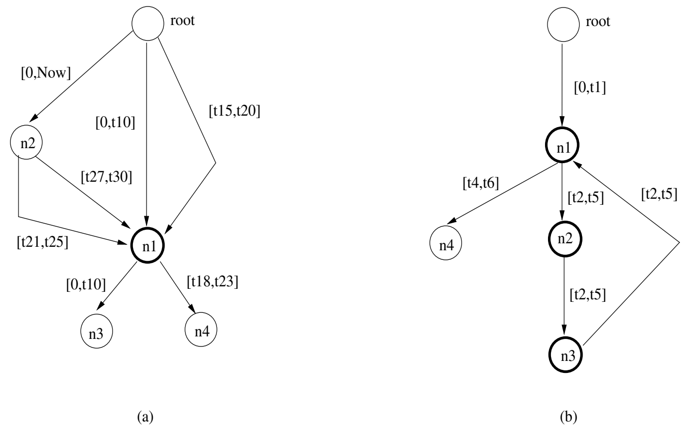

Fig. 5. Relevant and Irrelevant interferences

(c) the duplication of a node; (d) the physical deletion of an edge. For simplicity, we give the same weight to each change. We denote this metric κ. When there is more than one possibility for fixing an inconsistency, this metric will define which method we will use (i.e., the one with the smallest κ). In particular, in Section 4 we proposed two methods for fixing inconsistencies of type i: *correction by expansion* and *correction by reduction*. We denote κ_r and κ_e the number of changes required by a correction by reduction and expansion, respectively.

Definition 10 (Expansion Area). *We call* Expansion Area *of an inconsistency I, denoted $A_e(I)$ the set of nodes belonging to the expansion path(s) that compose the Area of Influence of I. Analogously, we call the* Reduction Area *of I, denoted $A_r(I)$ the set of nodes in all the mcps that compose the Area of Influence of I. It follows that, for inconsistencies of types i and iii, $A_{inf}(I) = A_e(I) \cup A_r(I)$.*

Proposition 2. *Let I_1, I_2 be two inconsistencies of type i, with intervals T_1 and T_2, respectively. If $A_e(I_1) \cap A_r(I_2) = A_e(I_2) \cap A_r(I_1) = \phi$, then, I_1 and I_2 can be solved by expansion, in any order.*

Now, we will give a set of propositions that allows us to determine if two inconsistencies occurring in the same document are irrelevant or not. We will address all possible combinations of inconsistencies, starting from concurrent inconsistencies of the same kind. For the sake of space will only give the proof of some of the propositions.

Proposition 3 (Inconsistencies of type i). *Let I_1, I_2 be two inconsistencies of type i, with intervals T_1 and T_2, respectively. Their interference is irrelevant if at least one of the following holds:*

a. $\kappa_r(I_1) < \kappa_e(I_1) \wedge \kappa_r(I_2) < \kappa_e(I_2) \wedge\ A_e(I_1) \cup A_r(I_2) = A_e(I_2) \cup A_r(I_1) = \phi$
b. $\kappa_r(I_1) > \kappa_e(I_1) \wedge \kappa_r(I_2) > \kappa_e(I_2) \wedge\ A_e(I_1) \cup A_r(I_2) = A_e(I_2) \cup A_r(I_1) = \phi$
c. $A_i(I_1, I_2) = A_r(I_1) \cap A_r(I_2) \wedge T_1 \cap T_2 = \phi$.
d. $A_i(I_1, I_2) = A_e(I_1) \cap A_r(I_2) \wedge T_1 \prec T_2$ *(for a path of youngest parents,*$T_1 \succ T_2$ *for a path of oldest parents)*

Condition (a) means that the number of changes needed to fix each inconsistency *by reduction* is less than the number of changes required to fix it *by expansion*, and the expansion area of one of them does not intersect with the expansion area of the other. Condition (b) is analogous.

Proof. Condition a. If the interference is not irrelevant, fixing one inconsistency would affect the remaining one. Suppose we fix I_1 and I_2 in that order. We know that $\kappa_r(I_1) < \kappa_e(I_1)$, so we must choose reduction for I_1. This implies that the number of changes required for I_2 can never be increased by this process, because $A_r(I_1) \cap A_e(I_2) = \phi$. Thus, reduction will be the also the choice for I_2. We arrive to the same conclusion following the order I_2, I_1. Thus, the interference is irrelevant.

Condition b. Again, suppose we fix I_1 and I_2 in that order. We know that $\kappa_r(I_1) > \kappa_e(I_1)$, so we must choose expansion for I_1. This choice can never increase the number of changes that will be produced expanding I_2. Thus, I_2 will also be fixed by expansion. Moreover, the expansion path remains the same. Thus, the interference is irrelevant.

Condition c. If the order is I_1, I_2, and I_1 is corrected by expansion, the solution for I_2 is not changed because $A_e(I_1)$ is not in the area of interference. I_1 is corrected by reduction $A_r(I_2)$ and $A_e(I_2)$ remain unchanged. This is also true for the order I_2, I_1.

Condition d. If the order is I_1, I_2, and we correct I_1 by reduction, the nodes in $A_r(I_2)$ are not affected because $A_r(I_2) \notin A_i(I_1, I_2)$. If I_1 is fixed by expansion, the nodes in $A_r(I_2)$ are not affected because $T_1 \prec T_2$, and can only be expanded to $T_1.TO$, they are not modified in the interval of inconsistency of I_2. The same occurs if the order is I_2, I_1.

Proposition 4 (Inconsistencies of type ii**).** *Let* I_1, I_2 *be two inconsistencies of type* ii*, with intervals* T_1 *and* T_2*, respectively. Their interference is always irrelevant unless* I_1 *and* I_2 *are both overlappings with a common edge, such that the intersection between the time labels of all edges involved is not empty.*

Proposition 4 states that the only case when the two inconsistencies interfere in a relevant fashion is when, given three edges incident to a node (i.e., there is a common edge), $e_1(n_1, n, T_1), e_2(n_2, n, T_2), e_3(n_3, n, T_3)$, it holds that $T_1 \cap T_2 \neq \phi$, and $T_1 \cap T_3 \neq \phi$.

Proof. We have four possibilities: (1) I_1 and I_2 are gaps over a node n; (2) I_1 and I_2 are overlappings not involving a common edge; (3) there is a common edge (i.e., I_1 and I_2 involve just three edges; (4) I_1 is a gap and I_2 is an overlap.

Case (1). Let $lifespan(n_1) = [T_1, T_2] \cup [T_3, T_4] \cup [T_5, T_6]$, with $T_2 < T_3 - 1, T_4 < T_5 - 1$. Solving I_1 creates a new node n_{1c}, such that we will have

$lifespan(n_1) = [T_1, T_2]$, and $lifespan(n_{1c}) = [T_3, T_4] \cup [T_5, T_6]$. Clearly, fixing I_2 will only affect n_{1c}. The same occurs if we first fix I_2.

Case (2). Let $e_1(n_i, n_1, T_1), e_2(n_j, n_1, T_2), e_3(n_k, n_1, T_3), e_4(n_l, n_1, T_4)$ be edges such that $T_1 \cap T_2 \neq \phi \wedge T_3 \cap T_4 \neq \phi \wedge T_1 \cap T_2 \cap T_3 \cap T_4 = \phi$; also, $e_1 \neq e_2 \neq e_3 \neq e_4$. Fixing the first inconsistency, one of the two edges in the intersection interval. This, clearly, does not affect the remaining inconsistency, and the interference is irrelevant.

Case (3). Let $e_1(n_i, n_1, T_1), e_2(n_j, n_1, T_2), e_3(n_k, n_1, T_3)$, be edges such that $T_1 \cap T_2 \neq \phi \wedge T_2 \cap T_3 \neq \phi \wedge T_1 \cap T_3 = \phi$. If T_2 is reduced in one of the inconsistencies, the other one is not affected. As we did not assume any order, this happens when choosing the orders I_1, I_2 or I_2, I_1.

Case (4). It is clear that a gap and an overlap cannot occur during the same interval. If we first fix the gap (via node duplication), the overlap will remain in one of the nodes and will be fixed as if the gap never existed. If we, instead, fix the overlap first, the gap will not be affected. Thus, the interference is irrelevant.

Now we will address cycles (inconsistencies of type *iii*). In Section 4 we presented two solutions to the problem of fixing a cycle: (a) removing all edges in the cycle, which implies changing the lifespan of the nodes in the cycle, and may produce new inconsistencies of type *i* and *ii*, that will be fixed by reduction and node duplication, respectively. All the nodes affected belong to the reduction area of the inconsistency. Thus, in what follows we will denote this solution, *correction by reduction*, like in Section 4. Analogously, solution (b) (removing one edge in the cycle, if possible), potentially generates an inconsistency of type *ii*, and, as it was explained, an inconsistency of type *i*, which are corrected by expanding the intervals of one of the edges. Thus, all of the affected nodes are in the expansion area of the inconsistency, and we will also denote this solution *correction by expansion.*

Proposition 5 (Inconsistencies of type *iii*). *Let I_1, I_2 be two inconsistencies of type iii, with intervals T_1 and T_2, respectively. Their interference is irrelevant if at least one of the following holds:*

a. $\kappa_r(I_1) < \kappa_e(I_1) \wedge \kappa_r(I_2) < \kappa_e(I_2) \wedge\ A_e(I_1) \cup A_r(I_2) = A_e(I_2) \cup A_r(I_1) = \phi$
b. $\kappa_r(I_1) > \kappa_e(I_1) \wedge \kappa_r(I_2) > \kappa_e(I_2) \wedge\ A_e(I_1) \cup A_r(I_2) = A_e(I_2) \cup A_r(I_1) = \phi$
c. $A_i(I_1, I_2) = A_e(I_1) \cap A_r(I_2) \wedge T_1 \prec T_2$.
d. $A_r(I_1, I_2) = A_r(I_1) \cap A_r(I_2) \wedge T_1 \cap T_2 = \phi$.
e. *Let n_1 be the only node belonging to the cycle in I_1. Then,* $(A_e(I_2) \cap A_r(I_1) = n_1 \vee (A_e(I_2) \cap A_r(I_1) = \phi)\ \wedge (A_r(I_2) \cap A_e(I_1) = \phi \wedge T_1 \cap T_2 = \phi)$

Conditions (a) and (b) are analogous to the ones in Proposition 3, considering the definition of *correction by reduction* and *correction by expansion* for inconsistencies of type *iii*. Condition (e) means that the only node in the Area of Interference is a node belonging to I_1, is in the reduction area of I_2 but not in the expansion area of I_2, and the inconsistency intervals are disjoint.

Proof. (sketch) For conditions (a) and (b), the proofs are similar to Proposition 3. For condition (c) the proof is based on showing that, for instance, for the order I_1, I_2, reduction for solving I_1 does not affect the nodes in $A_r(I_2)$, because $A_r(I_1) \notin A_i(I_1, I_2)$. Expanding, instead, affects the nodes in $A_r(I_2)$, but there is no node added to or deleted from $A_r(I_2)$, because $T_1 \prec T_2$. We proceed analogously for the order I_2, I_1. The proof of condition (d) has a similar mechanism: expanding does not affect, because the expansion areas are not in the area of interference. Eliminating a cycle (reduction) of I_1 (I_2) affects the nodes in $A_r(I_2)$ ($A_r(I_1)$), but in different intervals (because $T_1 \cap T_2 = \phi$.) The idea is analogous for condition (e).

Proposition 6 (Inconsistencies of types *i* and *ii*). *Let I_1, I_2 be two inconsistencies of types i and ii, respectively, with intervals T_1 and T_2. Their interference is irrelevant if one of the following holds:*

a. *I_2 is in the reduction area of I_1, and I_2 is not an overlapping or $T_2 \cap T_1 = \phi$.*
b. *If I_2 is a gap, it occurs on a node in the expansion area of I_1 (i.e., $A_i(I_1, I_2) = A_{inf}(I_2)$), and $T_2 \cap T_1 = \phi$.*

Proposition 7 (Inconsistencies of types *i* and *iii*). *Let I_1, I_2 be two inconsistencies of types i and iii, respectively, with intervals T_1 and T_2. Their interference is irrelevant if one of the following holds:*

a. *The number of changes needed for performing* correction by reduction *of I_1 and I_2 is less than the number of changes needed for performing* correction by expansion *over the same inconsistencies, and their areas of expansion and reduction have an empty intersection.*
b. *The number of changes needed for performing* correction by expansion *of I_1 and I_2 is less than the number of changes needed for performing* correction by reduction *over the same inconsistencies, and their areas of expansion and reduction have empty intersection.*
c. *If there is n in I_1 belonging to $A_r(I_2)$, then $T_1 \cap T_2 = \phi$.*
d. *If there is n in I_2 belonging to $A_e(I_1)$, then $T_1 \cap T_2 = \phi$.*

Proposition 8 (Inconsistencies of types *ii* and *iii*). *Let I_1, I_2 be two inconsistencies of types i and iii, respectively, with intervals T_1 and T_2. Their interference is irrelevant if I_1 is a gap, and the node where it occurs is not in the expansion path of I_2.*

6 Conclusion

We have studied the problem of validating a set of temporal constraints in a temporal XML document, based in the data model presented in [12]. We proposed methods for checking the presence of inconsistencies in a document, and fixing them. We studied individual and combined inconsistencies, and state a set of conditions that make irrelevant the interference between them (i.e., each one can be treated and fixed independently from any other one). These conditions can be incorporated into algorithms for efficiently performing the fixing procedure.

This work can be a good starting point for studying and reasoning about temporal constraints with indeterminate dates, of the types presented in [6,9].

Acknowledgements. This work was supported by the Millennium Nucleus Center for Web Research, Grant P04-67-F, Mideplan, Chile.

References

1. T. Amagasa, M. Yoshikawa, and S. Uemura. A temporal data model for XML documents. In *Proceedings of DEXA Conference*, pages 334–344, 2000.
2. A. Balmin, Y. Papakonstantinou, and V. Vianu. Incremental validation of xml documents. *ACM Transactions on Database Systems*, 29(4):710–751, 2004.
3. D. Barbosa, A.O. Mendelzon, L. Libkin, L. Mignet, and M. Arenas. Efficient incremental validation of XML documents. In *ICDE*, pages 671–682, 2004.
4. S. Chawathe, S. Abiteboul, and J. Widom. Managing historical semistructured data. In *Theory and Practice of Object Systems, Vol 5(3)*, pages 143–162, 1999.
5. S. Chien, V. Tsotras, and C. Zaniolo. Efficient management of multiversion documents by object referencing. In *Proceedings of the 27th International Conference on Very Large Data Bases*, pages 291–300, Rome, Italy, 2001.
6. C. Dyreson and R. Snodgrass. Supporting valid-time indeterminacy. *ACM Transactions on Database Systems*, 23(1):1–57, 1998.
7. C.E. Dyreson. Observing transaction-time semantics with TTXPath. In *Proceedings of WISE 2001*, pages 193–202, 2001.
8. C. Gao and R. Snodgrass. Syntax, semantics and query evaluation in the τXQuery temporal XML query language. *Time Center Technical Report TR-72*, 2003.
9. F. Grandi and F. Mandreoli. Effective representation and efficient management of indeterminate dates. In *TIME'01*, pages 164–169, 2001.
10. B. Kane, H. Su, and E. Rundensteiner. Consistently updating XML documents using incremental constraint check queries. In *WIDM*, pages 1–8, 2002.
11. A. Marian, S. Abiteboul, G. Cobena, and L. Mignet. Change-centric management of versions in an XML warehouse. In *Proceedings of the 27th VLDB Conference*, pages 581–590, Rome, Italy, 2001.
12. A.O. Mendelzon, F. Rizzolo, and A. Vaisman. Indexing temporal XML documents. In *Proceedings of the 30th International Conference on Very Large Databases*, pages 216–227, Toronto, Canada, 2004.
13. B Oliboni, E. Quintarelli, and L. Tanca. Temporal aspects of semistructured data. *Proceedings of the Eight International Symposium of Temporal Representation and Reasoning*, pages 119–127, 2001.
14. Y. Papakonstantinou and V. Vianu. Incremental validation of XML documents. In *ICDT*, pages 47–63, 2003.
15. F. Rizzolo and A. Vaisman. Temporal XML documents: Model, index and implementation. *Submitted*, 2006.
16. F. Wang and C. Zaniolo. Temporal queries in xml document archives and web warehouses. In *Proceedings of the 10th International Symposium on Temporal Representation and Reasoning (TIME'03)*, pages 47–55, Cairns, Australia, 2003.