ISSN:0975-9646

Ranking Of XML Documents by Using Adaptive Keyword Search

Varun Varma Sangaraju

Student of CSE
Sathyabama University
Chennai, India

Abstract— **The Ranking of XML documents can be performed by using Adaptive keyword search and Reverse indexing of the XML data within which critical metrics are weighed and assigned to the XML data. This proposed system can act as an upgrade to the existing XML keyword searching pattern. The search results obtained in LCA based systems are non-adjustable and some important features like compactness and size are missed. So this proposed system aims to overcome the disadvantages of LCA methods with adaptive system containing the metrics to make the system better.**

1. Introduction

XML keyword search is a user-friendly information retrieval technique from the XML files, which attracts many interests in the Data mining concept. Usually, all the search mechanisms are implemented for the HTML files to retrieve online content like Google, Yahoo etc., which are represented as web links. The search mechanism for the XML files is also a very important technique to retrieve the text content from the XML files required. As it is a difficult or sometimes almost impossible to identify users' intentions through the keywords. And it is still an issue in considering the structured or semi-structured XML data for searching.

Many valuable models are proposed to define the results of XML keyword search in which most important one is the Smallest Lowest Common Ancestor (SLCA) model [1] and its variants. MLCA[6] and GDMCT[3] also generates the same answers like the SLCA based methods, as the both models exhibit same results similar to SLCA. The result will not be accepted, when any two nodes in ELCA represent the same elementary data. Filtering is performed for such type of nodes and the left ELCA nodes are called as Valuable Lowest Common Ancestor (VLCA) nodes.

Let us consider an example to define the SLCA nodes. In Figure 1.1, the XML data is structured data which is partitioned based on the XML tags. Consider the keywords {XML, Varun} issued on the XML document in the below figure, and obviously the method can find two SLCA nodes {paper, paper} and the sub trees in them will be declared as the final results.

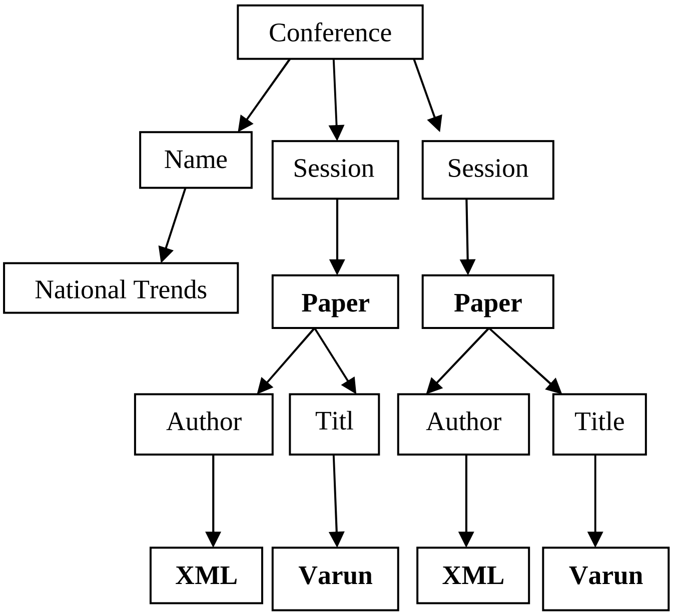

Fig. 1.1: XML document representation using SLCA

2. Related Work

The Searching of the XML documents from the database and also ranking the documents in the order of the search word repetition is also very important. This XML keyword search is implemented in tree data model and in the digraph data model.

Searching in XML Tree Databases:
Most of the LCA-based models are basically used for the searching of the keyword in XML Tree databases. The SLCA model is used to search the node where all the keywords are present. So, the keyword existing in the other nodes separately doesn't turn up and hence SLCA is not efficient. The ELCA process tries to remove the SLCA nodes present in it and halts the process if none of SLCA nodes are present. XSeek[7] model approaches semantic inference in improving keyword match pattern.

Searching in XML Graph Databases:
The Graph databases had been searched by using XKeyword[11], which is created on a relational database. BLINKS[12] is completely another type of approach which partitions the graph containing data into many blocks to store the memory and process it separately. EASE[14] is a

model which also works on the unstructured data as graphs, thus deals with huge amount of heterogeneous data.

Ranking of Searched Results:

XRank[2] is a ranking model which is implemented to rank all the LCA results from page rank to XML element level. RACE[15] is a mechanism used to rank the trees through structural and textual similarity. The structure and content are taken into consideration and they are ranked in a mechanism called the XSEarch[16] Ranker. XReal[18] decides a novel idea to implement relevance oriented ranking and develops formulae for identifying search nodes and nodes via the query.

3. Proposed System

The Ranking of XML documents has to be performed in a way that there will not be any inconsistency of retrieving the XML documents from the XML database. XML benchmark datasets are taken as inputs for the system. Implementing the adaptive search before the ranking of the XML documents is certainly a benefit to the overall efficiency of the system. The adaptive system gives the advantage to the user to select the search algorithm on his benefits. After the ranking of the retrieved XML documents, performs the reverse indexing process to improve the performance and data integrity across diverse applications.

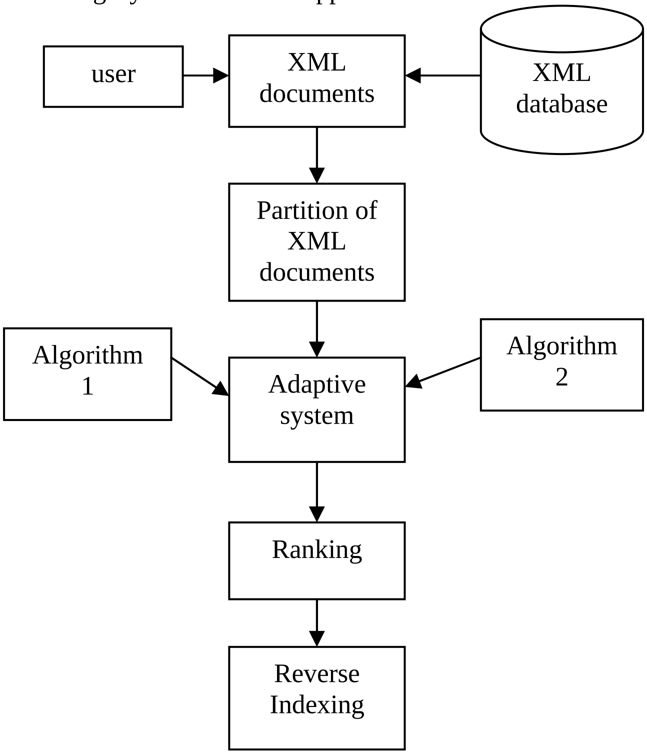

Fig. 3.1: Proposed System Architecture

A. Selection of data:

This is the first phase of the proposed method. The benchmark data sets have to be selected from the web which is appropriate to our content intentions. The relevant XML content has to be downloaded for further searching of keywords

B. Partition of XML documents:

After the selection phase, the data is retrieved and partitioned according to the XML tags. All the tags will be separated and the content of those tags will act as their child nodes. Every main tag has sub tags, and these sub tags also contain data. The head tag becomes the parent and the others will become the children.

C. Searching the XML content:

This phase contains the adaptive system which has the ability to implement two or more algorithms depending on the user selection. An option is given to the user to select the algorithm which he needs to process to the next phase.

The proposing algorithms are the Boolean Retrieval algorithm and any of the LCA based algorithm. The above two can be implemented. As it is an adaptive system, the algorithms can be changed according to the developer convenience and user requirements.

D. Ranking of XML documents:

After searching the documents in which the keywords are available, these documents have to be ranked in the sequence of containing more number of the searched keywords in it. The best document with more amount of the search content available in it will be ranked in the top and the other documents follow in the same order. Any of the efficient ranking algorithms like XRank[2], XReal[18] etc., can be used.

E. Reverse Indexing:

This phase is considered to be important as it shows us the keyword in the document after the ranking is completed. It is a type of backtracking algorithm which is used to track the word in the document and whether the search and rank algorithms are working properly. With this phase, the system can be used among various applications and platforms.

4. Experiment

Though the proposed system is said to be efficient, it needs some experimental results for the system to be built. Performance analysis has to be done to the system by using the famous performance metrics Precision and Recall. Precision[20] is the ratio of number of relevant records retrieved to the total number of relevant records in the database. Recall[20] is the ratio of number of relevant records retrieved to the total number of irrelevant and relevant records in the database.

TABLE 1: Performance Analysis

Total size of Documents	**Number of relevant documents retrieved**	**Number of not relevant documents retrieved**	**Number of irrelevant documents retrieved**	**Precision**	**Recall**
1 MB	2	0	0	100	100
5 MB	5	0	0	100	100
10 MB	9	0	0	100	100

5. CONCLUSION

This adaptive system, if implemented in a right way will be very much useful than any system for retrieving XML documents. Its' flexibility helps to change it in anyway desired by the programmer. This combination of searching algorithms, ranking and reverse indexing helps in the development of the easy processing of XML documents.

It also saves a lot of time and also the performance will be perfect, as we have seen it in the experimental section.

REFERENCES

[1]Y. Xu and Y. Papakonstantinou, "Efficient Keyword Search for Smallest LCAs in XML Databases," in Proceedings of the 2005 ACM SIGMOD International Conference on Management of Data (SIGMOD'05), 2005, pp. 537–538.

[2]L. Guo, F. Shao, C. Botev, and J. Shanmugasundaram, "XRANK: Ranked Keyword Search over XML Documents," in Proceedings of the 2003 ACM SIGMOD International Conference on Management of Data (SIGMOD'03), 2003, pp. 16–27.

[3]V. Hristidis, N. Koudas, Y. Papakonstantinou, and D. Srivastava, "Keyword Proximity Search in XML Trees," IEEE Trans. Knowl. Data Eng. (TKDE), vol. 18, no. 4, pp. 525–539, 2006.

[4]L. Kong, R. Gilleron, and A. Lemay, "Retrieving Meaningful Relaxed Tightest Fragments for XML Keyword Search," in Proc. 2009 International Conference on Extended Data Base Technology (EDBT'09), 2009, pp. 815–826.

[5]G. Li, J. Feng, J. Wang, and L. Zhou, "Effective Keyword Search for Valuable LCAs over XML Documents," in CIKM, 2007, pp. 31–40.

[6]Y. Li, C. Yu, and H. Jagadish, "Schema-Free XQuery," in Proceedings of the 30th International Conference on Very Large Data Bases (VLDB'04), 2004, pp. 72–83.

[7]Z. Liu, J. Walker, and Y. Chen, "XSeek: A Semantic XML Search Engine Using Keywords," in Proceedings of the 33rd International Conference on Very Large Data Bases (VLDB'07), 2007, pp. 1330–1333.

[8]C. Sun, C. Chan, and A. Goenka, "Multiway SLCA-basedKeyword Search in XML Data," in WWW, 2007, pp. 1043–1052.

[9]Y. Xu and Y. Papakonstantinou, "Efficient LCA Based Keyword Search in XML Data," in Proc. 2008 International Conference on Extended Data Base Technology (EDBT'08), 2008, pp. 535–546.

[10]Z. Liu and Y. Chen, "Identifying Meaningful Return Information for XML Keyword Search," in Proceedings of the 2007 ACM SIGMOD International Conference on Management of Data (SIGMOD'07), 2007, pp. 329–340.

[11]V. Hristidis, Y. Papakonstantinou, and A. Balmin, "Keyword Proximity Search on XML Graphs," in Proc. International Conference on Data Engineering (ICDE'03), 2003, pp. 367–378.

[12]H. He, H. Wang, J. Yang, and P. S. Yu, "BLINKS: Ranked Keyword Searches on Graphs," in Proceedings of the 2007 ACM SIGMOD International Conference on Management of Data (SIGMOD'07), 2007, pp. 305–316.

[13]G. Bhalotia, A. Hulgeri, C. Nakhe, S. Charkrabarti, and S.Sudarshan, "Keyword Searching and Browsing in Databases using BANKS," in Proc. International Conference on Data Engineering (ICDE'02), 2002.

[14]G. Li, B. C. Ooi, J. Feng, J. Wang, and L. Zhou, "EASE: Efficient and adaptive keyword search on unstructured,semi-structured and structured data," in Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data (SIGMOD'08), 2008, pp. 903–914.

[15]G. Li, J. Feng, J. Wang, B. Yu, and Y. He, "Race: Finding and Ranking Compact Connected Trees for Keyword Proximity Search over XML Documents," in WWW, 2008, pp. 1045–1046.

[16]S. Cohen, J. Mamou, Y. Kanza, and Y. Sagiv, "XSEarch: A Semantic Search Engine for XML," in Proceedings of the 29th International Conference on Very Large Data Bases (VLDB'03), 2003, pp. 1069–1072.

[17]Y. Huang and Y. C. Z. Liu, "eXtract: A Snippet Generation System for XML Search," PVLDB, vol. 1, no. 2, pp. 1392–1395, 2008.

[18]Z. Bao, T. Ling, B. Chen, and J. Lu, "Effective XML Keyword Search with Relevance Oriented Ranking," in Proc. International Conference on Data Engineering (ICDE'09), 2009, pp. 517–528.

[19]Z. Liu and Y. Chen, "Reasoning and Identifying Relevant Matches for XML Keyword Search," PVLDB, vol. 1, no. 1, pp. 921–932, 2008.

[20]W. Yang, H. Zhu, N. Li, and G. Zhu, "Adaptive and Effective Keyword Search for XML," in PAKDD, 2011, accepted.

[21]W. Yang and H. Zhu, "Semantic-Distance Based Clustering for XML Keyword Search," in PAKDD, 2010, pp. 398–409.