XML Indexing & Storage System

XISS
- Introduction
- Numbering
- Algorithms
- Data Set
- Experiments

XISS/R
- Introduction
- Schemas
- Architecture
- Query
- Data Set

Sponsor

This research is being sponsored by National Science Foundation CAREER Award IIS-9876037 and Research Infrastructure program EIA-0080123.

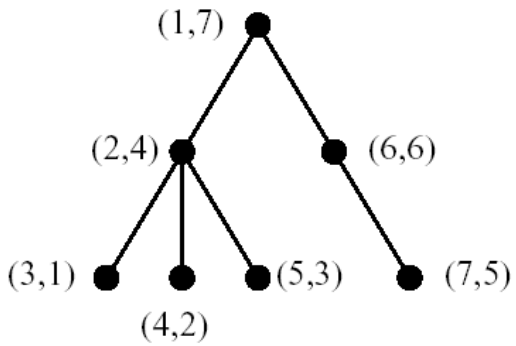
Extended Preorder Numbering Scheme

XML data objects are commonly modeled by a tree structure, where nodes represent elements, attributes and text data, and parent-child node pairs represent nesting between XML data components. To speed up the processing of regular path expression queries, it is important to be able to quickly determine ancestor-descendant relationship between any pair of nodes in the hierarchy of XML data.

For example, a query with a regular path expression `chapter3//figure` is to find all `figure` elements that are included in `chapter3` elements. Once all `chapter3` elements and `figure` elements are found, those two element sets can be joined to produce all qualified `chapter3` - `figure` element pairs. This join operation can be carried out without traversing XML data trees, if the ancestor-descendant relationship for a pair of `chapter3` and `figure` elements can be determined quickly. This is the main idea of the proposed algorithms.

To the best of our knowledge, it was Dietz's numbering scheme that was the first to use tree traversal order to determine the ancestor-descendant relationship between any pair of tree nodes. His proposition was:

For two given nodes x and y of a tree T, x is an ancestor of y if and only if x occurs before y in the preorder traversal of T and after y in the post-order traversal.



For example, consider a tree in the above figure, whose nodes are annotated by Dietz's numbering scheme. Each node is labeled with a pair of preorder and post-order numbers. In the tree, we can tell node (1,7) is an ancestor of node (4,2), because node (1,7) comes before node (4,2) in the preorder and after node (4,2) in the post-order.

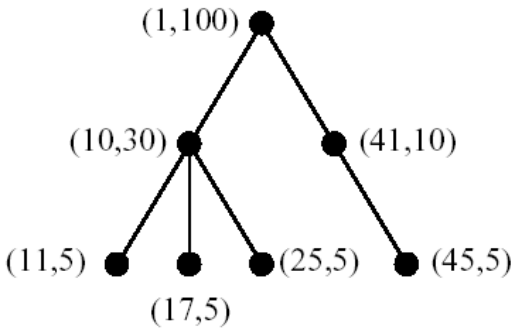
An obvious benefit from this approach is that the ancestor-descendant relationship can be determined in constant time by examining the preorder and post-order numbers of tree nodes. On the other hand, the limitation of this approach is the lack of flexibility. That is, the preorder and post-order may need to be recomputed for many tree nodes, when a new node is inserted.

To get around this problem, we propose a new numbering scheme that uses an *extended preorder* and a *range of descendants*. The proposed numbering scheme associates each node with a pair of numbers <order, size> in a way described as follows.

- For a tree node y and its parent x, order(x) < order(y) and order(y) + size(y) <= order(x) + size(x). In other words, interval [order(y), order(y)+size(y)] is contained in interval [order(x), order(x) + size(x)].
- For two sibling nodes x and y, if x is the predecessor of y in preorder traversal, order(x) + size(x) < order(y).

Then, For a tree node x, size(x) >= sum(size(y)) for all y's that are a direct child of x. Thus, size(x) can be an arbitrary integer larger than the total number of the current descendants of x, which allows to accommodate future insertions gracefully.

It is not difficult to show that the nodes ordered by this proposed numbering scheme is equivalent to that of preorder traversal. That is, the proposed numbering scheme guarantees that for a pair of tree nodes x and y, order(x) < order(y) if and only if x comes before y in preorder traversal. Furthermore, the ancestor-descendant relationship for a pair of nodes can be determined by examining their order and size values.



In the above figure, each node is labeled by a <order, size> pair, which defines an interval. The interval of a node is contained in the interval of its parent node. For example, a node (25,5) is contained in both (10,30) and (1,100). Hence, the node with order 25 is a descendant of nodes with order 10 and 1. This observation leads to the following lemma.

> *For two given nodes x and y of a tree T, x is an ancestor of y if and only if order(x) < order(y) <= order(x) + size(x).*

Compared with Dietz's scheme, our numbering scheme is more flexible and can deal with dynamic updates of XML data more efficiently. Since extra spaces can be reserved in what we call extended preorder to accommodate future insertions, global reordering is not necessary until all the reserved spaces (ie, unused order values) are consumed. Note that for both numbering schemes, deleting a node does not cause renumbering the nodes. However, it is easier for our numbering scheme to *recycle* the order values of deleted nodes.