Keyword Search in XML Database with Relevance Ranking and Maintaining Stored Websites

Anitha J[1], K. Jeyalakshmi[2]

Research Scholar, PG and Research Department of Computer Science, Hindusthan College of Arts and Science, Hindusthan Gardens, (Behind Nava India), Avanashi Road, Coimbatore - 641 028, India

Assistant Professor, PG and Research Department of Computer Science, Hindusthan College of Arts and Science, Hindusthan Gardens, (Behind Nava India), Avanashi Road, Coimbatore - 641 028, India

Abstract: ***Keyword search in XML Database is to provide access on XML database by overcoming keyword ambiguity. Here users are allowed to search on XML database using keyword search like Text Databases. A novel IR style approach which well captures XML's hierarchical structure, and works well on pure keyword query independent of any schema information of XML data. A search engine prototype called XReal is implemented to achieve effective identification of user search intention and relevance oriented ranking for the search results is implemented. Here users also allowed storing their frequently viewed websites on their account.***

Keywords: User intention search, Relevance result ranking, rank based results, keyword relevance matching**.**

1. Introduction

The extreme success of Web Search Engines makes Keyword Search the most popular search model for ordinary users. As XML is becoming a standard in data representation, it is desirable to support keyword search in XML database. It is a user friendly way to query XML databases since it allows users to pose queries without the knowledge of complex query languages and the database schema. XML keyword search exploit the statistics of underlying XML database to address Search Intention Identification, Result Retrieval, and Relevance Oriented Ranking as a single problem. For this combination a novel IR style approach is proposed to captures XML's hierarchical structure, and works well on pure keyword query independent of any schema information of XML data. A search engine prototype called XReal is implemented to achieve effective identification of user search intention and relevance oriented ranking for the search results in the presence of keyword ambiguities. Thus XReal may introduce answers that are either;

- Relevant to user search intention, or
- Answers that may be meaningful or
- Informative enough.

Users can add their website preferences by activating and inactivating it. By recording website preferences frequently visited websites are ranked by analyzing web search history. Visit Counts are also displayed to users. Thus user can activate are inactivate a website by seeing their search counts.

2. Challenges and Issues

Effectiveness in terms of result relevance is the most crucial part in keyword search, which can be summarized as the following three issues in XML field:

Issue 1: It should be able to effectively identify the type of target node(s) that a keyword query intends to search for. We call such target node as a *search for nodes*.
Issue 2: It should be able to effectively infer the types of condition nodes that a keyword query intends to search via. We call such condition nodes as *search via nodes*.
Issue 3: It should be able to rank each query result in consideration of the above two issues.
The first two issues address the search intention problem, while the third one addresses the relevance-based ranking problem.

Regarding to Issue 1 and Issue 2, XML keyword queries usually have ambiguities in interpreting the search for node(s) and search via node(s), due to three reasons.

- A keyword can appear both as an XML tag name and as a text value of some other nodes.
- A keyword can appear as the text values of different types of XML nodes and carry different meanings.
- A keyword can appear as an XML tag name in different contexts and carry different meanings.

Regarding to Issue 3, the search intention for a keyword query is not easy to determine and can be ambiguous, because the search via condition is not unique; so, how to measure the confidence of each search intention candidate, and rank the individual matches of all these candidates are challenging.

To overcome these challenges, our project follows three step procedures as following:

- First identifies the search intention of the user, i.e., to identify the most desired search for node type. In particular, it first collects all distinct node types in XML document. Then, for each node type, we compute its confidence to be a search for node through, and choose the one with the maximum confidence as the desired search for node type Tfor.

- Second, for each search for node candidate Nfor, it computes the XML TF*IDF similarity between n and the given keyword query. We maintain a rankedList to contain the similarity of each search for node candidate. Nfor is initially set to the first node of type Tfor in document order. The computation of XML TF*IDF similarity between an XML node and the given query is computed recursively in a bottom-up way: Then, it computes the similarity between current Nfor and the query, inserts a pair (Nfor) into rankedList, and moves the cursor to next Nfor by calling function getNext and calculates the similarity of next Nfor in the same way. Function isAncestor(N1;N2) returns true if N1 is an ancestor of N2.
- Third, it collects the results in rankedList, where their relevance difference is less than a specified threshold, and computes their popularities by calling Function and adjusts their ranking positions in ranked List.

3. Search

- User can search the reviled detail in search enjoin based on the key beck and will result the relining data's from xml database using popularity algorithm.
- If both the search module and the menu module are enabled, from the *menus* page (*administer* >> *menus*) you can enable on the Navigation Menu the item *Search*. Below keyword query is used for search.
- $C_{for}(T,q) = \log_e(1+\pi_{bcq \int^{T}_{k)} *} T^{depth(T)}$
- You can also place a link to *Search* among your site's primary and secondary links or on any other menu as well.

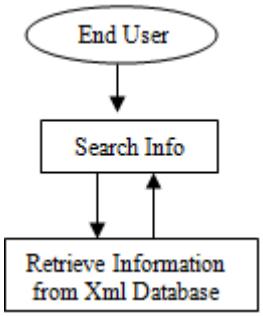

Figure 1: Data Flow Diagram for Search Module

Figure 2: Searching with Keyword

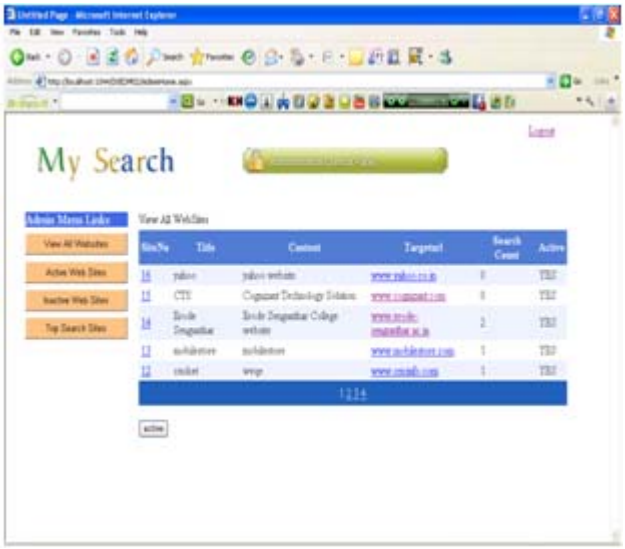

Figure 3: Relevance Ranking

4. Maintaining Stored Websites

In this Module websites can be added or removed from our application. Thus a faster way of extracting frequently searched websites is easily done. Here websites are monitored as follows.

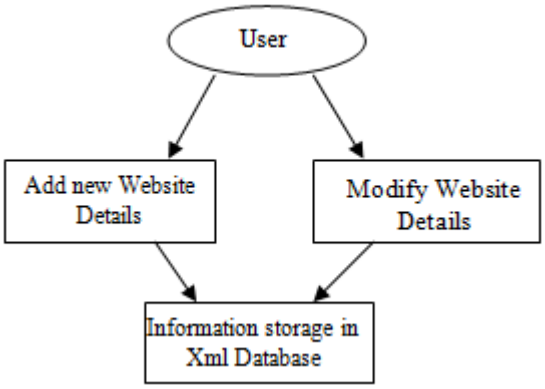

Figure 4: Data Flow Diagram for Adding or Modifying Websites

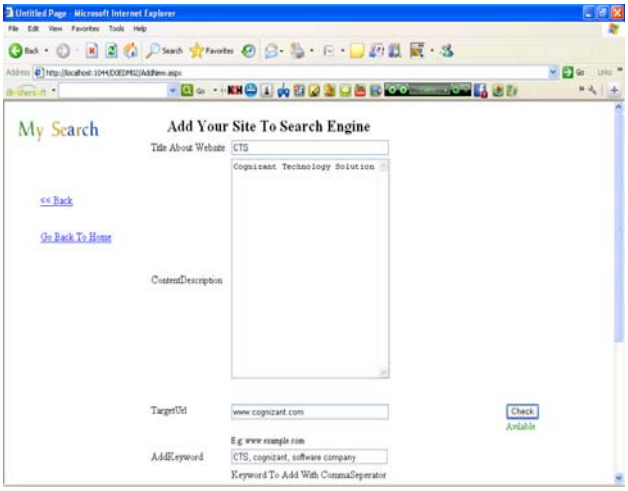

Figure 5: Website Registration

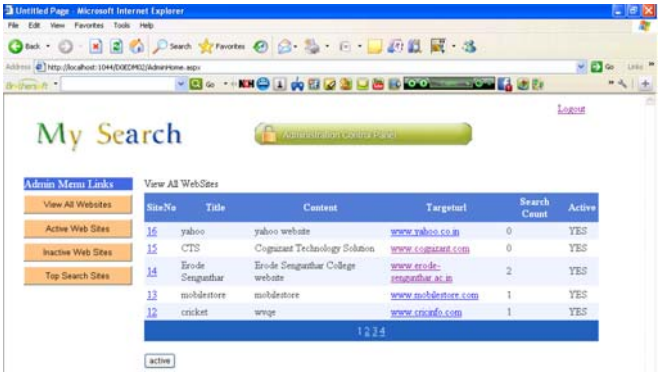

Figure 6: Viewing Registered Websites

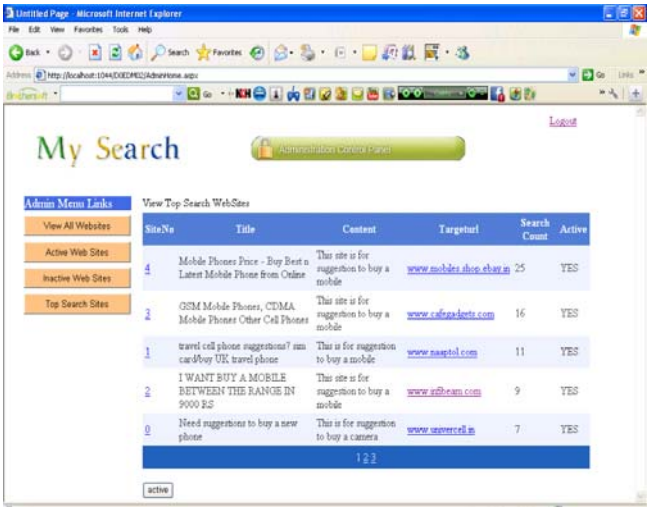

Figure 7: Viewing Top Searched Sites

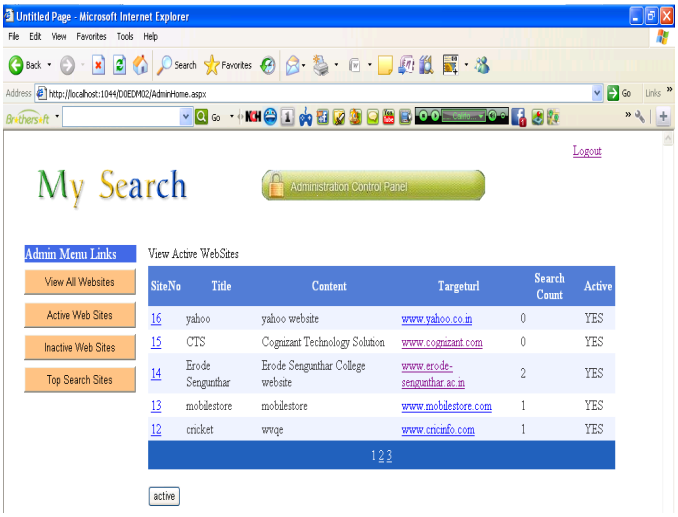

Figure 8: Viewing Active Websites

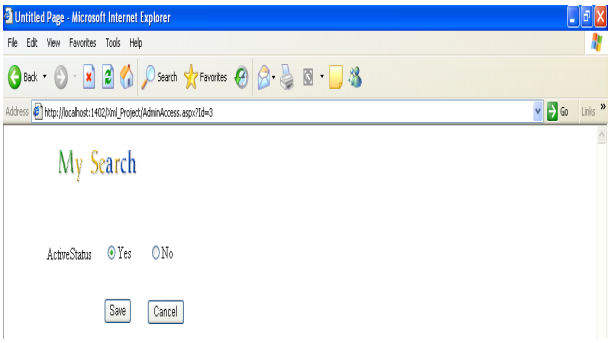

Figure 9: Activating or Inactivating Websites

5. Other Related Works

5.1 Keyword Search Using SLCA

One widely adopted approach so far is to find the Smallest Lowest Common Ancestor (SLCA) of all keywords. Each SLCA result of a keyword query contains all query keywords but has no sub tree which also contains all the keywords. In particular, regarding to search intention such as effectively identify the type of target node(s) that a keyword query intends to *search for* and effectively infer the types of condition nodes that a keyword query intends to *search via*. SLCA does not work well in search intention problem. SLCA may introduce answers that are either;

- Irrelevant to user search intention, or
- Answers that may not be Meaningful or
- Informative enough.

5.2 Supporting Top-K Keyword Search in XML Databases

In this method a series of join-based algorithms that combine the semantic pruning and the top-K processing to support top-K keyword search in XML databases. The algorithms essentially reduce the keyword query evaluation to relational joins, and incorporate the idea of the top-K join from relational databases. Extensive experimental evaluations show the performance advantages of our algorithms.

5.3 Using Structural Information in XML Keyword Search Effectively

In this method for data-centric XML, coherency ranking (CR) is proposed. CR is a domain- and database design-independent ranking method for XML keyword queries that is based on an extension of the concepts of data dependencies and mutual information. With coherency ranking, the results of a keyword query are invariant under a class of equivalency-preserving schema reorganizations. We analyze the way in which previous approaches to XML keyword search approximate coherency ranking, and present efficient algorithms to process queries and rank their answers using coherency ranking. Our empirical evaluation with two real-world XML data sets shows that coherency ranking has better precision and recall and provides better ranking than all previous approaches.

5.4 Keyword Proximity Search on XML Graphs

Keyword proximity search is a user-friendly information discovery technique that has been extensively studied for text documents. A keyword proximity search does not require the user to know the structure of the graph, the role of the objects containing the keywords, or the type of the connections between the objects. The user simply submits a list of keywords and the system re- turns the sub-graphs that connect the objects containing the keywords.

5.5 Efficient SLCA-Based Keyword Search on XML Databases: An Iterative-Skip Approach

Keyword search is a popular way to discover information from XML data. To return meaningful results, SLCA (smallest lowest common ancestor) has been proposed to identify interesting data nodes. Since the SLCAs are obtained from a series of intermediate LCAs, it can often incur many unnecessary LCA computations even when the size of final results is small. In this paper, an Iterative-Skip approach is proposed to address this challenge. At first relativity study between SLCA candidates are made and to that a series of properties are developed. Then based on these properties, we design a novel skip strategy to skip more SLCA computations.

5.6 Multi-way SLCA-based Keyword Search in XML Data

In this paper, a useful search paradigm is proposed to support keyword search beyond the traditional AND semantics to include both AND and OR Boolean operators as well. We first analyze properties of the LCA computation and propose improved algorithms to solve the traditional keyword search problem (with only AND semantics). Then our approach is extended to handle general keyword search involving combinations of AND and OR Boolean operators. The effectiveness of our new algorithms is demonstrated with a comprehensive experimental performance study.

5.7 Efficient Keyword Search for Smallest LCAs in XML Databases

The proposed keyword search returns the set of smallest trees containing all keywords, where a tree is designated as “smallest” if it contains no tree that also contains all keywords. Their come with Indexed Lookup Eager algorithm that exploits key properties of smallest trees in order to outperform prior algorithms by orders of magnitude when the query contains keywords with significantly different frequencies and the Scan Eager Variant is tuned for the case where the keywords have similar frequencies. They also analytically and experimentally evaluate two variants of the Eager algorithm, along with the Stack algorithm and present the XKSearch system which system, which utilizes the Indexed Lookup Eager, Scan Eager and Stack algorithms and a demo. Finally, they extend the Indexed Lookup Eager algorithm to answer Lowest Common Ancestor (LCA) queries. In order to efficiently compute Smallest LCAs in XML Database, the authors propose Indexed Lookup Eager algorithm in case the keyword search includes at least one low frequency keyword along with high frequency keywords.

6. Conclusion

The problem of effective xml keyword search which includes the identification of user search intention and result ranking in the presence of keyword ambiguities is seen. We utilize statistics to infer user search intention and rank the query results. in particular, we define xml tf and xml df, based on which we design formulae to compute the confidence level of each candidate node type to be a search for/search via node, and further propose a novel xml tf*idf similarity ranking scheme to capture the hierarchical structure of xml data. Lastly, the popularity of a query result is considered to handle the case that multiple results have operable relevance scores. In future, we would like to extend our approach to handle the xml document conforming to a highly recursive schema as well.

References

[1] S. Amer-Yahia, L.V.S. Lakshmanan, and S. Pandit, “Flexpath: Flexible Structure and Full-Text Querying for XML,” Proc. ACM SIGMOD Conf., 2004.
[2] Z. Bao, B. Chen, T.W. Ling, and J. Lu, “Effective XML Keyword Search with Relevance Oriented Ranking,” Proc. IEEE Int’l Conf.Data Eng. (ICDE), pp. 517-528, 2009.
[3] D. Carmel, Y.S. Maarek, M. Mandelbrod, Y. Mass, and A. Soffer, “Search XML Documents via XML Fragments,” Proc. ACM SIGIR, pp. 151-158, 2003.
[4] S. Cohen, Y. Kanza, B. Kimelfeld, and Y. Sagiv, “Interconnection Semantics for Keyword Search in XML,” Proc. ACM Int’l Conf. Information and Knowledge Management (CIKM), pp. 389-396, 2005.
[5] S. Cohen, J. Mamou, Y. Kanza, and Y. Sagiv, “XSEarch: A Semantic Search Engine for XML,” Proc. Int’l Conf. Very Large Data Bases (VLDB), pp. 45-56, 2003.

[6] N. Fuhr and K. Großjohann, "XIRQL: A Query Language for Information Retrieval in XML Documents," Proc. ACM SIGIR, pp. 172-180, 2001.
[7] L. Guo, F. Shao, C. Botev, and J. Shanmugasundaram, "XRANK: Ranked Keyword Search over XML Documents," Proc. ACM SIGMOD Conf., 2003.

Websites:
[1] Berkeley DB, http://www.sleepycat.com
[2] http://www.cs.washington.edu/research/xmldatasets
[3] http://www.xml-benchmark.org

Author Profile

Anitha J, MCA, M.Phil (Computer Science), received her B.C.A degree in St.Joseph College for Women, Tirupur and M.C.A degrees in Erode Sengunthar Engineering College, Erode in 2008 and 2011, respectively. Now she is pursuing M.Phil computer Science in Hindusthan College of Arts and Science, Coimbatore. She is currently working as an Assistant Professor in Info Institute of Engineering, Coimbatore in the Computer Applications Department. She has two years of experience in Teaching.

Mrs K. Jeyalakshmi, MCA, M.Phil, (Ph.D), received her B.Sc (Computer Science) degree in Vellalar College for Women, Erode, M.C.A degrees in Madurai Kamaraj University College, Madurai and M.Phil (Computer Science) in KG College of Arts and Science in 1999, 2002 and 2004, respectively. Now she is pursuing Phd in Bharathiar University. Currently she is working as an Assistant Professor in Hindusthan College of Arts and Science, Coimbatore. She has decade of experience in Teaching.