

# A Path Clustering based on Structural Similarity in XML

Bum-Suk Lee, Byung-Yeon Hwang

Dept. of Computer Engineering, The Catholic University of Korea

Yokgok2 Dong, Bucheon, 420-743, Korea.

leez79@gmail.com byhwang@catholic.ac.kr

## ABSTRACT

Since the XML standard was introduced by the W3C in 1998, documents that have been written in the XML format have been gradually increasing. Accordingly, several systems have been developed in order to efficiently manage and retrieve massive XML documents. The path bitmap indexing system is a representative system for this field of research. The path bitmap indexing system has the advantage of a higher retrieval speed in not only exact matched path searching but also similar path searching. However, the similarity calculation algorithm of this system has a few particular problems. Consequently, it sometimes cannot calculate the similarity even though the two paths have extremely similar relationships; further, it results in an increment in the number of meaningless clusters. In this paper, we have proposed a novel method that calculates the similarity between the paths in order to solve these problems. The proposed system yields a stable result for clustering, and it obtains a high score in clustering precision during a performance evaluation against LH06.

**Keywords** : XML, Indexing system, Similar path, Clustering

## 1. Introduction

XML[1] can be used to easily define the structure of data and is useful in B2B and B2C. Recently, massive XML documents have been circulated through various Web applications that support XML, such as RSS (RDF site summary). However, the flexibility of XML enables individuals to redefine or modify the structure of XML; consequently, several XML documents were formed with a diverse structure. Accordingly, the development of a system that can manage massive XML documents is necessary. The BitCube [2,3] system is a related research that uses a

three-dimensional bitmap index. It generates a three-dimensional bitmap index having the axes of the document name  $d$ , path  $p$ , and word  $w$  of the XML document. Further, if  $(d, p, w)$  exists, then the bit has a value of 1; otherwise, its value is 0. It performs the clustering of documents according to the similarity with the existence of path  $p$ . This system exhibited a good performance in an evaluation test with other systems [4,5]. However, it calculates a totally different path even when a single node (element or attribute) is added to the existing path. The path bitmap indexing system (LH06) [6] is a similar-path clustering system that can solve the abovementioned problem. LH06 proposes an equation for calculating the path construction similarity, and it performs the clustering of similar paths. The system solves the problem of BitCube; however, it sometimes cannot calculate the similarity when the node order is randomly shuffled.

In this paper, we propose a novel method to calculate the similarity for solving the abovementioned problem. The proposed method defines the insertion, deletion, and replacement operations to yield two similar paths. It calculates the distance of the two paths by using their operation costs. The distance value represents the similarity between the two paths: it has a value of 1 if they are the same paths; otherwise, its value is 0. The proposed system yields a stable result for clustering, and it obtains a high score in clustering precision during a performance evaluation against LH06.

This paper has been organized as follows. Section 2 introduces LH06 as a related work and discusses its problems. Section 3 defines the proposed method for calculating the path similarity and describes a system that has been built according to an equation. Section 4 provides the proof that the proposed system has a better performance with regard to clustering than LH06 by conducting a performance evaluation. Finally, section 5 concludes the paper and discusses the future works.

## 2. Related works

### 2.1 Path construction similarity

Path  $p$  is the basic unit to generate an index in the BitCube system—the three-dimensional bitmap indexing system used for retrieving XML documents. When the structure of a document is changed, the paths of the document are determined to be entirely different from the earlier determined paths. Therefore, BitCube cannot retrieve similar paths and it results in a rapid increment in the size of the bitmap index for the new paths. To solve this problem, LH06 proposes the following two definitions:

**Definition 2.1:** The path construction similarity is defined as an average of the node values in the path that has a greater number of nodes. The path construction similarity ( $P.C.Sim(P_1, P_2)$ ) between the paths  $P_1$  and  $P_2$  can be defined as follows:

$$P.C.Sim(P_1, P_2) = \frac{\sum_{i=1}^{NodeNum(P_2)} NodeValue(P_2, N_i)}{NodeNum(P_2)}$$

**Definition 2.2:** The value of node  $N$  that exists in path  $p$  ( $NodeValue(P, N)$ ) can be defined as follows. Here,  $d$  denotes the number of nodes to be inserted in  $P_2$ . In other words, it is the number of unmatched nodes.

$$NodeValue(P, N) = \frac{1}{2^d}$$

For instance, in Figure 1, suppose the path A.B.C is the criterion path and an arbitrary path comprising A.D.E.B.F.C is the target path to be compared. The values of nodes A, B, and C are 1s since they match the nodes in the criterion path. The values of nodes D, E, and F are 0.25, 0.25, and 0.5, respectively, according to definition 2.2. As a result, the path construction similarity is  $(1+0.25+0.25+1+0.5+1)/6 \doteq 0.67$ .

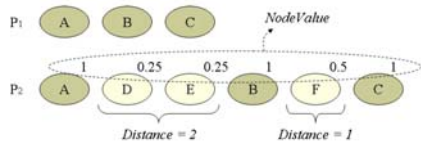


Figure 1: Calculation of the path construction similarity

### 2.2 Problems in LH06

The equation for calculating the path construction similarity suffers from the following problems. (1) The calculation becomes impossible when the two paths have different node constructions. Figure 2-(a) shows an example. Node  $F$  of  $P_2$  cannot have a  $NodeValue$  because the equation attempts to determine node  $E$  after node  $F$  of  $P_2$ . (2) When two paths have the same nodes with a different order, such as that in Figure 2-(b), the equation cannot be used to obtain the

$NodeValue$  of node  $D$  of  $P_2$  by using the procedure described in (1). (3) Additionally, because the equation is based on the distance of the inserted node, the two target paths  $P_{2-1}$  and  $P_{2-2}$  shown in Figure 2-(c) have different similarities with respect to the criterion path  $P_1$  (0.7 and 0.8, respectively).

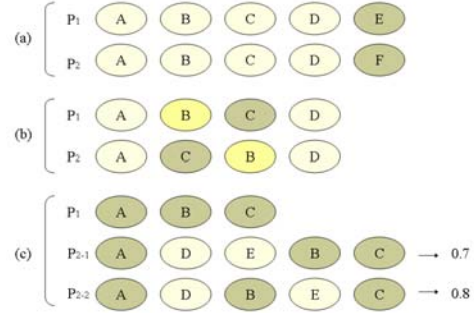


Figure 2: Examples of problems

## 3. System for clustering similar paths

### 3.1 Calculating the path similarity

This section proposes a novel method for calculating the path similarity to nullify the problems in the existing equation. Consequently, we define three operations, namely, insert, delete, and replace.

**Definition 3.1:**  $insert(x)$  is an operation that inserts a new node  $x$  into the path; the cost of this operation is 1.

$$Cost(insert(x)) = 1$$

**Definition 3.2:**  $delete(x)$  is an operation that deletes an existing node  $x$  from a path; the cost of this operation is also 1, similar to that in Definition 3.1.

$$Cost(delete(x)) = 1$$

**Definition 3.3:**  $replace(x,y)$  replaces node  $x$  with node  $y$  in the path. The cost of this operation is 2 because it has the same performance cost as the total cost of  $delete(x)$  and  $insert(y)$  operations.

$$Cost(replace(x,y)) = Cost(delete(x) + insert(y)) = 2$$

We can calculate the cost of an arbitrary path transformation to another path having a particular form by using the abovementioned definitions. The similarity between the two paths  $S(P_1, P_2)$  can be defined by using the minimum cost  $Min.Cost(P_1, P_2)$  as follows.

**Definition 3.4:** The similarity value is 1 when  $P_1$  is exactly the same as  $P_2$ ; otherwise, its value is 0. The similarity  $S(P_1, P_2)$  between  $P_1$  and  $P_2$  is defined as follows:

$$S(P_1, P_2) = 1 - \frac{Min.Cost(P_1, P_2)}{NodeNum(P_1) + NodeNum(P_2)}$$

Figure 3 shows an example to calculate the similarity value. Assume that  $P_1$  comprises the node order A.B.E.C.D and  $P_2$ ,

A.B.C.D.F. The two operations  $delete(E)$  and  $insert(F)$  are needed to transform  $P_1$  to  $P_2$ . Thus, the minimum cost of these operations is 2 and the total number of nodes in the two paths is 10. The similarity between the two paths can be calculated to be 0.8 according to Definition 3.4.

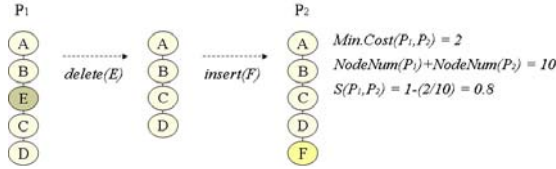


Figure 3: Similarity between the paths  $P_1$  and  $P_2$

The proposed method can be used when LH06 cannot be used to calculate the similarity value, such as that in Figure 2. The case in Figure 2-(a) is that the two paths have a different node construction. As a result, a similarity value of 0.8 is obtained by using the proposed method; however, the value yielded by LH06 is 0 since it cannot calculate the similarity. Figure 2-(b) shows a case in which two paths have a different node order; the similarity is calculated as 0.75 by using the proposed method. Figure 2-(c) shows a particular case in which LH06 yields different values of similarities because its equation is based on the distance of the nodes. By using the proposed method, the similarity values are the same—0.75.

### 3.2 Implementation of clustering system

This section introduces an indexing system of XML documents with an index cluster for similar paths using the proposed method in order to calculate the path similarity. Figure 4 shows the system architecture of the clustering of similar paths.

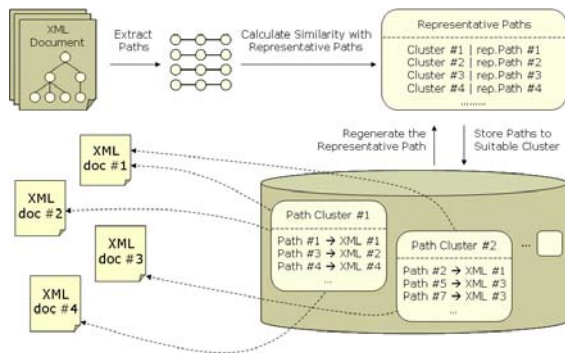


Figure 4: Architecture of the path clustering system

First, when the XML document is inserted for storing, the system extracts the paths and compares the extracted paths with the representative paths of clusters to calculate the similarity value. The paths are stored in a similar cluster or a

new cluster is created if no similar cluster exists. A new representative path is created after the new path is stored in the cluster. The paths in the clusters maintain links to their own XML document.

Figure 5 shows the algorithm of the clustering process (shown in Figure 4). When a new document  $d$  is inserted, the system extracts the paths and calculates the similarity value between the paths and the representative paths of the clusters. If no existing clusters have the same similarity value or a value greater than the threshold value, the system creates a new cluster for that path. On the other hand, if a cluster has a similarity greater than the threshold value, the path is stored in this cluster. Subsequently, the system regenerates the representative path of the created cluster or updated cluster. The proposed system always provides an adaptive representative path for each cluster.

```

Algorithm clustering( $d$ )
for (int  $i = 0$ ; path[ $i$ ] != null;  $i++$ ){
  if (numberOfCluster = 0){
    cid = createNewCluster(path[ $i$ ]);
  }else{
    (sim, cid) = getClusterSim(path[ $i$ ]);
    if (sim >= threshold){
      insertPath(cid, path[ $i$ ]);
    }else{
      cid = createNewCluster(path[ $i$ ]);
    }
  }
  repPath = getRepPath(cid);
  updateRepPathTable(cid, repPath);
}

```

Figure 5: Path clustering algorithm

## 4. Performance evaluation

This section describes the experimental results which compare the path clustering system that is implemented with proposed method in this paper to the existing method (LH06). The system has been implemented by using Java 1.4.2 for performance evaluation. The system environment for our experiment comprised a PC with a Pentium 1.7 GHz CPU, 1024 MB RAM, and Windows 2000 Server platform. We use a set of real XML documents with the NewsML [7] schema. There are 1670 paths from 80 documents. The experiment comprises two sections. The evaluation was performed to evaluate the precision of the clustering result as compared to that of the existing method.

### 4.1 Comparing the numbers of clusters

We compared the numbers of created clusters. Figure 6 shows the result of the created clusters by changing the

threshold for the existing and proposed methods. The existing method creates 1.12 times or 2 times the number of clusters, which are changed according to the threshold value, than those created by using the proposed method.

Threshold	LH06	Proposed method
0.2	15	7
0.4	18	16
0.6	262	120
0.8	429	302

Figure 6: Comparison of the numbers of created clusters

#### 4.2 Clustering precision

To evaluate the clustering results, we used two metrics, namely, *precision*  $P$  and *recall*  $R$ , which are popular in information retrieval [8]. Let (a)  $a_i$  be the number of XML documents in the extracted cluster  $C_i$  that were indeed members of that cluster (correctly clustered), (b)  $b_i$  be the number of XML documents in  $C_i$  that were not members of that cluster (misclustered), and (c)  $c_i$  be the number of XML documents not in  $C_i$ , although they should be  $C_i$ 's members. Then,  $P = \sum a_i / (\sum a_i + \sum b_i)$  and  $R = \sum a_i / (\sum a_i + \sum c_i)$  [9]. This experiment is performed for the threshold values of 0.2 and 0.4. Figure 7 shows the  $P$  and  $R$  values obtained using the two methods.

While we regard  $P$  and  $R$  of proposed method is 1.0, the evaluation results reveal that the existing method yields respectively lower values. Thus, by using the proposed method, we can obtain a distinct improvement in the performance as compared to the existing method.

Threshold	Measured	LH06	Proposed method
0.2	<i>Precision</i>	0.779	1.0
	<i>Recall</i>	0.859	1.0
0.4	<i>Precision</i>	0.579	1.0
	<i>Recall</i>	0.346	1.0

Figure 7: Comparison of the clustering precision values

## 5. Conclusion

This study presents a novel method for calculating path similarities in order to improve some problems of the existing method used for clustering the paths of XML documents. To compare a criterion path  $P_1$  and a target path  $P_2$ , the existing method is limited by the following restrictions: (1)  $P_1$  is always shorter than (or the same as)  $P_2$ . (2) All the nodes of  $P_1$  have to be included in  $P_2$ . (3) The shared nodes in  $P_1$  and  $P_2$  should be located with the same

ordering. Additionally, another problem is that the problem yields different similarities in some path pairs, for example,  $P_1$ - $P_{2,1}$  and  $P_1$ - $P_{2,2}$ , where  $P_{2,1}$  and  $P_{2,2}$  have a different node ordering.

The proposed method is based on the cost calculation of the path transformation. It solves the problems of the existing method. In order to experimentally validate our proposals, we implemented a test-bed using clustering methods and datasets. We performed an evaluation using real datasets and compared the two methods. Our results showed that (1) the experimental results obtained from the proposed method showed a stable clustering quality than the existing method and (2) the proposed method yielded an improved performance as compared to the existing method and also solved the abovementioned problems of the existing method. In conclusion, this research demonstrates that the clustering method can be successfully employed for grouping the paths having a similar structure; for the performance evaluation, we performed a comparison of the proposed and existing methods. As a future work, the calculation of path similarities might be expanded such that it can be applied for calculating the similarity values of XML documents, and a research on the extraction method for representative paths is needed to improve the extraction time.

## References

- [1] <http://www.w3.org/TR/2000/REC-xml-20001006>
- [2] J. P. Yoon, V. Raghavan, V. Chakilam, and L. Kerschberg, "BitCube: A Three-Dimensional Bitmap Indexing for XML Documents," *Journal of Intelligent Information System*, Vol.17, pp. 241-254, 2001.
- [3] J. Yoon, V. Raghavan, and V. Chakilam, "BitCube: Clustering and Statistical Analysis for XML Documents," In *Proc. of the 13th Int'l Conf. on Scientific and Statistical Database Management*, Virginia, Jul. 2001.
- [4] D. Egnor and R. Lord, "XYZFind: Structured Searching in Context with XML," In *Proc. of ACM SIGIR Workshop*, Athens, Greece, 2000.
- [5] XQEngine. <http://www.fatdog.com>
- [6] J. M. Lee and B. Y. Hwang, "Path Bitmap Indexing for Retrieval of XML Documents," *Lecture Notes in Computer Science*, Vol.3885, Springer-Verlag, Apr. 2006.
- [7] NewsML, <http://www.newsml.org>
- [8] C. J. van Rijisbergen. "Information Retrieval," Butterworths, London, 1979.
- [9] T. Dalamagas, T. Cheng, K. J. Winkel, and T. Sellis, "A Methodology for Clustering XML Documents by Structure," *Information Systems*, Vol. 31, Issue 3, Elsevier Science Ltd., pp. 187-228, May, 2006.