International Journal of Electrical and Computer Engineering (IJECE)
Vol. 5, No. 6, December 2015, pp. 1472~1479
ISSN: 2088-8708

An Optimistic Approach for Clustering Multi-version XML Documents Using Compressed Delta

Vijay Sonawane, D. Rajeswara Rao
Department of CSE, K.L.University, Green Fields, Guntur, Andhra Pradesh

Article Info

Article history:

Received May 20, 2015
Revised Aug 12, 2015
Accepted Aug 28, 2015

Keyword:

Clustering
Compressed Delta
Homomorphic
Multi-version
XML

ABSTRACT

Today with Standardization of XML as an information exchange over web, huge amount of information is formatted in the XML document. XML documents are huge in size. The amount of information that has to be transmitted, processed, stored, and queried is often larger than that of other data formats. Also in real world applications XML documents are dynamic in nature. The versatile applicability of XML documents in different fields of information maintenance and management is increasing the demand to store different versions of XML documents with time. However, storage of all versions of an XML document may introduce the redundancy. Self describing nature of XML creates the problem of verbosity, in result documents are in huge size. This paper proposes optimistic approach to Re-cluster multi-version XML documents which change in time by reassessing distance between them by using knowledge from initial clustering solution and changes stored in compressed delta. Evolving size of XML document is reduced by applying homomorphic compression before clustering them which retains its original structure. Compressed delta stores the changes responsible for document versions, without decompressing them. Test results shows that our approach performs much better than using full pair-wise document comparison.

Corresponding Author:

Vijay Sonawane,
Department of CSE, K.L.University,
Green Fields, Guntur, Andhra Pradesh, 522 502.
Email: vijaysonawane11@gmail.com

1. INTRODUCTION

W3c established the extensible mark-up language (XML-1.0) in 1988 and XML has become the defacto standard for data representation and exchange of information on the internet [1]. Today it is also important to know the business values of information present on-line. Information available on-line is not only useful for individual user but also to business organizations mostly for decision making purpose. XML offers many features of business functions such as content integration, intelligence and salvage. It is very important to maintain those documents properly and know how to use efficiently information in it.Clustering is data mining task always and always performed using distance measurement on dense regions in dataset [2]. XML documents are semi-structured in nature and it is encode by different textual format [3]. Hence many traditional approaches failed to handle clustering of XML documents. Several clustering approaches are discussed in [4]-[6].

Alternative option to handle XML data is snapshot XML document but in real world content of XML documents is dynamic in nature and changes in it are limitless and not predictable [7]. Every time change in original document is completely treated as completely new XML (version) document rather than considering the degree of change in it. Dynamic XML documents create the demand for multi-version support as it is applicable in many fields of information maintenance and management [8]. So it is necessary to store different versions of XML documents with time. Storage of all the versions of an XML document

increases the redundancy and make searching and querying harder on growing documents.The self-describing feature of XML provides immense flexibility but it introduces the main problem of verbosity which results in huge document sizes, which creates difficulty same as redundancy [9].

This paper introduces an optimistic approach to find new clustering solution of multi-version XML documents which are dynamic in nature. In this approach we do not treat each new XML document as completely new document (version) rather we considers amount of the documents affected. To limit the size and reduce the storage space each document is compressed using designed homomorphic compression scheme (which retain structure like original document) and compressed delta is used to record only affected part of the documents (without decompressing it). Finally up-to-date clustering solution is obtained by using previous know distances calculated during initial clustering and knowledge recorded in compressed delta.

2. WHY XML COMPRESSION?

The self-describing property of XML (schema is repeated for each record) provides flexibility to XML but it also introduces the major problem of verbosity of XML documents which results in huge document size. Large document size affect on storage space, network bandwidth used for data exchange and main memory required for processing. It makes documents searching, querying, and clustering harder. Homomorphic compression scheme retain the original hierarchical structure of the document and compressed format can be accessed and parsed in the same way of the original one [9]. If XML document changes its structure or content, then compressed delta is used to records the changes between two versions without decompressing it. Figure 1 Shows documents homomorphic compressed view of our compressor, it avoids element/attribute values repetition by replacing it with unique character.

```
<Employees>                    <T0>
 <emp type="M">                 <T1 A0="M">
  <ID>1</ID>                     <T2>V0</T2>
  <Name>ABC</Name>               <T3>V1</T3>
  <City>Mumbai</City>            <T4>V2</T4>
  <Dept>ECE</Dept>               <T5>V3</T5>
  <Salary>13092</Salary>         <T6>V4</T6>
 </emp>                         </T1>
 <emp type="M1">                <T1 A0="M1">
  <ID>2</ID>                     <T2>V5</T2>
  <Name>BCD</Name>               <T3>V6</T3>
  <City>Hydrabad</City>          <T4>V7</T4>
  <Dept>ECE</Dept>               <T5>V3</T5>
  <Salary>15790</Salary>         <T6>V9</T6>
 </emp>                         </T1>
<Employees>                    </T0>
```

Figure 1. Homomorphic compressed view

3. LITERATURE SURVEY

In this section we discuss existing work in the area of change detection and clustering of multi-version XML documents, stressing the fact that any of existing work does not deal time efficiently with reassessing clusters of multi-version XML documents by using notion of compressed delta [10].

Document version detection and management is important in various applications such as software plagiarism detection, web document searching and ranking. To detect the versions, similarities between various files need to be analysed, for this selection of similarity function and threshold value are important. The content and structure similarity as well as application requirement are important factor in designing similarity function. Various schemes have been proposed for detecting changes in multi-version XML documents based on the *diff* algorithm [12]-[15]. Documents textual content [16], Structure of document [17]-[20] and Document Classification [21], [22].

To do the comparative analysis of change detection schemes need the analysis of various parameters such as change detection between two versions of an XML document, use of relational, delta, or object-referencing approach for change detection,support for ordered XML documents, scalability, supported file size, and representation of unchanged parts [8].

Some change detection schemes explicitly shows the user the changed part of the documents [23]-[25], while other may not show [7], [26]-[30]. Specified schemes should be scalable enough to detect the

changes in XML documents. The storage of intermediate complete versions of XML documents improves the efficiency and space complexity as the required version can be created by using the appropriate intermediate complete version. Query processing becomes faster while a system stores the intermediate complete versions because there is no need to reconstruct the intermediate versions run time.

To cluster XML documents, researcher in recent decade years proposed various schemes. In [31] author proposed methods to discover frequently changing structure (FCS) from a sequence of versions of dynamic XML document, then proposed method named CDX to cluster dynamic XML document collection by using FCSes. In [32] author proposed SemXClust framework, and divide XML document into tree tuple set, then proposed XK-means algorithm and XFIHC algorithm to cluster XML document by using WordNet. A novel Weighted Element Tree Model (WETM) is proposed in [33] for measuring the structural similarity of XML documents, and the WETM model enhances the expression ability of structural information of sub trees. Author in [34] proposed a framework for clustering XML documents by structure, they model the XML documents as rooted ordered labelled trees, then studied the usage of structural distance metrics in hierarchical clustering algorithms to detect groups of structurally similar XML documents. Author in [35] proposed an approach for detecting structural similarity between XML documents which significantly differs from standard methods based on graph-matching algorithms, and allows a significant reduction of the required computation costs.

Approaches for clustering XML documents are useful tools for XML stream mining [36], gene team [37] and web service retrieval [38]. A different technique for clustering series of heterogeneous XML documents proposed by [39]. This considers series (streams) of XML documents and calculates the similarity between each incoming XML document and the existing clusters by using the concept of level structure. Similarity is determined at cluster level rather than pair-wise for individual documents in the clusters.

4. PROPOSED SYSTEM OVERVIEW

Proposed approach overview to find up-to-date clustering solution of multi-version XML documents which changes in time is shown in Figure 2.

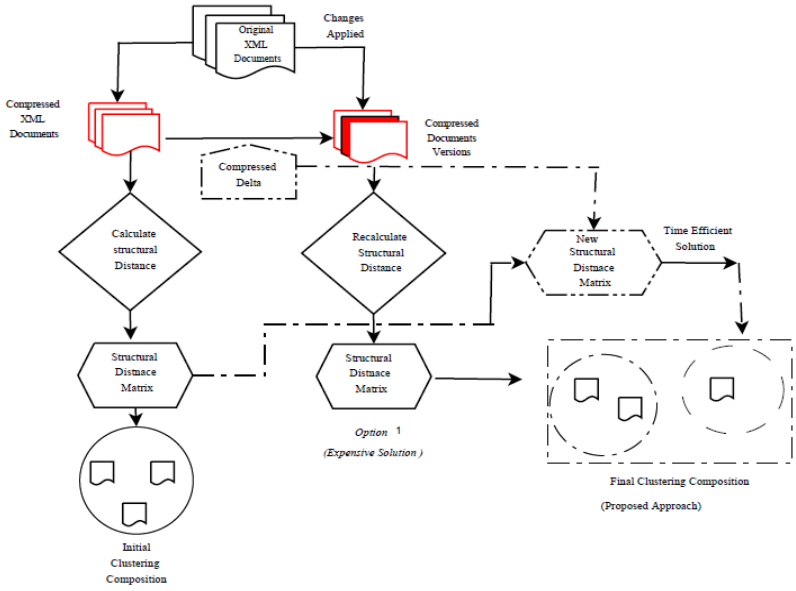

Figure 2. Overview of the Proposed Approach

Given set of S XML documents, its clustering composition S can be represented as graph of fully connected edges with $S * (S — 1)=2$ with total number of weighted edges, to get single link clusters of level L, the edges with weight $w \geq L$ have to pruned, resulting connected edges give result cluster [5]. If similarity between two XML documents is used as measure to find clustering solution then weight of the edges connecting documents symbolize the distance between them.

Distance: Distance between two XML documents can be defined by edit operations *Op (insert, delete, update)* with minimum cost that transform one document into other.

dist(Doc1,Doc2) = minimum((dist(Doc1→Doc2),dist(Doc2→Doc1)))

If distance between two documents is smaller, it indicates the higher similarity between them. Hence when total cost operation between two document is equal to zero then these document are said to similar. When any document in initial clustering solution changes then its distance with rest of the documents in the cluster also changes. Depending on the change (i.e number and type) responsible for document version will decide residence of the version (new document) in clustering solution. When new version appears it can be part of original cluster or it might get contained by another one or it form its own cluster. To get up-to-date clustering solution it is required to find the new distances between all XML documents pairs.

To get up-to-date clustering solution after each set of changes applied(when the documents in the initial cluster changes their distances), comparison between all the documents pair is not cost effective way because most of time new version may carries small amount of change or may not modified at all. Option 1 shown in diagram redundantly calculates distances between each pair of documents by making full comparison between them. This option incurs more cost as it does not consider degree of changes in the documents, hence most of the operations are unnecessarily repeated.

Our proposed approach is time efficient solution to get current clustering solution by reassessing pairwise distances between the documents. Compressed delta record all the changes responsible for multiple version of the documents without decompressing them. To get current clustering solution, it makes the use of known distances between documents pair before changes applied and set of changes recorded in the compressed delta. It has following advantages; 1) Compression scheme used retains documents in original structure with reduced size, Compressed documents are accessed and parsed in the same way of the original format. 2) Compressed delta records changes applied on documents without decompressing it,helpful for change detection. 3) It considers the degree of changes applied on the documents before finding new clustering solution.

4.1. Working

1) Document Compression - Given set of XML documents are compressed using homomorphic compression scheme, it retains structure of original documents.

2) Finding Initial Clustering Solution - This step gives initial clustering solutions by using distance based clustering algorithm and records distance matrix between all the pairs of the XML documents with direction of minimum cost (this step is executed only once).

3) Obtaining documents version - Here document versions are created (using our version creation program) by applying sort of changes on compressed input XML documents. Compressed delta records all the set of changes between the initial clustering run and current timestamps without using decompression.

4) Obtaining final Clustering Solution - This step is repetitively executed whenever versions appears or current clustering solution is required. It recalculates the new distances between all pairs by using knowledge about changes from compressed delta and distance matrix recorded during initial clustering run.

In this, critical part is finding new distances based on data in compressed delta and previous know distances, in next section we present technique to achieve this.

5. DISTANCE MEASUREMENT

Hierarchical structure of XML easily allows performing operations on the nodes of the documents. Compression scheme we used here retains original document structure. In multi-version XML documents, new versions are obtained by applying insert, update and delete operations in combination on document version nodes and sum of these operations are stored in compressed delta.

Compressed Delta(Cδ) - Given dynamic XML document Doc1 with its version *Doc*1´, compressed delta records the changes from one state of document to another. It consists of a set operations *Op(insert, delete, update)*, execution of it on one state of document Doc1 will return document in state *Doc*1´.

Cost of Compressed Delta(CCδ) - The cost of compressed delta is sum of operations cost *Op(insert, delete, update)* listed in the compressed delta.

Inverted Operation (O^{invert}) - Given dynamic XML document Doc1 with its version *Doc*1´, If an execution of operation *Op(insert, delete, update)* on document Doc1 returns its version *Doc*1´, then execution of inverted operation on version *Doc*1´ returns original document Doc1. i.e insert(A) is inverted operation of corresponding delete(A) and update (A→B) is inverted operation of corresponding update(B→A) where A, B are the nodes in an XML documents.

Given an initial clustering solution S containing Doc1 and Doc2 with dist(Doc1, Doc2) is distance between them and (Doc1→Doc2) is the minimum cost direction, if set of changes stored in compressed

delta (Cδ) transform one document into other, then new distance *dist´* between them can be defined by using following formula:

$$dist'(Doc1' \rightarrow \text{Doc2}) = (\text{dist}(\text{Doc1},\text{Doc2}) \cup C\delta\ (\text{Doc1}\ ,\ Doc2')) - \text{dist}(\text{Doc1},\text{Doc2}) \cap C\delta\ (\text{Doc1},Doc2')) \quad (1)$$

When Doc1 changes to *Doc*1´ then to find its new distance with Doc2, common set of operations that transforms Doc1 to *Doc*1´ with minimum distance need to be subtracted while calculating the distances between *Doc*1´ and Doc2 as they are equal in results, so only distinct operations need to considered.

$$dist'(\text{Doc1} \rightarrow Doc2') = (\text{dist}(\text{Doc1},\text{Doc2}) + C\delta\ (\ \text{Doc2} \rightarrow \text{Doc2'}) - \sum_{i=1}^{q} {}_{Q}i + Q_i^{invert} \quad (2)$$

Here O_i are the p operations from dist (Doc1, Doc2) which have consequent inverted operations Oi^{invert} in Cδ (Doc2, *Doc*2´),1≤i≤p. It means the set of operations which gives minimum distance between Doc1 and Doc2, and were subsequently inverted during Doc2 transformation into *Doc*2´ need to be subtracted when calculating the distance between Doc1 and *Doc*2´, as their combined effect is null, whereas only the distinct non-inverted operations need to be counted.

$$dist'(Doc1' \rightarrow Doc2') = [(\text{dist}(\text{Doc1},\text{Doc2}) \cup C\delta 1(\text{Doc1},Doc1')) - (\text{dist}(\text{Doc1},\text{Doc2}) \cap C\delta 1(\text{Doc1},Doc1')] + C\delta 2\ (\text{Doc2},Doc2') - \sum_{i=1}^{q} {}_{Q}i + Q_i^{invert} \quad (3)$$

Here O_i are q residual operations from dist(Doc1, Doc2) after removing repeated operations from Cδ1 which have consequent inverted operations Oiinvert in Cδ2 (Doc2, *Doc*2´),1≤i≤p. When both Doc1 and Doc2 have changed into its respective versions *Doc*1´ and *Doc*2´, above both formulas are applicable to find new distance *dist´*(*Doc*1´, *Doc*2´). First we add the cost of d and Cδ1 and purge the operations that are repeated in both d and Cδ1 and remove remaining operations those are inverted in Cδ2.

$$dist'(\text{Doc1},\text{Doc2}) = \text{d}(\text{Doc1},\text{Doc2}). \quad (4)$$

When both documents Doc1 and Doc2 do not change, then the new distance dist is the same with the old distance dist.

6. EXPERIMENTAL EVALUATION

Proposed approach is implemented using Java. To evaluate the performance of proposed approach we extracted documents of variable size from XML data repository with an average number of levels of 4 [40]. Initial clustering solution is obtained for compressed set of input XML documents by finding minimum distances between all the document pairs. Then distances between all document pair in the clustering solution along with the set of operations corresponding to each minimum distance is recorded. To get new documents version we applied different percentages of changes on compressed input XML documents. Main objective of the evaluation is to compare (after changes applied) time required to find new clustering (Re-clustering) solution of the documents with and without compression using full document comparision scheme (FDC) with our proposed approach.

Table 1. Experiment results

No of Documents	Documents Original Size (Kb)	Size after Compression (Kb)	Changes Applied to get Versions (%)	Clustering Time (millisecond)		
				Full Document Comparision(FDC)		Proposed Approach
				without compression	with compression	
10	50	35	10	756	396	90
25	100	70	20	2150	1548	334
50	500	350	50	4589	3225	964
100	1000	700	80	9947	5610	1465
150	1500	1040	100	14135	8256	2104

Experimental results are shown in Table 1. These results clearly demonstrate that applied compression scheme satisfactorily reduces the documents size. Last column shows proposed approach is time

efficient to find new clustering (by reassessing new distances) than using full pair-wise comparison between the documents.

Result charts are shown in Figure 3. Charts also clearly demonstarte that our porposed approach perfomr well even if number of documents and applied percentage of chages (to get new version) increased.

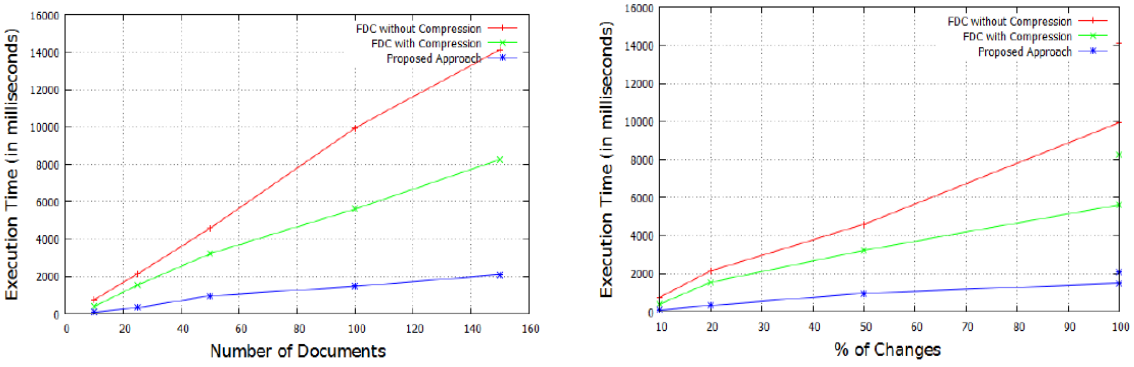

Figure 3. Experiment Result Chart

7. CONCLUSION

In this paper we have proposed an optimistic approach for clustering multi-version xml documents when distances between initially clustered documents have changed. After document version appears every time running of full pair-wise documents comparision on the entire modified document set is expensive solution as it incur redundant operations. Our proposed approach allows reassessing the pair-wise XML document distance by finding effect of the temporal changes on known distances between initially clustered documents.This proposed approach is both time and I/O effective, as homomorphic compression scheme is applied on input XML documents and the numbers of operations involved in reassessing the distance are highly reduced.

REFERENCES

[1] S. Abiteboul, P. Buneman, and D. Suciu, ”Data on the Web”, San Francisco: Morgan Kaufmann, 2000.
[2] Chen M. S., Han J., and Yu P., ”Data Mining: An Overview from Database Perspective”, *IEEE Transactions on Knowledge and Data Engineering,* Vol. 8, 866-883, 1996.
[3] Sonawane Vijay., D. R. Rao., ”XML Document Mining”, *Journal Research in Computer Engineering and Electronics,* Vol. 2, No. 2, pp. 1-4, 2013.
[4] Costa G., Manco G., Ortale R., and Tagarelli A., ”A tree-based Approach to Clustering XML documents by Structure”, *PAKDD 2004,* pp. 137-148, 2004.
[5] Dalamagas T., Cheng T., Winkel K. J., and Sellis T., ”Clustering XML documents by Structure”, *SETN 2004,* pp. 112-121, 2004.
[6] Shen, Y. and Wang, B., ”Clustering Schemaless XML documents”, Springer, pp. 767-784, 2003
[7] Dyreson C., and Grandi F., “Temporal XML”, *Database Systems*, pp. 3032-3035, 2009.
[8] Sidra F., Mansoor S., ”Temporal and multi-versioned XML documents: A survey”, *Information Processing and Management,* Vol. 50, No. 1, pp. 113-131, 2014.
[9] Sherif Sakr., ”XML compression techniques: A survey and comparison”, *Journal of Computer and System Science*, Vol. 75, No. 5, pp. 302-322, 2009.
[10] Vijay Sonawane, D. R. Rao, ”A Comparative Study: Change Detection and Querying Dynamic XML Documents”, *International Journal of Electrical and Computer Engineering (IJECE),* Vol. 5, No. 4, 2015.
[11] Samini S., Su-Cheng H., Poo Kuan H., ”Bridging XML and Relational Databases: An Effective Mapping Scheme based on Persistent”, *International Journal of Electrical and Computer Engineering (IJECE)*,Vol. 2, No. 2, pp. 239-246, 2012.
[12] Al-Khalifa S., Jagadish H. V., Koudas N., Patel J. M., Srivastava D., and Wu Y., ”Structural joins: a primitive for efcient XML query pattern matching”, *Proceedings of the eighteenth international conference on data engineering*, pp. 141-154, 2002.
[13] Rusu L. I., Rahayu W., Taniar., ”Storage techniques for multi-versioned XML documents”, *Proceedings of the thirteenth international conference on database systems for advance applications*, pp. 538-545, 2008.
[14] Saccol D de B., Edelweiss N., Galante R de M., Zaniolo C., ”XML version detection”, *Proceedings of the ACM symposium on document engineering*, pp. 7988, 2007.

[15] Wang Y., DeWitt D. J., and Cai J. Y., "X-Diff: an effective change detection algorithms for XML documents", *Proceedings of the nineteenth international conference on data engineering,* pp. 519-530, 2003.
[16] Baeza-Yates R., and Ribeiro-Neto B., "Modern information retrieval: The concepts and technology behind search", *ACM Press/Addison-Wesley,* 2011.
[17] Flesca S., and Pugliese A., "Fast detection of XML structural similarity", *IEEE Transactions on Knowledge and Data Engineering,* Vol. 17, No 2, pp. 160-175, 2005.
[18] Gao M., and Chen F., "Clustering XML Data Streams by Structure based on SlidingWindows and Exponential Histograms", *Proceedings of the international conference on advances in databases, knowledge, and data applications*, pp. 224-230, 2013.
[19] Wan X., Yang J., "Using proportional transportation similarity with learned element semantics for XML document clustering", *Proceedings of the fteenth international conference on world wide web*, pp. 961-962, 2006.
[20] Elham B. F., Hasan K., "Improving semantic clustering using with Ontology and rules", *International Journal of Electrical and Computer Engineering (IJECE),* Vol. 4, No. 1, pp. 7-15, 2014.
[21] Pon R. K., Crdenas A. F., Buttler D., Critchlow T., "iScore: Measuring the interestingness of articles in a limited user environment", *Proceedings of the IEEE symposium on computational intelligence and data mining*, pp. 354-361, 2007.
[22] Wang Y., Hodges J., and Tang B., "Classication of web documents using a naive bayesian method", *Proceedings of the fteenth IEEE international conference on tools with, articial intelligence*, pp. 560-564, 2003.
[23] Leonardi E., Bhowmick S. S., Madria S., "Xandy: Detecting changes on large unordered XML documents using relational databases", *Proceedings of the international conference on database systems for advanced applications*, pp. 711-723, 2005.
[24] Rusu L. I., Rahayu W., Taniar D., "Maintaining versions of dynamic XML documents", *Proceedings of the sixth international conference on web information, systems engineering*, pp. 536-543, 2005.
[25] Wuwongse V., Yoshikawa M., and Amagasa T., "Temporal versioning of XML documents", *Proceedings of the Seventh International conference on digital libraries: International collaboration and cross-fertilization*, pp. 419-428, 2004.
[26] Cavalieri F., "EXup: an engine for the evolution of XML schemas and associated documents", *Proceedings of the international conference on extending database technology*, pp. 110, 2010.
[27] Cavalieri F., Guerrini G., Mesiti M., and Oliboni B., "On the reduction of sequences of XML document and schema update operations", *Proceedings of the IEEE twenty seventh international conference on data engineering workshops*, pp. 77-86, 2011.
[28] Guerrini G., and Mesiti M., "X-Evolution: A comprehensive approach for XML schema evolution", *Proceedings of the international workshop on database and expert systems application*, pp. 251-255, 2008.
[29] Rosado L. A., Mrquez A. P., and Gil, J. M., "Managing branch versioning in versioned/temporal XML documents", *Proceedings of the international symposium on XML, database*, pp. 107-121, 2007.
[30] Zholudev V., Kohlhase M., "TNTBase: A versioned storage for XML", *Balisage: The Markup Conference,* 2009.
[31] Wei Li, Xiongfei Li, and Regen Te., "Cluster Dynamic XML Documents based on Frequently Changing Structures", *Advances in information Sciences and Service Sciences(AISS),* Vol. 4, No. 6, pp. 70-76, 2012.
[32] A. Tagarelli., and S. Greco., "Semantic clustering of XML documents", *ACM Transactions on Information Systems*, Vol. 28, No. 1, pp. 1-56, 2010.
[33] C. Y. Wang, X. J. Yuan, and H. Ning, *et al.,* "Similarity Evaluation of XML Documents Based on Weighted Element Tree Model", *In Advanced Data Mining and Applications,* Vol. 5678, R. Huang,et al., Eds., ed: Springer Berlin Heidelberg, pp. 680-687, 2009.
[34] T. Dalamagas, T Cheng, K. J. Winkel, *et al.,* "A methodology for clustering XML documents by structure", *Information Systems,* Vol. 31, No. 3, pp. 187-228, 2006.
[35] S. Flesca, G. Manco, E. Masciari, *et al.,* "Fast detection of XML structural similarity", *IEEE Transactions on Knowledge and Data Engineering,* Vol. 17, No. 2, pp. 160-175, 2005.
[36] M. X. Gao, W. J. Yao, and G. J. Mao, "Exponential histogram of cluster feature for XML stream", *Journal of Beijing University of Technology,* Vol. 37, pp. 1242-1248, 2011.
[37] B. Wang, and C. Lin, "Improved algorithms for finding gene teams and constructing gene team trees*", IEEE/ACM Transactions on Computational Biology and Bioinformatics,* Vol. 8, pp. 1258-1272, 2011.
[38] P. Plebani, and B. Pernici, "URBE: Web Service Retrieval Based on Similarity Evaluation", *IEEE Transactions on Knowledge and Data Engineering,* Vol. 21, No. 11, pp. 1629-1642, 2009.
[39] Nayak R., Xu S., "XCLS: A Fast and Effective Clustering Algorithm for Heterogeneous XML Documents", *Proceedings of the 10th Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining,* Singapore, pp. 3918, 2006.
[40] XML data repository, online at www.cs.washington.edu/research/projects/xmltk/xmldata.

BIOGRAPHIES OF AUTHORS

Vijay Sonawane obtained Bachelor Degree in Information Technology from North Maharashtra University (MS), India. Then he earned Master in Technology (Computer science) from Shivaji University, Kolhapur (MS), India. He is currently PhD Research scholar in computer science and engineering in K.L.University, Vijayawada. He is working as Assistant Professor in Sandip Institute of Technology and Research Centre, Nashik. His major area of interest is Data mining, Information retrieval, web information management. He published various papers in Journals and Conferences.

D.Rajeswara Rao, he is a Professor in department of CSE in K L University, Vijayawada. He is acting as heads of knowledge enginering research group and Research policy advisory committee in K.L.University. He is He had published various research papers at National and International Journals/Conferences in the area of data mining, soft computing.