Global Journal of Advanced Engineering Technologies Volume 5, Issue 4- 2016
ISSN (Online): 2277-6370 & ISSN (Print):2394-0921

A SURVEY: TOP-DOWN XML KEYWORD QUERY PROCESSING

[1] Nikita R. Alai, [2] A.S. Vaidya
[1]Student, ME (CSE), Gokhale Education Society R.H.Sapat College of Engineering management studies and Research , Savitribai Phule Pune University,Nashik, Maharashtra, India.
[2]Professor, Gokhale Education Society R.H.Sapat College of Engineering management studies and Research, Savitribai Phule Pune University, Nashik, Maharashtra, India.

Abstract: *From the past many years there has been much scope in efficiently replying to the XML (eXtended Markup Language) keyword queries. Existing system has limitations as the common-ancestor-repetition (CAR) and visiting-useless-nodes (VUN) which results inefficiency. In order to solve CAR problem we introduce a generic top-down strategy to answer keyword query. To solve the issue of VUN we implement to make use of child nodes, instead of descendant. In this survey, we propose algorithms like L List based algorithm, Hash search based algorithms, optimized hash search algorithm to improve the performance of the system [1]. To answer a given keyword query basic semantics like lowest common ancestor (LCA), exclusive LCA (ELCA), smallest LCA (SLCA) are proposed. For the efficiency of query processing we use XML keyword, which results in the faster retrieval of the document. With all these techniques and algorithms, various case studies have been performed and results are analyzed to speed up the performance of the system.*

Keywords*-CAR, ELCA, LList, LCA, SLCA, query processing, top-down strategy, XML keyword*

I. INTRODUCTION

The different techniques and algorithms of top down strategy for XML keyword query processing are illustrated n this paper. The aim of this survey is to make sure that answering the given query in faster and efficient way. Many applications in the business and scientific domains XML has been widely used for storing, exchanging and publishing data. When compared with structured query languages such as X Query and XPath, keyword search is also popular over XML data. It acts as substitute for users from understanding complex queries. One of the strengths of XML is that it can be used to represent structured and unstructured data. Keyword search is important to query XML data. There are various indexing techniques are used to solve the searching problem as well as with the help of tree model which is used to store XML data, query processing is accelerated. A system that supports query semantics captures more meaningful results.

The common issues that results in redundancy are CAR and VUN problems.

CAR problem:

In graph theory the lowest common ancestor of two nodes v and w in a tree T is the lowest i.e. deepest node that has both v and w as descendants, where each node to be descendant to itself. While multiple operations results in all common ancestors on the path from root to visiting nodes to be repeatedly visited, which is called as common ancestor repetition (CAR).

VUN problem:

Given a keyword query Q and XML document D, let v be the set of nodes D that contains one keyword query in their sub trees then we can classify them into following categories:

- *Common ancestor (CAs);*
- *Useless nodes (UNs);*
- *Auxiliary nodes (AUs).*

Considering these problems they proposed to support various query semantics with generic processing strategy which is used to solve CAR and VUN problems more efficiently.

In order to address CAR problem they proposed XML keyword query processing with generic top town approach. To address VUN problem they proposed to use the child nodes instead of descendants with respect to query semantics to test LCA, SLCA, ELCA nodes. They also proposed labeling scheme independent inverted list (LList) and to improve the performance of the system they also used the hash index.

II. XML KEYWORD AND DATA

In order to manage XML documents Lukas Kircher et al. proposed a technique known as structural bulk updates that is used with X Query update facility to support Pre/Dist/Size encoding. Updating XML is challenging work as structural order of documents had to be observed. They introduced a method known as avoiding redundant distance adjustment which avoids repeated and redundant distance. They demonstrated how the cost of maintaining the document order as well as this technique reduces processing time also large bulk updates are feasible [2]. The basic technique for XML keyword search is LCA. Manoj Agarwal et al. presented a novel system Generic Keyword Search (GKS) which is able to find highly relevant XML schema elements and keywords, deeper analytics insights known as DI in the XML data. They proposed XML node categorization model to expose XML elements. They introduced ranking methodology for data keywords discovery. They also presented GKS system used for real datasets. This technique over the real data sets shows highly relevant responses to keyword queries efficiently. Users navigated XML data seamlessly and which was highly relevant data [3]. The problem of effective keyword search over XML documents had been studied by Guo liang Li et al. They introduced Valuable Lowest Common Ancestor (VLCA) to answer keyword queries over XML documents efficiently and effectively. They also proposed Compact VLCA (CVLCA) for optimizing strategy for speeding up the computation and for answering keyword queries accurately. The experimental result showed that the proposed methods achieve high quality results on both real and synthetic datasets [4]. Keyword search is a user friendly way to query HTML document.

Yu Xu and Yannis Papakonstantinou contributed the indexed lookup eager and scan eager algorithm which exploits the key properties of smallest trees in order to achieve order of magnitude of keyword containing queries with different frequencies. These algorithms produce answer quickly and for first few answers users did not have to wait. The analytical as well as experimental results showed that the algorithm outperforms by order of magnitude when keywords had different frequencies [5]. A user query is a set of keywords which match with labels or values of nodes in XML trees. Khanh Nguyen and Jinli Cao introduced novel approach known as Relevant LCA (RLCA) to accurately and efficiently capture relevant fragments to XML keyword search. Experimental results showed the effectiveness of RLCA and carefully measured precision, recall and F-measure which achieved high effectiveness [6].Answering keyword queries on XML data with keyword is studied extensively. Rui Zhou et al. proposed Hash Count Algorithm which is used to find ELCA to answer keyword queries on XML data which showed complexity O(kd|S1|). They also introduced two versions, a naïve and optimized [7].Vagelis Hristidis et al. presented an algorithm to compute the Minimum Connecting Trees (MCT) of the nodes that contains keywords. They designed and analyzed efficiency of algorithm in two cases; 1) when relevant indices had been constructed and XML data had been preprocessed 2) when XML data had not been preprocessed. XML keyword queries can be efficiently determined as a part of query evaluation. Information retrieval can be done [8]. To address the SLCA computation problem in XML data Ba Quan Truong et al. proposed a property called optionality resilience which specified behaviors of an XKS for queries with missing elements. The experimental results showed quality of search, execution time, scalability, number of missing elements, number of keywords and heuristics for algorithm selection. It also showed that MESSIAH not only produced high quality result but also faster computation speed [9].

III. INDEXING ALGORITHMS

The indexing technique's performance is studied based on characteristics and requirements. For effective, efficient and accurate retrieval of documents, existing techniques for indexing became inefficient with the tremendous and rapid growth of index size as well as seek time regarding optimized index scheme. Data is rapidly increasing in terms of structured and non-structured. To handle such large amount of data efficiently indexing techniques are designed. Indexing is developed in order to tolerate high cost and precise search. Basically indexing techniques are categorized based in three methods:

1. *Non Artificial Intelligent (NAI),*
2. *Artificial Intelligent (AI) And*
3. *Collaborative Artificial Intelligent (CAI).*

Non artificial or traditional indexing techniques are tree based indexing, bitmap indexing, graph query processing, hashing, B tree, R tree which uses classifiers for indexing. Artificial indexing techniques are more accurate method in constructing hybrid indexing mechanism. The techniques used under AI are fuzzy decision tree (FDT) and machine learning which produces efficient result [10]. Standard disk based structure and algorithms which includes B+ trees, heap files, disk based prefix trees, inverted indexes, binary large object (BLOB) files, m way posting list intersection, LRU buffer manager and external sorting used to build storage engine. Wook-Shin Han et al. implemented graph indexing techniques and demonstrated various datasets and workload to show various unique features such as performance analysis. They showed that tool supported for importing dataset, selecting index algorithm, building index structure, specifying query workload, executing query and navigating through the results [11]. Indexing techniques are used to speed up the data retrieval. A. John et al. studied various approaches used to minimize the history data. Update on change and sampling are the main two approaches which are based on spatiotemporal indexing method. Data is represented in two types:

- *Certain Data (Constant Value)*
- *Uncertain Data (Inexact Data).*

Both this data types had its indexing technique based on which tree structure is used. Main indexing techniques are HBase index, Threshold interval index, External interval tree index, U-Grid, PTI index, MON tree, LGU tree, Gauss tree, Segment based index, FUR tree, RUM tree [12].

Guimei Liu et al. studied three structures for indexing as well as querying frequent item sets:

- *Signature Files*
- *Inverted Files*
- *CFP Tree. .*

Four algorithms are used such as

- *Superset Search Algorithm,*
- *Subset Search Algorithm,*
- *Tree Exact Match Algorithm*
- *Exact Search Algorithm.*

Query processing was done by using CFP tree. Experimental result showed that no structure can outperform other structure also CFP tree showed better performance than other two techniques [13]. For storing, querying and analyzing massive data, it had become tedious to develop effective techniques for databases. Benjarath Phoophakdee and Mohammed J. Zaki proposed a novel disk based suffix tree algorithm named as trellis which effectively scales up performance in terms of querying time and indexing time [14]. Answering XML queries using by using XML indexes is a basic approach. Wei Wang et al. proposed highly optimized disk organization method for an F&B Index with clustering properties. Experimental results showed that F&B Index can scale up with good query performance for large data size compared with XML query processing algorithms. It showed that all structural indexes for XML data took a path query as input and reported exactly all the matching nodes as output within the indexes via searching. XML data itself rich in structure which call for indexing techniques which facilitate query processing efficiently. Experiments showed several features as cache friendliness and good scalability [15].

IV. QUERY PROCESSING

According to the various functions of the query, classifications are made such as moving object, range based query, location based query, trajectory queries, future data detection query. According to user requirements the query processing varies. Query processing is nothing but a collection of data which are arranged for speed and ease of search for retrieval of data. Indexing and query processing are interrelated [12]. Cheng YR et al. proposed filtering and verification framework to improve search efficiency. They defined a r-clique keyword query result for knowledge base environment and derived tightest upper bound. They designed an index which facilitates pruning algorithm and sampling algorithm. The demonstrated result showed that proposed definition satisfied the user requirements compared with r-clique definition and algorithm were efficient [16]. Querying XML document effectively and efficiently is a challenging issue. Mikael Fernandus Simalango studied query processing issues and proposed solutions for querying XML databases. They reviewed evolving path for XML query languages also provided different approaches for XML query processing. Several challenges for the realization of scalability of XML database management system still exist [17].

For query processing, List intersection is a central operation which is utilized excessively on text and databases. Query processing is method of retrieving the inverted indices which corresponds to query keywords and intersecting them for identification of relevant documents. Sudipto Guha et al. presented algorithm to compute intersection of an arbitrary number of sorted as well as unsorted lists which shows superior performance to achieve good speed up, effectiveness and load balance. They studied list intersection algorithm to reduce overhead of cache hierarchy and to take benefit of parallelism. Result evaluation is based on real and synthetic data which validates efficiency of proposed algorithm [18]. Vishwakarma Singh et al. studied queries that ask for satisfying given set of keywords of the tightest groups of points. They proposed a novel method known as ***Pro MiSH (Projection and Multiscale Hashing)*** which is used to achieve high scalability and speedup using random projection and hash based index structure. They presented an algorithm for finding top k tightest clusters in subset which retrieve the points from disk using B+ tree for exploration of final set of result. The results on real as well as synthetic data showed that ProMiSH had up to 60times of speed up over tree based techniques

[19]. In order to improve scalability Evandrino G. Barros et al. introduced PMK Stream (Parallel MK Stream) which evaluated multiple keyword queries for multiple parsing stacks. The experimental results showed that PMK Stream is efficient for supporting keyword based search over XML data [20]. Prefix-based numbering (PBN) was proposed by Curtis E. Dyreson et al. which is a popular method for numbering nodes in the hierarchy. They presented a strategy to virtually transform the data without renumbering and instantiating. The result was concise, support efficient querying, updating was efficient and practical [21]. DipaliPal et al. proposed a way for indexing for large database which includes small and medium size graphs. For query processing, a query graph was mapped into signature which was used to search results. The experimental results are carried on both real as well as synthetic dataset and this approach provided a scalable, effective and efficient disk based solution for large and medium dataset [22].Donald Kossmann presented a technique for query processing which is used for information systems and distributed database. He proposed an architecture known as textbook which uses various techniques for parallelism. He also discussed distributed systems such as middle ware, client server and heterogeneous database system used for query processing [23]. With the help of experimental results Daniela Florescu et al. showed how XML query language could be elaborated to support keyword search. By combining structured query processing and keyword search was useful for both knowledge structure and XML data. Query performance calculated on the basis of three types such as structured, partially structured and unstructured. Indexing process is done by using inverted files and relational database. The result showed XML query processing performed efficiently and non-structure query executed faster than structured query [24].

V. CONCLUSION

Key factors which results in inefficiency for existing XML keyword search considering algorithms are CAR and VUN problems. These problems are solved by using generic top down strategy and use of child nodes. For query semantics independent approach is used where efficient algorithms are used such as LList and Hash based methods which reduce time complexity. To reduce memory overload size of index became too large. For which we can propose disk based index approach which can reduce memory overload and improve the performance of the XML keyword search for the query processing.

VI. ACKNOWLEDGMENT

I wish to express my sincere thanks and the deep sense of gratitude to respected guide A.S. Vaidya in Department of Computer of Gokhale Education Society's, R. H. Sapat College of Engineering Management Studies & Research, Nashik, for the technical advice, encouragement and constructive criticism which motivated to strive harder for excellence. I also wish acknowledgment to the people who gives support direct or indirectly to the paper writing.

REFERENCES

[1] Junfeng Zhou, Wei Wang, Ziyang Chen and Jeffrey Xu Yu," Top-Down XML Keyword Query Processing", in IEEE Transactions on Knowledge and Data Engineering,Volume:28,Issue: 5, May 1 2016,pp. 1340 – 1353.

[2] L. Kircher, M. Grossniklaus, C. Grun, and M. H. Scholl, "Efficient structural bulk updates on the pre/dist/size XML encoding", in Proc. IEEE 31st Int. Conf. Data Eng., 2015, pp. 447–458.

[3] M. K. Agarwal and K. Ramamritham, "Enabling generic keyword search over raw XML data", in Proc. 31st Int. Conf. Data Eng., 2016, pp. 1496–1499.

[4] G. Li, J. Feng, J. Wang, and L. Zhou, "Effective keyword search for valuable LCAS over XML documents", in Proc. 16th ACM Conf. Conf. Inform. Knowl. Manage., 2007, pp. 31–40.

[5] Yu Xu and Yannis Papakonstantinou, "Efficient Keyword Search for Smallest LCAs in XML Databases", in SIGMOD '05 Proceedings of the 2005 ACM SIGMOD international conference on Management of data, June 2005, pp. 527-538.

[6] Khanh Nguyen and Jinli Cao, "Exploit Keyword Query Semantics and Structure of Data for Effective XML Keyword Search", in Proc. 21st Australasian Database Conference, Brisbane, Australia, Volume 104, January 2010, pp. 133-140.

[7] R. Zhou, C. Liu, and J. Li, "Fast ELCA computation for keyword queries on XML data", in Proc. 13th Int. Conf. Extending Database Technol., 2010, pp. 549–560.

[8] Vagelis Hristidis, Nick Koudas, Yannis Papakonstantinou, and Divesh Srivastava,"Keyword Proximity Search in XML Trees", in IEEE transactions on knowledge and data engineering, Volume. 18, NO. 4, APRIL2006, pp. 1-15.

[9] B. Q. Truong, S. S. Bhowmick, C. E. Dyreson, and A. Sun, "MESSIAH: Missing element-conscious SLCA nodes search in XML data", in Proc. SIGMOD, 2013, pp. 37–48.

[10]Abdullah Gani, Aisha SiddiqaShahaboddin Shamshirband and Fariza Hanum, "A survey on indexing techniques for big data: taxonomyand performance evaluation", in Knowledge and Information Systems, Volume 46, Issue 2, March 2015, pp. 241-284.

[11] Wook Shin Han, Minh Duc Pham, Jinsoo Lee, Romans Kasperovics, and Jeffrey Xu Yu, "iGraph in Action: Performance Analysis of Disk-BasedGraph Indexing Techniques", in SIGMOD Conference, 2011, pp. 1241-1242.

[12]A John, M Sugumaran, and RS Rajesh," Indexing And Query Processing Techniques In Spatio-Temporal Data", in Ictact Journal On Soft Computing, Volume: 06, Issue: 03, April 2016, pp. 1198-1217.

[13] Guimei Liu, Andre Suchitra, and Limsoon Wong, "A performance study of three disk-based structures for indexing and querying frequent itemsets", in Proceedings of the VLDB Endowment, Vol. 6, No. 7, May 2013, pp. 505-516.

[14] Benjarath Phoophakdee and Mohammed J. Zaki, "Genome- scale disk-based suffix tree indexing", in SIGMOD '07 the 2007 ACM SIGMOD international conference on Management of data, June 2007, pp. 833-844.

[15] Wei Wang, Hongzhi Wang, Haifeng Jiang, "Efficient processing of XML path queries using the disk-based F&B Index", in LDB '05 Proceedings of the 31st international conference on Very large data bases, 2005, pp. 145-156.

[16] Cheng YR, Yuan Y, Li JY and Lei Chen, "Keyword Query over Error-Tolerant Knowledge Bases", in Journal Of Computer Science And Technology, July 2016, pp. 702-719.

[17] Mikael Fernandus Simalango, "XML Query Processing and Query Languges: A Survey", in Data Structures and Algorithms, October 2010.

[18] Dimitris Tsirogiannis, Sudipto Guha and Nick Koudas, "Improving the Performance of List Intersection", in Proceedings of the VLDB Endowment, August 2009, pp. 838-849.

[19] Vishwakarma Singh, Bo Zong, and Ambuj K. Singh, "Nearest Keyword Set Search in Multi-Dimensional Datasets", in IEEE Transactions On Knowledge And Data Engineering, Vol. 28, No. 3, March 2016, pp. 741-755.

[20] Evandrino G. Barros, Fernando G. D. C. Ferreira, Alberto H. F. Laender, "Parallelizing Multiple Keyword Queries over XML Streams", in Data Engineering Workshops (ICDEW), 2016 IEEE 32nd International Conference, June 2016, pp. 1-4.

[21] Curtis E. Dyreson, Sourav S. Bhowmick and Ryan Grapp, "Querying virtual hierarchies using virtual prefix-based numbers", in SIGMOD '14 Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data, June 2014, pp. 791-802.

[22] Dipali Pal, Praveen Rao, Vasil Slavov and Anas Katib, "Fast processing of graph queries on a large database of small and medium-sized data