Transformation of XML Data Sources for Sequential Path Mining

Ruth McNerlan[1], Yaxin Bi[1*], Guoze Zhao[2]and Bing Han[2]

[1]School of Computing and Mathematics
University of Ulster at Jordanstown
Newtownabbey, Co. Antrim, BT37 0QB, UK
[2]State Key Laboratory of Earthquake Dynamics
Institute of Geology, China Earthquake Administration
Beijing 100029, China

Abstract. In recent years XML has become one of the most promising ways to define semi-structured data. Data mining techniques devised for detecting interesting patterns from semi-structure data have also grown in popularity, but carrying out such techniques on XML data can be problematic due to its hierarchical structure. Therefore, it has become necessary to transform XML into flattened, path data, so as to enable data mining to be carried out efficiently. However, problems may arise when the XML tree needs to be reconstructed from the traversal path. There are currently many transformation techniques for XML data, many of which take advantage of its tree-like hierarchical structure; but most of these approaches do not allow the XML tree to be reconstructed from the traversal path. In this paper we propose a new approach to the transformation of XML data into path data. The new approach employs a 5 step transformation process along with a new 'Postorder Sequencing' method of traversing the XML tree. The proposed method, on the one hand, can be seen an efficient and effective way of transforming XML data into collections of paths, and on the other hand enables XML trees to be generated from the traversal paths

Keywords: XML, Transformation, XPath, Sequential Data Mining.

1 Introduction

The eXtensible Mark-up Language (XML) has established itself in recent years as the de facto language for data exchange on the internet. Since XML is self-describing, it is considered one of the most promising means to define semi-structured data which is expected to be ubiquitous in large volumes from diverse data sources and applications on the web [1]. However, information is encoded differently by different information systems. Therefore, to let these information systems communicate and interoperate, it is necessary to transform XML documents [2]. It can be argued that the pinnacle of

* Corresponding author: y.bi@ulster.ac.uk

XML usefulness is its ability to be transformed into a number of different formats including web pages, databases, knowledge bases and other XML documents. Transformation of XML data is especially important when XML is used as the universal data interchange format among web based applications.

Data mining is another field which is growing in popularity with techniques such as association, clustering and classification being used to find similarities and interesting patterns in data. However, the hierarchical, tree-like structure of XML data makes it very difficult to carry out these data mining techniques. Transforming XML data into path data is one way of facilitating data mining on such documents. However, as the result of the data mining will be in the form of path data, it is necessary to have a way of transforming the results into XML as this is a much more effective way of representing data. Therefore, an approach to transforming XML data into path data which enables the generation of XML trees is required.

There are currently many different approaches to transforming XML data. The hierarchical nature of XML lends itself very well to being modelled as a tree-like structure and therefore many approaches to transforming XML involve traversing the XML tree – i.e. – visiting each node of the XML document systematically. There are three main methods of tree traversal – preorder, inorder and postorder – none of which alone permit the tree to be reconstructed from the traversal sequence. Therefore, this paper focuses on the development of a transformation approach for XML data which allows an XML tree to be generated from the traversal path.

2 XML Transformation

There are a number of different approaches to XML transformations that that have been implemented in recent years that attempt to make XML transformations quicker and more efficient. However, there is very limited amount of literature that focuses on developing a transformation approach that allows for the generation of an XML tree from the traversal path. Areas of current research within the topic of XML transformations include transforming between relational databases and XML documents, ways of making XML queries more efficient, and structural join operations in XML queries. Therefore, it can be argued that there is a need for research within the area of XML tree generation from path data.

2.1 Tree Traversal

An XML document can be modelled as an ordered, labelled tree (as illustrated in Fig. 1), with a document node serving as the root node [3]. Each node in the tree corresponds to an element or a value. Values are represented by character data and occur at the leaf nodes. The tree edges represent a relationship between two elements or between an element and a value; and each element can have a list of attribute-value pairs associated with it [4]. The tree structure is considered as the most advantageous data

structure in terms of knowledge representation, navigation and access time and therefore many approaches to transforming XML data are based on the tree structure [5]. Please note that the first paragraph of a section or subsection is not indented. The first paragraphs that follows a table, figure, equation etc. does not have an indent, either.

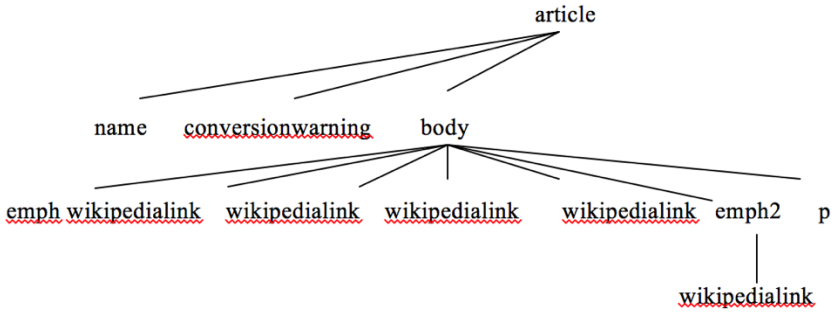


Fig. 1. A sample XML tree

Trees are well studied mathematical entities that are used in many ways for organizing and storing information. In all such cases, the information which is represented as terminal nodes (leaves) is retrieved through simple search algorithms that systematically visit every node in the tree. They typically do this in a recursive manner, starting at the root node and descending the tree node by node in the particular order dictated by the algorithm until the search is terminated. The three most commonly used algorithms are, preorder, postorder and inorder with each method producing a different order in which the nodes are visited during the traversal [6]. For example, consider the binary tree in Fig. 2. The preorder traversal visits the root node first and then each child in turn from left to right. The preorder traversal sequence of the tree in figure 1 would be A, B, D, E, C, F, G. The inorder traversal method visits the root node in between visiting the left and right subtrees. The inorder traversal sequence of the tree in Fig. 1 would be D, B, E, A, F, C, G. Post order traversal visits the children first before the parent and then visits the root node last. The postorder traversal of the tree in Fig. 1 would be D, E, B, F, G, C, A.

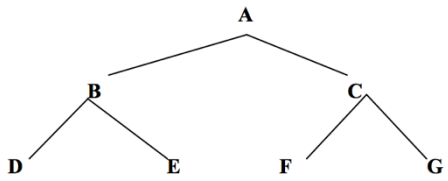


Fig. 2. A binary tree

2.2 Tree Reconstruction

A major limitation of these traversal methods is that it is impossible to reconstruct an XML tree from the traversal path. For example, if ABC is the preorder sequence of a binary tree we can see that A must be the root, but the sequence doesn't tell us if B is the left child of the root or the right child. If ABC is the inorder sequence we don't even know which is at the root node. And if ABC is the postorder sequence, other than knowing that C is at the root we don't know if A is the left or right child and whether B is the left child of A or the right child. We need the inorder sequence and either the preorder or postorder sequence in order to be able to reconstruct the original tree. The inorder sequence will resolve the left-right problem and the preorder or postorder sequence will tell us the roots of the various subtrees. For example, if the inorder sequence is CBA and the preorder sequence is CAB, then we know that C is at the root and both B and A are in the right subtree. The left subtree of the root is empty. Since A comes next in the preorder sequence it is the root of the right subtree and looking at the inorder sequence B is its left child and we have a unique tree [7].

However, XML trees are general trees rather than binary trees as they have nodes with more than two children and there is no sense of left or right. The inorder traversal method is only applicable to binary trees, thus the method of combining the inorder and preorder or postorder traversal sequences to reconstruct the tree does not work for XML trees. Therefore, an alternative approach to traversing XML documents is required that allows the tree to be reconstructed from the traversal path.

2.3 XPath

There are currently many different approaches to transforming XML data including the W3 standard eXtensible Stylesheet Language for Transformations (XSLT) and the XML Query Language (XQuery); and the one thing the majority of these approaches have in common is the use of regular path expressions. Regular path expressions are based on the syntax of the XML Path Language (XPath). They describe traversals through an XML document and consist of one or more steps separated by a slash (/) [8].

The name of the XPath language derives from its most distinctive feature, the path expression, which provides a means of hierarchic addressing of the nodes in an XML tree. XPath gets its name from its use of a path notation for navigating through the hierarchical structure of an XML document. It uses a compact, non-XML syntax to facilitate use of XPath within URIs and XML attribute values. The language provides several kinds of expressions which may be constructed from keywords, symbols, and operands.

3 Tree Reconstruction from Path Data

There are two main problems associated with attempting to generate a tree from a traversal path. These are determining ancestor-descendant relationships and differentiating between identical siblings. Therefore, an approach to XML transformation is required which provides a solution to both these problems and allows an XML tree to be generated from the traversal path.

3.1 Ancestor-Descendant Relationships

XML indexing is one common method that is used to determine ancestor-descendant relationships [9]. XML data can be queried by a combination of value search and structure search. The former includes matching document names, element names/values, and attribute names/values. The latter is usually specified by regular path expressions and is done by examining ancestor-descendant relationship in an XML tree. The main role of indexing XML data is to support both these searches [6].

Another approach to determining ancestor-descendant relationships involves labelling the nodes of an XML tree with a numbering scheme. It was Dietz's numbering scheme that was first to use tree traversal order to determine the ancestor-descendant relationship between any pair of nodes. Dietz uses preorder and postorder traversal order to

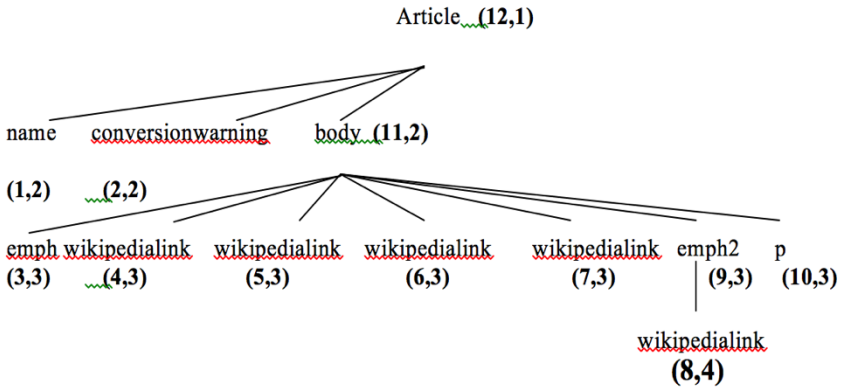


Fig. 3. An XML tree labeled with Dietz's numbering scheme

achieve this [10]. Dietz's approach is based on the proposition that for any 2 given nodes x and y of an XML tree, T, x is an ancestor of y if x occurs before y in the preorder traversal of T and after y in the postorder traversal. Dietz's numbering scheme labels each node with a pair of preorder and postorder numbers, as illustrated in Fig. 3. Examining these numbers allows the ancestor-descendant relationship to be determined.

3.2 Identical Siblings

Along with ancestor-descendant relationships, another problem which is encountered when attempting to reconstruct the XML Tree from the traversal path is identical siblings. For example, if we look again at the example in Fig. 1. A combination of inorder and either preorder or postorder traversal can only be used to reconstruct the binary tree from the traversal path if the tree does not contain any identical siblings – i.e. – any two or more nodes that are the same. Therefore, an approach to XML transformation is needed which solves both problems of ancestor-descendant relationships and identical siblings.

4 A New Approach to XML Transformations

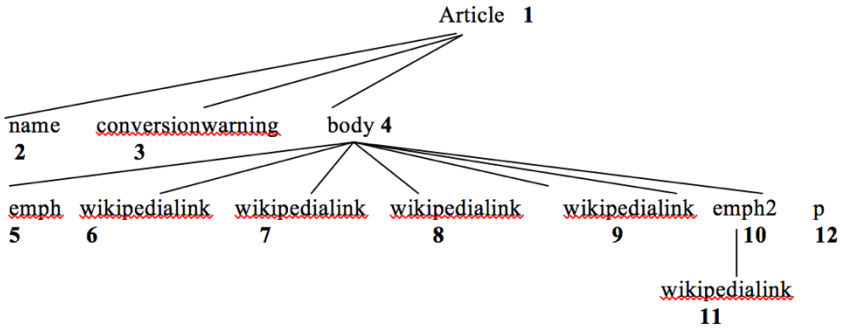
In this study, we propose Postorder Sequencing as a solution to the problem of tree generation. This new approach to XML transformations provides an effective and efficient way to transform XML data into path data and allows XML trees to be generated from the traversal path. We also propose using the Postorder Sequencing method as part of a 5 step approach to the transformation of XML documents.

4.1 Postorder and Level Order Numbering

Given an XML tree are many possible traversal paths that may be navigated through it. Fig. 4 depicts an XML tree labelled with a preorder numbering scheme and shows several of the possible paths that can be traversed through the tree. However, none of these paths make it possible to reconstruct the XML tree; more information is needed to be able to determine the exact positioning of each node. We discovered that one solution to this problem is to use a combination of traversal paths to provide the information necessary to enable tree reconstruction. In a similar scheme to Dietz numbering scheme [11], we labeled an XML tree with postorder and level order numbers as shown in Fig. 5 which can be seen as a solution to the problem of tree reconstruction. Level order traversal visits the nodes in order of their depth in the tree, i.e. the root node is at level one, the immediate children of the root node are at level two and their children are at level three etc. Being equipped with both postorder and inorder traversal number patterns when attempting to reconstruct the original XML tree, enables the ancestor-descendant relationship to be determined as the exact position of the node in the tree including depth and horizontal position can be ascertained.

4.2 Postorder Sequencing

We propose postorder sequencing as a new approach to XML transformations which allows for the generation of XML trees as it solves both problems of ancestor-descendant relationships and identical siblings. It can be seen as a development of the level order and postorder numbering scheme but represented in a more effective way.



Path 1: 1,2,3,4,5,6,7,8,9,10,11,12 (preorder)
Path 2: 2, 3, 5,6,7,8,9,11,10,12, 4,1 (postorder)
Path 3: 1,2,3,4,5,6,7,8,9, 10,12, 11 (level order)
Path 4: 1, 4, 12, 10, 11, 9, 8, 7, 6, 5, 3, 2 (this is a kind of reverse preorder – the root is visited first, then the rightmost child of the root, then its children starting from the rightmost, then the children, etc.)
Path 5: 12, 11, 10, 9, 8, 7, 6, 5, 4, 3, 2, 1 (this is a kind of reverse postorder –the children are visited before the parents in postorder fashion but the ordering is right to left rather than left to right and the root is visited last)

Fig. 4. Shows a selection of the possible paths through an XML document

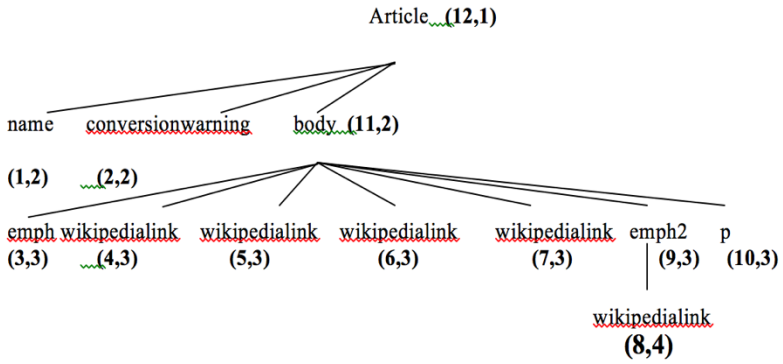


Fig. 5. An XML tree labelled with a postorder and level order numbering scheme

The work carried out by Wang and Meng's on the sequencing of tree structures for XML indexing [12] can be seen as relevant to the development of an XML transformation approach based on postorder and level order numbering schemes. Their transformation approach involves assigning a 'designator' to each attribute or element of the XML tree. They then map out a preorder sequence of the tree using the designators. Figure 6 shows an XML tree encoded with a designator for each node. Wang and Meng [12] outline how the encoded nodes in a preorder sequence give enough information to reconstruct the tree. For example, the preorder traversal of the tree in Fig. 6 would be

A, AN, AC, AB, ABE, ABW, ABW, ABW, ABW, ABM, ABMW, ABP. It is evident that by using this sequence alone the XML tree would be able to be reconstructed.

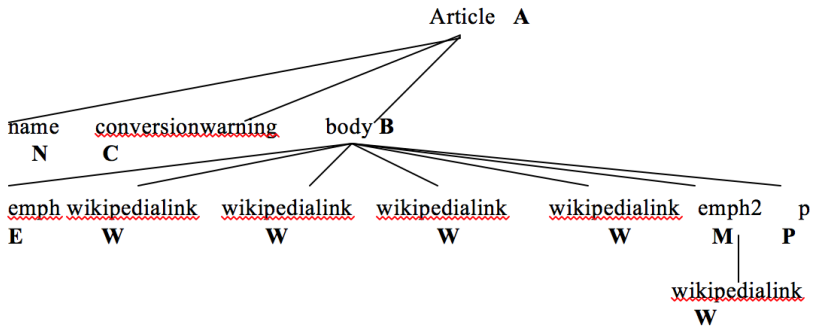


Fig. 6. An XML tree encoding using Wang and Meng's sequencing method

However, Wang and Meng [12] did not consider the possibility of postorder sequencing. The postorder sequencing of the tree in figure 6 would be - NA, CA, EBA, WBA, WBA, WBA, WBA, WEBA, MBA, PBA, BA, A. This sequence allows the tree to be reconstructed as it maps each node from last child to root. It differs from preorder sequencing in that the child nodes are visited before the parent. Postorder sequencing can be seen as a solution to the problem of XML tree generation from path data as it solves both the problems of ancestor-descendant relationships and of identical siblings. The postorder sequence maps out the exact position of the node in the tree determining which node is the parent and which is the child. Some numbering schemes may encounter problems when identical siblings were encountered such as in the tree in Fig. 6. However, the postorder sequencing approach allows there to be many identical siblings in an XML document and yet still be able to differentiate between them when reconstructing the tree as the order of the nodes in a sequence can supply the structural information required to differentiate between identical siblings. Therefore, postorder sequencing can be seen as a solution to the problem of tree reconstruction. This approach can be used to transform XML data into path data for data mining purposes. Once the association, classification or clustering techniques have been carried out, the order of the nodes in the sequence allow for the generation of an XML tree.

Postorder sequencing can be seen as furthering the idea of numbering XML trees with postorder and level order numbers. However, it has the advantage of only having to label the tree with one designator, which, when the sequence path is written out, conveys all the necessary information to reconstruct the tree from the sequence. Postorder sequencing allows all of the child nodes to be collected before the parent node which I believe to be an effective way of grouping XML data.

Postorder sequencing can be considered a novel approach to XML transformations as there is a distinct gap in research when it comes to postorder traversal of XML documents. The majority of indexing or numbering schemes for XML data are based around preorder traversal. e.g., ViST [13], XISS [1], Wang and Meng's Sequencing of Tree

Structures for XML Indexing [12], and Tulder's Modified Preorder Tree Traversal [14]. Therefore, there is a need for an evaluation of a postorder traversal approach for XML data in order to determine its usefulness. From my own experiences to date, I consider the postorder sequencing approach to XML transformations to be extremely useful and efficient in transforming XML data as to enable the output to be used to generate trees.

4.3 A Five Step Approach to XML Transformations

We propose a 5 step approach to the transformation of XML data into path data: 1) examine XML document and construct a Document Component Model; 2) construct a tree structure from the XML document; 3) construct DTD for the XML document; 4) label the tree with the designators; 5) transform the XML document into path data using the postorder sequencing approach.

Firstly, the XML document is examined and a Document Component Model is constructed. This is a document that lists all the components of the XML document and acts as a kind of mapping between the XML file and the transformation format. A hierarchical diagram of the XML tree is then constructed which provides a map through the XML document and allows a DTD to be constructed from this which is the third step in the approach. The tree can then be labelled with the node designators which aids the final step of transforming the XML document by traversing through the tree using the postorder sequencing method. This 5 step approach can be seen as encompassing several of the principals of the 'Document Engineering Approach' as outlined by Glushko and McGrath [15].

5 Summary

As XML has become arguably the most popular method of data exchange on the internet, it has become necessary to develop ways of transforming it into other formats such as web pages, databases, and knowledge bases etc. The ability of XML to be transformed into other formats becomes important in an age when data mining techniques such as association, clustering and classification gain popularity. As such techniques are difficult to carry out on XML data it becomes necessary to transform it into path data. One particular approach to the XML transformations involves traversing the tree structure of XML data; and while this proves to be a very effective approach to the transformation of XML data, the majority of tree traversal approaches do not allow an XML tree to be reconstructed from the traversal path. This is mainly because the ancestor descendant relationship is not able to be determined, and further problems may arise when identical siblings are present. Our study discovered that one way of enabling tree reconstruction from path data was through a numbering scheme that allowed both the horizontal position and the depth of each node in the tree to be identified. Postorder sequencing enables this information to be determined and allows an XML tree to be generated from the sequence path. Postorder Sequencing can be seen as an effective and efficient approach to transforming XML data that allows the XML tree to be gen-

erated from the path data. The 5 step approach to XML transformations that was implemented proved to be an effective way of ensuring that XML documents are adequately prepared for transformation. It can also be concluded that transformation approaches based on postorder numbering schemes can be seen as being effective in aiding the transformation of XML data.

Acknowledgement

This work is partially supported by the project funded by the National Natural Science Foundation of China (Grant No. 41374077).

References

1. Li, Quanzhong and Moon, Bongki.: Indexing and Querying XML for Regular Path Expressions. In: Proceedings of the 27th International Conference on Very Large Data Bases. Roma, Italy (2001)
2. Edler, Johann and Strametz, Walter.: Composition of XML Transformations. Electronic Commerce and Web Technologies: Second International Conference, EC-Web. Munich, Germany, September 4-6, 2001
3. Robie, Jonathan.: The Tree Structure of XML Queries. Software AG (1999)
4. Rao, Praveen and Moon, Bongki.: PRIX: Indexing and Querying XML Using Prufer Sequences. The 20th International Conference on Data Engineering. Boston, USA, March 2004
5. Miniaoui,S. and Wentland Forte, M.: Data Mining: From Trees to Strings (2004) available online at http://ali2.unil.ch/articles/miniaoui_ICICIS05.pdf
6. Kural, Murat.: Tree Traversal and Word Order. In Linguistic Inquiry Vol 36, Number 3, Summer 2005, P 367-387
7. Rutgers School of Computer Science Website http://www.cs.rutgers.edu/~kaplan/503/handouts/btreconNgentrees.html
8. Pokornyy, Jaroslav.: XML Documents Searching and Indexing. Available from Grant Agency of the Czech Republic – www.ksi.mff.cuni.cz/disg/grant/0912.html
9. Toman, Kamil.: Storing and Indexing XML Data. 2005 Available online at http://www.ksi.mff.cuni.cz/publications?target=file&field=File&id=2540183668933389818
10. Zhang, Wansong., Lui, Daxin., and Li, Jian.: An Encoding Scheme for Indexing XML Data. In IEEE International Conference on e-Technology, e-Commerce and e-Service. 28-31 March 2004 Page(s): 525 - 528
11. Xing, Guangming.: Indexing XML Data Using Extended Order and Path Index. ACM (2002)
12. Wang, H and Meng, X.: On the Sequencing of tree structures for XML Indexing. In Proceedings of the 21st Conference on Data Engineering (2005)
13. Wang, H; Park, S; Fan, W and Yu, P.S.: ViST: A Dynamic Indexing Method for Querying XML Data by Tree Structures. In Proceedings of the 2003 ACM SIGMOD international conference (2003)
14. Tulder, Gijs Van.: Storing hierarchical data in a database. (2003) Available online at www.sitepoint.com/print/hierarchical-data-database
15. Glushko, Robert and McGrath, Tim.: Document Engineering. MIT Press. Cambridge, Massachusetts (2005)