MA Algorithm to generate Semantic Web Based Clustered Hierarchy for Keyword Search

Archana Mathur
Nitte Meenakshi Institute Of Technology, Bangalore, India
44 PUBLICATIONS **125** CITATIONS
SEE PROFILE

SH Manjula
UVCE, Bangalore University
94 PUBLICATIONS **202** CITATIONS
SEE PROFILE

Venugopal K R
University Visvesvaraya College of Engineering
925 PUBLICATIONS **3,719** CITATIONS
SEE PROFILE

Lalit M Patnaik
Indian Institute of Science
838 PUBLICATIONS **8,689** CITATIONS
SEE PROFILE

Int. J. of Recent Trends in Engineering & Technology, *Vol. 11, June 2014*

MA Algorithm to Generate Semantic Web Related Clustered Hierarchy for Keyword Search

Leena Giri G[1], Archana Mathur[1], Manjula S H[1], Venugopal K R[1], L M Patnaik[2]

[1] Department of Computer Science and Engineering
University Visvesvaraya College of Engineering
Bangalore University, Bangalore – 560 001
Email: leenagiri@gmail.com
[2] Honorary Professor, Indian Institute of Science,
Bangalore 560 001, India

Abstract - **Keyword search in XML documents is based on the notion of lowest common ancestors in the labeled trees model of XML documents. It has recently gained a lot of research interest in the database community. In this paper we propose the Modified Active (MA) algorithm which is an improvement over the Active clustering algorithm. The algorithm takes into consideration the entity aspect of the nodes to find the level of the node pertaining to a keyword input by the user. A portion of the Bibliography database is used to experimentally evaluate the Modified Active algorithm. Evaluation results show that MA algorithm generates clusters faster than the Active algorithm and this increases the efficiency of the system.**

Index terms **- Keyword Matching Patterns, MA algorithm, Semantic search**.

I. Introduction

The World Wide Web (WWW) is a huge collection of information with representation of data in diverse formats such as text, XML, multimedia streams and images. It serves as a container of largely unstructured data which is exchanged between different platforms across the globe. The predominant language XML, which has gained popularity because of the framework it provides to facilitate the transfer of information on the web, has completely changed today's keyword search paradigm.

Human language uses endless variety and combination of words to express thoughts, concepts and meanings which could create ambiguity in keyword searches. The user of the system often believes that the keyword search terms input by him are precise and self-descriptive. However, the same keywords when used for search on the web often returns matches completely hiding the relevant documents (even documents containing the synonyms of the keyword terms entered by the user) making keyword searches an iterative and exhaustive process. Terms that are common, but appear in different domains, tend to produce a huge result-set of which a major portion would be irrelevant to the user query. This forces the user to sieve through the obtained result-set to obtain a more precise result-set.

Various methods and techniques have been developed to rank the relevant documents based on the query submitted by users. XRank semantics [1] gained popularity in the research community because of its ability to rank the documents based on the query submitted by the user. Similarly, the notion of lowest common ancestor (LCA) used by many researchers to propose techniques like SLCA(Smallest Lowest Common Ancestor), MLCA(Multi-way Lowest Common Ancestor) and ELCA(Exclusive Lowest Common Ancestor) has further enhanced the keyword search domain. However, very little effort has been made in the path of developing a search technique wherein the semantics of the query would be understood before proceeding to find the answers relating to the query. Tim Berners-Lee, the inventor of World Wide Web, has defined Semantic Web as a system which enables machines to *"understand*" and respond to complex human requests based on their meaning. Such an "*understanding*" requires that the relevant information sources be semantically structured and retrieved [2]. Xiping Liu, Changxuan Wan, Lei Chen [3] have proposed an efficient way to understand the semantics of the keyword terms. Further exploring their work, this paper proposes a Keyword Search technique, which initially finds the semantic relationship of the query terms and generates output in the form of clusters and matching patterns later. This generated output makes it convenient for the user to view the relevant answers at a faster rate.

The proposed algorithm considers the level information of entity nodes and modifies the existing active algorithm [3] to generate clusters dynamically. MA algorithm uses the concept of LCEA (Lowest common entity ancestor) to generate clusters by taking into consideration the level information of the keyword and the entity nodes that is generated when the algorithm is run. This modification in the algorithm provides an improved response time and a means by which the user interaction with the search engine interface is increased. When clusters are generated, the interface waits for user input to display patterns (similar to KMPs) for the user to explore a specific cluster. Displaying patterns to the user brings clarity in the answers returned to users. A portion of the Bibliography database has been used for the experimental evaluation and execution of the algorithm. The modified algorithm finds its entire means to generate answers only for the keywords in which the query terms are semantically related. The algorithm generates no answers if the keywords are not related to one another. This is as explained below:

Consider the keyword query ***Q*** = ***(t1,t2)*** = ***(Mysore, Univ.)***

When the query ***Q*** is executed, the algorithm returns no answers even though the keywords exist in the XML documents. The reason for this is that the LCEA for these query terms is void. This means that the common ancestor node (which is an entity node) is void. The concept of an entity node plays a vital role in finding whether the keyword terms are semantically related or not.

Another the keyword query ***Q*** = **(*t1, t2*)** = **(*ieee, QoS*)** returns clusters related to conferences on *QoS* from the XML database. The LCEA for these keywords is *conf* which is a common entity node for the keyword terms *ieee* and *QoS*. It can be observed, from the extended bibliography database that the keywords *ieee* and *QoS* are semantically related to one another.

The above queries clearly state that until the query terms submitted don't hold any meaning, the result generated is null. The keyword terms submitted are explored for semantic relationship before presenting any answers to enhance the quality of the response generated. This is to ensure that the users are not forced to sift through the irrelevant results that is returned.

A. Motivation

Xiping Liu, Changxuan Wan, Lei Chen [3] have proposed a novel approach to perform clustering in an active way, i.e., the algorithm first computes KMPs, and then generates clustered search results using these KMPs. The generated clusters are further improved by organizing them into a hierarchy. By this method, the algorithm returns fuzzy answers in the form of clusters. A naïve user is likely to find this approach confusing if he is not sure of which cluster he should consider to find his answer.

B. Contribution

This work proposes the MA algorithm which is based on the Active algorithm to generate clusters for a keyword query on an XML document for a Bibliography Database. The approach is modified in such a way that instead of deriving KMP's first, the algorithm generates clusters in response to user queries. A selected cluster is considered by the user to obtain the KMP of the cluster of his interest. By modifying the already existing approach, the clarity of the answers returned to the user is improved. This approach improves the response-time and the user interaction with the system. The algorithm merges both the cluster generating and cluster hierarchy constructing algorithms and the result is generated as a hierarchy in an enhanced interface for the user.

C. Paper Organization

Section 2 presents a discussion on research objectives in the area of XML keyword search. Section 3 defines the problem and discusses the algorithm and its implementation. Section 4 gives the performance analysis and the difference in processing time for identical keywords by the Active algorithm and the MA algorithm. Section 5 gives the Conclusion and Future Enhancement.

II. Related Work

There continues to be extensive research work in the area of XML keyword search as well as search results clustering. Xiping Liu, Changxuan Wan, Lei Chen [3] address the problem of returning clustered results for keyword search on XML documents. They propose a novel semantics for answers to an XML keyword query i.e., the conceptually related relationship between keyword matches, which is based on the conceptual relationship between nodes in XML trees. A new clustering methodology for XML search results which clusters results according to the way they match the given query is proposed.

Ziyang Liu , Jeffrey Walker, Yi Chen [4] presented XSeek, a search engine that returns an XML fragment or a *Subtree Return or a Path Return,* in response to a keyword query. The algorithm returns meaningful answers allowing users to specify an XML Document and keyword query for retrieval. It suffers drawbacks in terms of presentation of result. A naïve user of the search engine may find it difficult to explore the answers in the form of XML fragments. The user interaction with the system is considerably low and the relevant ranking for the result is missing. Guo et. al., has proposed XRank, an algorithm that considers the problem of efficiently producing ranked results for keyword search queries over hyperlinked XML documents and the challenges in evaluating keyword search queries over hierarchical XML documents. XML keyword search queries can return deeply nested XML elements that contain the desired keywords and the nested structure of XML documents implies that the notion of ranking is at the granularity of an XML element.

Sara Cohen, Yaron Kanza, Benny Kimelfeld and Yehoshua Sagiv [6] proposed a framework to decide when the tags of an XML document tree are semantically related. The author has made an effort to reduce the irrelevant answers by displaying only those answers which are related to the query. However the work does not consider the ranking of the answers returned to a user. Most of the keyword search algorithms are based on the notion of LCA. However their semantics do not consider if the keyword matches are semantically related or not. . Schmidt, M. Kersten, and M. Windhouwer [7] address this problem by introducing the concept of interconnection on homogeneous nodes. They do not have any restriction and is not comprehensive enough. Most of the algorithms for computing LCAs of two nodes on trees are meant for main-memory resident data, and do not consider computing LCAs for sets of nodes. The "meet" operator is proposed for querying XML documents by computing the LCAs of nodes in XML trees. This does not give a generalized algorithm.

Ziyang Liu, Y. Chen [8], in their work gave a novel way of answering a query by using materialized views. In this, the performance of the search increases by many folds. The drawback is that the algorithm requires views to be created and stored in advance. This may impact the space complexity while enhancing time complexity. They propose MaxMatch, an XML keyword search technique to identify relevant matches that incorporates two aspects during the search: *consistency* and *monotonicity of data and query*. Y. Huang, Z. Liu, and Y. Chen [9] discuss the problem of generating result snippets for XML search. Recently, Z. Liu, P. Sun, and Y. Chen [10] address the problem of comparing and differentiating search results on structured data. Their emphasis is on the contents of XML fragments and the concentration is on differences in processing matching semantics. Only a few of the clustering of search results are based on XML search. The only major work that addresses this problem is done by V. Hristidis, N. Koudas, Y. Papakonstantinou, and D. Srivastava [11]. Another related problem is the grouping of XML search results, which has been addressed in [12]. However, grouping is different in nature from clustering.

Liang Jeff Chen, Yannis Papakonstantinou [13] propose an algorithm that combines semantic pruning and top-K processing to generate top-K keyword searches in an XML document. Most of these algorithms do not consider an important aspect of keyword search, that is the presentation of the answers to be returned to user. Recently, XML document clustering has been the topic of many research papers. W. Lian, D.W.-L. Cheung, N. Mamoulis, S.-M. Yiu [14], propose S-GRACE, a hierarchical algorithm to cluster XML document using structural information of the data. However the technique does not take care of unstructured information stored on the Web. Chong Sun, Chee-Yong Chan, Amit K. Goenka 15] have proposed an algorithm, Inks, which predicts a word or a phrase the user may type in, by simultaneously searching the XML data and returning the best answer ranked by relevance. The interactive search method uses ELCA to find the relevant answers. In this technique, the predicted XML elements for all the keyword tokens are extracted from a given XML tree. ELCAs are computed on the resulting elements followed by ranking the answers using *tf-idf* based ranking function [16]. Rui Zhou, Chengfei Liu, Jianxin Li, Jeffrey Xu Yu [17], contributed to the world of keyword search by proposing PrELCA, which defines probabilistic ELCA semantics by designing a stack-based algorithm for keyword search on probabilistic XML documents Z. Bao, T.W. Ling, B. Chen, and J. Lu [18], proposed XReal which attempts to infer the user's intention from the query by calculating the similarity (using the *tf-idf* cosine similarity) between keywords. Once computed, the relevance oriented ranking is achieved to return keywords as result.

Ch. Lavanya Susanna [19], analyzes various methods of querying XML data based on LCA, ELCA and heap. From these methods, LCA does not produce optimal results and heap-based method is more efficient.   The method concentrates more on generating relevant results for keyword queries without recognizing query semantics and cannot be used to solve all problems.

III. Problem Definition

Given a set of keywords for search, the algorithm takes into account the entity aspect of the nodes in the XML tree to find the level of the node pertaining to a particular keyword input by the user. The MA algorithm improves the response time of the system and hence increases the efficiency of the system. The algorithm uses key information to generate clusters for a keyword query that is input by the user. User inputs are considered during execution for enhanced performance to generate clusters for the semantically related results that is obtained for the user's query. The cluster generating and the cluster hierarchy constructing algorithms are merged and the result is generated in a hierarchy to provide an enhanced interface for users to view.

A. Algorithm and Implementation

To explain the proposed algorithm, the example of a portion of a Bibliography database [3] is considered.

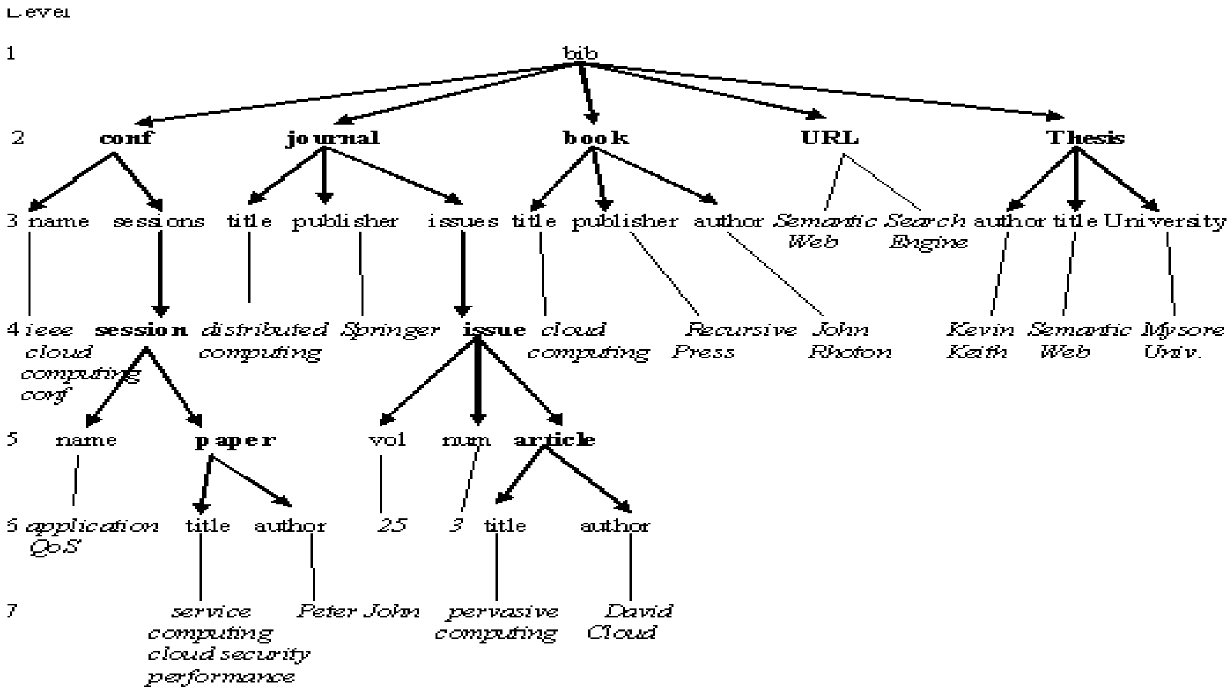

Figure 1. XML tree for the Bibliography Database

There can be multiple interpretations, for the keyword '*John Computing'* which is two-keyword query. A user may be looking for a *book* titled *Computing* that is authored by *John* or could be searching for a *paper* or *article* titled *Computing* that is written by *John*. Without reasoning and with such different interpretations, finding the correct answer becomes difficult. Depending on the answer nodes that lead to the keywords, the results are clustered and presented to the user.

As another example, for the query *'Computing Peter',* nodes *conf1*, *paper3* and *paper1* can be the answer nodes, as in Fig. 1, since the user may be keen on finding a *conference* or a *paper* on *computing* that is authored by *Peter*. These results are clustered together and presented to the user.

B. Preliminaries

Data Model : The XML Document is modeled as a rooted, unordered, labeled tree. The nodes have a parent-child relationship represented as $u < v$. The type of the node is the label path from root to the node. A word or phrase that is used to begin a search is keyword query. The terms of the keyword query are an XML tag or a leaf. A query match is a node whose text contains the terms which should be meaningful to the user [3]. The term entity refers to any real-world entity which is a node in the XML document. In the XML tree shown in Fig. 1, *conf, article, book* are examples of entity nodes. An entity node represents a node that a user may be interested in exploring.

Exploring LCEA: The notion of LCEA (Lowest Common Entity Ancestor) is introduced by Xiping Liu et.al.[3] in their work. The same notion is considered in the MA algorithm to generate clusters for the keyword query.

Concise Tree: The *concise tree* (CT)in the MA algorithm is a summary tree, obtained by merging the leaf nodes of the original Bibliography tree.

C. The Modified Active Algorithm

It is assumed that CT has all the keywords a user could input as the query.

Let $t_1,\ldots,t_k$ be the keyword terms which the user has submitted and $m_1,\ldots,m_k$ be the set of matches on the Concise Tree.

Step 1: Calculate $E = lcea\ (t_1,\ldots,t_k, CT)$

Step 2: Find $L = Level\ (e, t_1,\ldots,t_k, CT)$ for each **e Є E**

Step 3: Compute $C = cluster\ (L, e, m_1\ldots m_k)$ for each **e Є E**

Step 4: For each **c Є C**, display cluster with distinct **e**

Step 5: Wait for user ***input i*** (input is the cluster the user wants to see as KMP)

Step 6: Display $pattern\ (i, L, m_1,\ldots,m_k)$

Function cluster (L, e, $m_1\ldots m_k$) /* *function to compute the different clusters based on the lcea(E) and level(L) information**/

For each **e Є E**

if (($m_1,\ldots,m_k$) are the term matches and l^e is the level)) where l^e **Є L**

display **e** and $m_1,\ldots,m_k$ as cluster.

Function $pattern(i, E, L, m_1,\ldots ,m_k)$ /* *to display the corresponding KMP* */

Step 1: Assume **i** is the input choice such

that **i Є E,** $m_1,\ldots,m_k$ are the term matches, and l Є L where l is the level information of i .

Step 2: Trace the path from root of the CT to the term matches $m_1,\ldots,m_k$.

Step 3: Merge the paths from the matches $m_1,\ldots,m_k$ to the corresponding ***lcea***.

Step 4: Display ***P***, the path from root to the term (pattern).

The *time complexity* of the Modified Active Algorithm is given by

$$O(l\sum_{i=1}^{k}|m_i| + l\sum_{j=1}^{E}\prod_{i=1}^{k}|m_i^j| + l|P|^2|A|)$$

where l is the depth of the data summary tree, k is the number of keywords and m_1 to m_k are the set of keyword matches in the tree. $|m_i^j|$ is the set of matches in the concise tree of term t_i under a top entity type ***ej***. For a keyword query, the MA algorithm first finds out the matches of each term in the Concise Tree, and evaluates the matches on the Document Tree. Once all the clusters are formed, the clusters with distinct ***lcea*(E)** are displayed. Here, the user input ***i***, where ***i*** represent the cluster the user is interested in exploring, is taken. For *i*, the corresponding KMP is displayed by calling the Pattern function.

IV. Performance Analysis

The performance of the Modified Active algorithm is compared with the performance of the Active algorithm and it is shown that the MA algorithm generates clusters faster than the active algorithm. The cluster generating and the cluster hierarchy constructing algorithms are merged and the result is generated in a hierarchy to provide an enhanced interface for users to view. The XML Concise Tree of Fig. 2 is considered, and the difference in execution times for single, two and three keyword queries are explained. Table 1 illustrates the difference in execution time, in milliseconds, of the Active algorithm and the Modified Active algorithm. It is represented in the graph of figure 3.

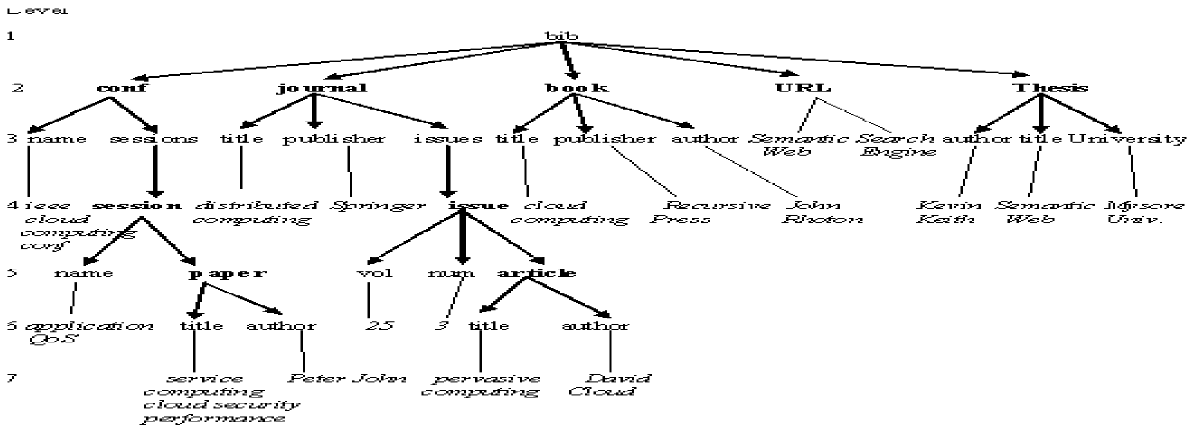
Level
1 bib
2 conf journal book URL Thesis
3 name sessions title publisher issues title publisher author Semantic Web Search Engine author title University
4 ieee cloud computing conf session distributed computing Springer issue cloud computing Recursive Press John Rhoton Kevin Keith Semantic Web Mysore Univ.
5 name paper vol num article
6 application QoS title author 25 3 title author
7 service computing cloud security performance Peter John pervasive computing David Cloud

Figure 2. Extended Bibliography Database

It can be observed that the keywords *Semantic* and *Recursive* are at the same level in the CT and hence the time taken by both the algorithms for searching these keywords is approximately the same. However the keyword *computing* takes more time as this keyword occurs at the lower level on the Concise Tree and more over it exists at different levels throughout the Consice Tree.

Table I. One keyword Comparison

One keyword query	Active Algorithm(Time in msecs)	Modified Active Algorithm(Time in msecs)
a: Semantic	49	25
b: Recursive	53	29
c: Computing	94	55
d: Pervasive	57	57

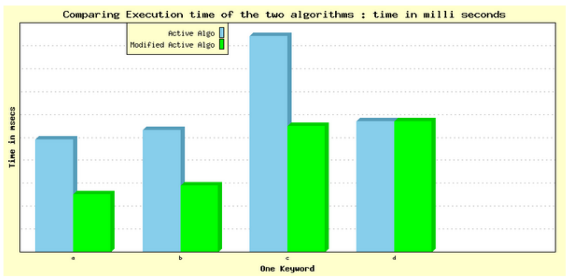

Figure 3. Graph of Comparison Table I

Table 2 below illustrates the difference in execution time, in milliseconds, of the Active algorithm and the Modified Active algorithm for Two Keyword Queries. It is represented in the graph of figure 4. The two keyword query *cloud computing* is located at higher level in the CT and both the terms appear at various levels in the CT. This increases the response time of the query *cloud computing* when compared with the query *Mysore Univ*., as *Mysore Univ*., appears at a lower level when compared with the query *cloud computing*. It can be observed from the table that the response time of the Modified Active algorithm for the same keyword query is comparatively less when compared with the response time of the Active Algorithm.

Table II. Two Keyword Comparison

Two Keyword query	Active Algorithm(Time in msecs)	Modified Active Algorithm(Time in msecs)
a:Cloud Computing	5489	5020
b:Mysore University	1784	1205

c: Cloud Security	2753	1211
d: Peter Cloud	2720	1698
e: Semantic Web	2838	1792

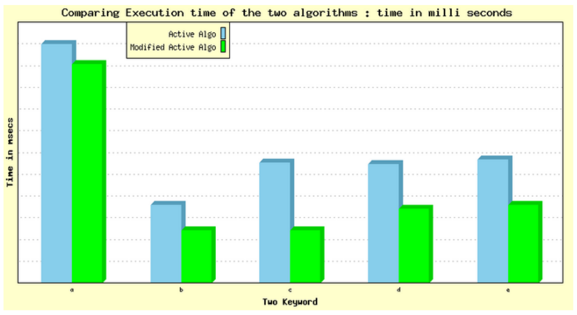

Figure 4. Graph of Comaprison Table II.

Table 3 below illustrates the difference in execution time, in milliseconds, of the Active algorithm and the Modified Active algorithm for three keyword queries. It is represented in the graph of figure 5. It can be observed that the three keyword query with terms *cloud* and *computing* (a and c of table 3) is having a higher response time when compared with keywords that do not have *cloud* and *computing* as keyword terms. This is because of the position at which the keyword terms cloud and computing appear in the CT.

Table III. Three Keywords Comparison

Three keyword query	**Active Algorithm(Time in msecs)**	**Modified Active Algorithm(Time in msecs)**
a: Cloud Computing John	5939	4114
b: Kevin Mysore University	1899	1393
c: Springer Cloud Computing	5288	4514
d: Peter Cloud Security	4815	3381
e: Keith Semantic Web	2554	2124

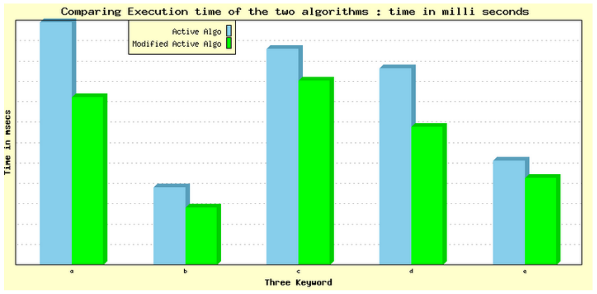

Figure 5. Graph of Comparison Table III.

Fig. 6 illustrates the difference in execution time of the two algorithms for the following one-word, two-word and three-word queries.

a – ieee	e – cloud computing
b - cloud	f – John cloud computing
c - computing	g – security cloud Peter
d – cloud John	h – computing Springer cloud

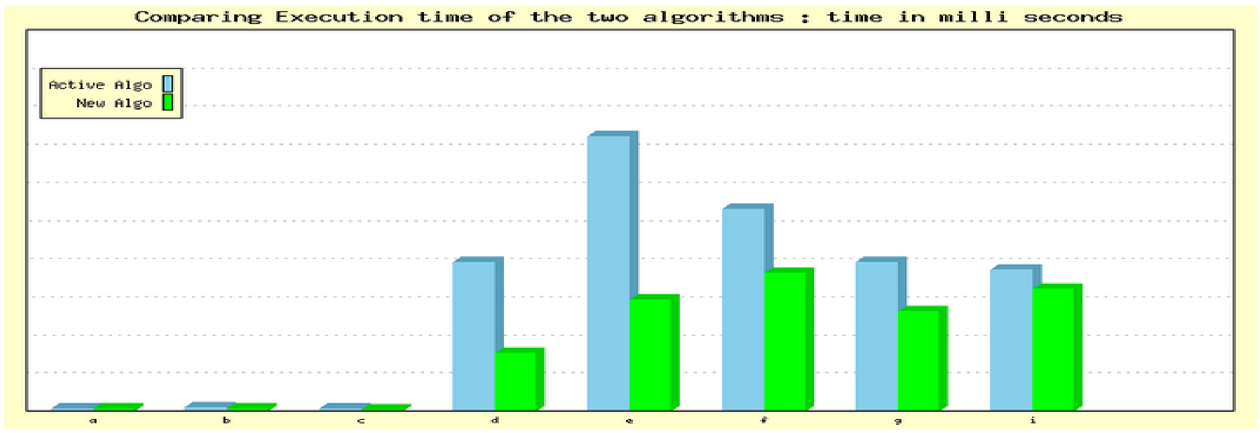

Figure 6. Comparison of execution time for the list of terms

It is concluded that the level information is of primary importance in generating clusters and patterns that match the keywords input by the user. The Active algorithm [3] uses the concept of *lcea, entity nodes* and *pids* whereas the MA algorithm uses the concept of *lcea, entity nodes* and *levels* to derive clusters and patterns.

V. Conclusions and Future Enhancements

The Modified Active Algorithm takes into consideration the level information and entity value while fetching the clusters that match the keyword input by the user. This improves the efficiency of the algorithm in terms of the execution time and response time. As future improvement, the Concise Tree can be built using disk-based inverted index so that searching for the related keyword can become even faster. If the Modified Active algorithm is implemented using a real time database, recall and precision measures can be used to verify its efficiency.

References

[1]. G. Li, J. Feng, J. Wang, L. Zhou, "Effective Keyword Search for Valuable LCAs over XML Documents," *Proceedings of the 16th ACM Conference on Information and Knowledge Management*, 2007.

[2]. Tim Berners Lee, "The Semantic Web," in *Scientific American,* May 2001

[3]. Xiping Liu, Changxuan Wan, Lei Chen, "Returning Clustered Results for Keyword Search on XML Documents," *IEEE Transactions on Knowledge and Data Engineering*, vol. 23, no. 12, December 2011.

[4]. Ziyang Liu , Jeffrey Walker, Yi Chen, "XSeek: A Semantic XML Search Engine Using Keywords", *Proceedings of the VLDB Conference,* 2007

[5]. L. Guo, F. Shao, C. Botev, and J. Shanmugasundaram, "XRANK: Ranked Keyword Search over XML Documents," *Proeedingsof ACM SIGMOD International Conference on Management of Data*, 2003.

[6]. Sara Cohen, Yaron Kanza, Benny Kimelfeld and Yehoshua Sagiv, "Interconnection Semantics for Keyword Search in XML," *Proceedings of the 2005 ACM International Conference on Information and Knowledge Management (CIKM),* 2005.

[7]. A. Schmidt, M. Kersten, and M. Windhouwer, "Querying XML Documents Made Easy: Nearest Concept Queries," *Proceedings of the 17th International Conference on Data Engineering*, 2001.

[8]. Ziyang Liu, Yi Chen, "Answering Keyword Queries on XML Using Materialized Views," *IEEE 24th International Conference on Data Engineering*, 2008

[9]. Y. Huang, Z. Liu, and Y. Chen, "Query Biased Snippet Generation in XML Search," *Proceedings of the ACM SIGMOD International Conference on Management of Data*, 2008.

[10]. Z. Liu, P. Sun, and Y. Chen, "Structured Search Result Differentiation," *Proceedings of the VLDB Endowment*, vol. 2, no. 1, pp. 313- 324, 2009.

[11]. V. Hristidis, N. Koudas, Y. Papakonstantinou, and D. Srivastava, "Keyword Proximity Search in XML Trees," *IEEE Transactions on Knowledge and Data Engineering*, vol. 18, no. 4, pp. 525-539, Apr. 2006.

[12]. Z. Liu, Y. Chen, "Reasoning and Identifying Relevant Matches for XML Keyword Search," *Proceedings of VLDB Endowment*, vol. 1, no. 1, pp. 921-932, 2008.

[13]. Liang Jeff Chen, Yannis Papakonstantinou, "Supporting Top-K Keyword Search in XML Databases," *IEEE International Conference on Data Engineering*, pp. 689-700, 2010

[14]. W. Lian, D.W.-L. Cheung, N. Mamoulis, S.-M. Yiu, "An Efficient and Scalable Algorithm for Clustering XML Documents by Structure," *IEEE Transactions on Knowledge and Data Engineering*, vol. 16, no. 1, pp. 82-96, Jan. 2004.

[15]. Chong Sun, Chee-Yong Chan, Amit K. Goenka, "Multiway SLCA-based Keyword Search in XML Data," *Proceedings of the 16th International Conference on the World Wide Web*, 2007.

[16]. Zhifeng Bao, Jiaheng Lu, Tok Wang Ling and Bo Chen, "Towards an Effective XML Keyword Search," IEEE *Transactions on Knowledge and Data Engineering*, vol. 22, no. 8, pp.1077-1092, August 2010.

.

[17]. Rui Zhou, Chengfei Liu, Jianxin Li, Jeffrey Xu Yu, "ELCA Evaluation for Keyword search on Probabilistic XML Data," in *World Wide Web*, vol. 16, no. 2, March 2013.

[18]. Z. Bao, T.W. Ling, B. Chen, and J. Lu, "Effective XML Keyword Search with Relevance Oriented Ranking," *Proceedings of the 25th International Conference on Data Engineering,* 2009.

[19]. Ch. Lavanya Susanna," Interactive Search over XML Data to Obtain Top-K Results," *International Journal of Soft Computing and Engineering (IJSCE),* vol. 3, Issue 3, 2013

[20]. C. Carpineto, S. Osi_nski, G. Romano, and D. Weiss, "A Survey of Web Clustering Engines," in *ACM Computing Surveys*, vol. 41, no. 3, pp. 1-38, 2009.

[21]. S. Flesca, G. Manco, E. Masciari, L. Pontieri, and A. Pugliese, "Fast Detection of XML Structural Similarity," *IEEE Transactions on Knowledge and Data Engineering*, vol. 17, no. 2, pp. 160-175, Feb. 2005.

[22].http://www.cs.washington.edu/research/xmldatasets.

[23].http://www.sigmod.org/publications/sigmod-record/xmledition.

[24]. http://www.xml-benchmark.org/, 2011.

[25]. A.V. Aho, J.E. Hopcroft, and J.D. Ullman, "On Finding Lowest Common Ancestors in Trees," *Proceedings of the Fifth Annual ACM Symposium on Theory of Computing* , 1973