XML based Keyword Search

G. Seethalakshmi
Assistant Professor
PSG College of Technology
Coimbatore-641 004

J. Swathi
Assistant Professor
PSG College of Technology
Coimbatore-641 004

ABSTRACT
The success of information retrieval style keyword search on the web leads to the emergence of XML based keyword search. The text database and XML database differences leads to three new challenges: 1) The users search intention is to be identified, i.e., the XML node types that user wants to search for and search via is identified. 2) The similarities in tag name, tag value and the structure of tags are identified. 3) New scoring function is needed to estimate the output of the search results (XML document) relevance to the given query. However, these challenges cannot be addressed by the existing system, which results in low quality results in terms of query relevance. In this paper, an IR-style approach is proposed which basically utilizes the statistics of underlying XML data to address these challenges. First, specific guidelines that a search engine should meet in both search intention identification and relevance oriented ranking for search results is proposed. Then, based on these guidelines, a novel XML TF*IDF ranking strategy to rank the individual matches of all possible search intentions is proposed.

Keywords
XML, Search, ranking.

1. INTRODUCTION
The extreme success of the web search engines makes the key word search a more popular search model for all computations. The success in the keyword search and its disadvantages lead to the emergence of the XML keyword search in XML databases. The XML databases are user friendly to work with since it does not need any knowledge of complex query languages and the database schema.

The most crucial part in the keyword search is in terms of the effectiveness in terms of the result relevance to the user query, which can be addressed with the following three issues.

1. It should effectively identify the target node that a user's keyword query intends to search for.
2. It should effectively identify the type of nodes that a keyword query intends to search via.
3. It should rank the query results based on users request correctly.

2. RELATED WORK
In Reasoning and Identifying relevant matches for XML Keyword search, four stages were used in realizing the semantics of the given query. First the matches to each keyword in the query are to be retrieved. Then smallest lowest common ancestor (SLCA) nodes are to be computed from the matches. All the keyword matches are to be grouped according to their SLCA ancestor, so that the group will be having a SLCA node t and the set of matches that are descendants of t. When a set of keyword inputs are given, the list of data nodes sorted in the order of the ID will be matched with the input and the relevant matches will be given.[1] XKSearch [2]. XKSearch proposes a concept of Smallest Lowest Common Ancestor (SLCA). For a query Q on data D, an XML node is an SLCA if it contains matches to all keywords in Q in its subtree, and none of its descendants does. For each SLCA, all its descendant matches are considered as relevant to Q. XRank [3]: In XRank, an XML node is the root of a query result if it contains at least one occurrence of each keyword in its subtree, after excluding the occurrences of the keywords in its descendants that already contain all the keywords. All descendant matches of such nodes are considered relevant.

XSeek [4] generates the return nodes which can be explicitly inferred by keyword match pattern and the concept of entities in XML data. However, neither the ranking problem nor the keyword ambiguity problem is addressed. Besides, it relies on the concept of entity (i.e., object class) and considers a node type t in DTD as an entity if t is "*"-annotated in DTD. As a result, customer, interest, and art in Fig. 1 are identified as object classes by XSeek. However, it causes the multivalued attribute to be mistakenly identified as an entity, causing the inferred return node not as intuitive as possible. In fact, the identification of entity is highly dependent on the semantics of underlying database rather than its DTD, so it usually requires the verification and decision from database administrator. Liu and Chen [1] propose an axiomatic way to judge the completeness and correctness of certain keyword search semantics.

Data from the XML documents can be extracted using any of the query languages such as XQUERY. But these query languages are not sufficient for XML keyword search, since it will not provide any of ranking mechanisms. These languages combine the SQL type logical conditions and the regular expression pattern mechanism. The result of the query will be based on Boolean retrieval which does not support relevance ranking [5].

In Schema free XQuery [6] the problem of the keyword search is addressed using the Meaningful Lowest Common Ancestor (MLCAs) for finding the related nodes within the XML document. Automatically calculating the MLCAs and expanding the ambiguous tags will lead to the addition of a new functionality to the XQuery [7] and enable the users to take advantage of querying the XML documents without the knowledge of the document structure.

Keyword Proximity search that conforms to the XML Schema is proposed in the Xkeyword. It needs to compute the candidate networks and thus is constrained by schemas [8].

Most of the XML query languages lack most IR-related features, such as weighting and ranking, relevance-oriented search, data types with vague predicates, and semantic relativism. XIRQL integrates these features by using ideas from logic-based probabilistic IR models, in combination with concepts from the database area. For processing XIRQL queries, path algebra is used, that also serves as a starting point for query optimization. XIRQL queries will be transformed into path algebra expression and then processes this expression

which results in a better result and also provides query optimization. [9]

Query substitution is a method of substituting the users query with a new query. The new query generated will be closely related to the original query. [10]

The tree structure of the XML documents has been proposed in the nearest concept queries [11] which equip the user with a powerful tool called the meet operator. This operator helps the user to query the database without requiring the knowledge of the tags and hierarchies. This approach is based on computing the lowest common ancestor of nodes in the XML syntax tree: e.g., given two strings, the nodes whose offspring contains these two strings is to be taken. The meet operator lets users to query XML databases without knowing the schema or structure of that content.

3. TF*IDF COSINE SIMILARITY

TF*IDF (Term Frequency * Inverse Document Frequency) similarity is one of the most widely used approaches to measure the relevance of keywords and document in keyword search over flat documents. But if the original TF*IDF similarity is adopted there are 2 problems 1) If the structure in customer nodes is not considered, customer A may have a lower rank than B if A happens to have more keywords in its subtree (analog to long document in IR) than B. 2) Even worse, suppose a customer C is not interested in "art" but has address in "art street." If C has less number of keywords than A and B in the underlying XML data, then C may have a higher rank than A and B.

These disadvantages can be overcome by adopting the following data model which includes the Node type, Data node and the structure node.

i) Node type of a node n in an XML document is the prefix path from root to n. Two nodes are of

the same node type if they share the same prefix path.

ii) The text values that are contained in the leaf node of XML data and have no tag name are defined as data node.

iii) An XML node labeled with a tag name is called a structural node. A structural node that contains other structural nodes as its children is called an internal node; otherwise, it is called a leaf node.

4. KEYWORD SEARCH

4.1. Inferring the Node type that is to be Searched for in the XML Database

Given a query k, the node type n can be selected by considering

1. Whether the node selected is related to all the keywords k given in the query, there should be some n type nodes which have k in their sub trees.

2. XML nodes n should be informative enough to have enough information.

3. XML node n should not contain too much of irrelevant information

Consider an XML database for an online bookstore in Figure1 as example.

If "Customer interests art" is given as the query, the given query will be searched in the XML database and whichever path is having the maximum relevance to the given query that path will be selected as the search for node. For the above query search for node will be customer since the number of sub trees rooted at customer has all the keywords in the given query. Here it was assumed that each keyword has at least one occurrence in the XML document being queried.

Thus with the confidence of each node type being desired, the one with the highest confidence is selected. If more than one node has comparable confidence level then either the user can offered a choice to decide the type of node or the system has to check for the convincing candidate node.

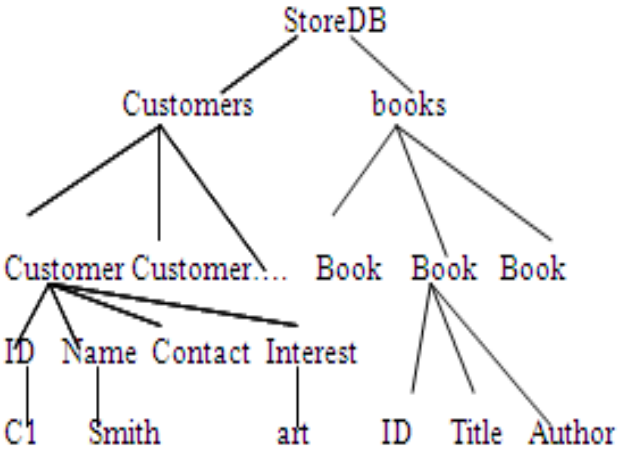

Figure 1. XML database for online book store

4.2 Inferring the Node Type via which the Keyword is to be Searched in the XML Database

If the search for case is considered the node type selected should be related to all the keywords given in the query. But here in search via case it is enough for a node type to have high confidence as the desired search via node if it is closely related to some keywords, because a query may intend to search via more than one node type.

Consider Fig 1. The user can search for a customer with name smith and interest fashion by giving the query as "name smith interest art". In this case the system should effectively identify that name and interest are the node types to search via.

Therefore the confidence C_{via} (n,k) of the node type to search via is calculated by

$$C_{via}(n, k) = \log_e (1+\sum_{I \varepsilon k} f_i^n)$$

After the confidence is being calculated the result should be incorporated into XML TF*IDF similarity to provide results of high quality.

4.3 Finding Keyword Co-Occurrence

Although statistics provide a way to compute the confidence of the structural node type to search via, it alone is not sufficient to infer the likelihood of an individual data node to search via for a given keyword in the query.

In order to overcome this, for every data matching query node q in XML data in order to capture the co occurrence of the keyword k_t matching the node type of the ancestor node of q and keyword w matching a value in v in both query and XML data respectively, the following instances where identified.

1. The In-Query Distance $Dist_k(k,k_t,w)$ between keyword w and node type k_t in a query k is defined as the absolute value of the position distance between k_t and w in k; otherwise, $Dist_k(k,k_t,w) = \infty$.
2. The Structural distance $dist_s$ (k,v,k_t,w) between k_t and w w.r.t to a data node v is defined as the depth distance v and the nearest k_t typed ancestor node of v in XML node.
3. The value type distance $dist(k,v,k_t,w)$ between k_t and w w.r.t a data node v is defined as

   Max $(Dist_k(k,k_t,w), dist_s(k,v,k_t,w))$

In general, the smaller the value of dist (k,v,k_t,w) is, it is more likely that k intends to search via the node v with a value matching keyword w. The confidence of a data node v as the node to search via with respect to a keyword w appearing in both query k and v can be defined as follows:

$$C_{via}(k,v,w) = 1 + \sum_{k_t \varepsilon k \cap anctype(v)} 1/\, dist_s(k,v,k_t,w)$$

Consider the query k "customer name rock interest". Let n3and i3 represent the data nodes under name (i.e., Art Smith) and interest (i.e., rock music) of customer C3. Similarly, let n4 and i4 be the data nodes under name and interest of customer C4. Now, the in-query distance between name and Art is 3, i.e., $Dist_k(k,name,art)=3$; $dist_s(k,n3,name,art)=1$; as a result Dist(k, n3, name,Art)= 3 and $C_{via}(k,n3, Art) = 4/3$. Similarly, C_{via} (k, i3, Rock) = 1; C_{via} (k, n4, Rock) = 2; and C_{via} (k, i4, Art) = 2.

From the above inference it was found that the two predicates of customer C4 have a larger confidence to be searched via than those of customer C3. Intuitively, C4 should be more preferred than C3 as the result of q.

5. RELEVANCE ORIENTED RANKING

The similarity value between the internal node a and a keyword query q is computed and then it recursively computes the similarities between the internal node n and the query q based on the similarity value of each child c of n. For this the term frequency and the inverse document frequency are to be calculated and based on this the ranking will be done.

6. CONCLUSION

In this paper, the problem of effective XML keyword search which includes the identification of the similarity and the semantics of the users query has been introduced. The similarity and the semantics of the users query will be found by considering the search for node which will find the sub tree of the XML database under which the search is to be made and the search via node which in turn will specify the node via which the keyword is to be searched under the XML database. Effective XML TF*IDF ranking which has advantage over the traditional TF*IDF has been proposed. This ranking will find the similarity value between the internal node and the given keyword query. This computation will be recursively done and the similarity is calculated. Once if the similarities are calculated the term frequency and inverse document frequency are to be calculated and based on that calculated value ranking is to be done. Since the relevance oriented ranking has been done it will fetch the most relevant keywords. But this existing system cannot give the relevant results when a negative query with NOT semantics is given.

In future the above system can be extended to handle the AND and NOT semantics. Most of the search engines purely rely on the metadata of image which in turn results in a lot of garbage in the search result. In order to over this metadata based search this simple XML based keyword search with XML can also be extended to the content based image retrieval. The content based image retrieval will analyze the contents of the image rather than the metadata such as tags or keywords associated with the image. The content may refer to color, shape, texture or any other information of that image. Also this keyword search can be extended for structure and content based retrieval of XML documents which can retrieve XML documents based on both the document structure and the image content.

7. REFERENCES

[1] Ziyang Liu Yi Chen, "Reasoning and Identifying Relevant Matches for XML Keyword Search", Proceedings of the VLDB Endowment Volume 1 Issue 1, August 2008.

[2] Y. Xu and Y. Papakonstantinou. Efficient Keyword Search for Smallest LCAs in XML Databases. In SIGMOD, 2005.

[3] L. Guo, F. Shao, C. Botev, and J. Shanmugasundaram. XRANK: Ranked Keyword Search over XML Documents. In SIGMOD, 2003.

[4] Z. Liu and Y. Chen, "Identifying Meaningful Return Information for XML Keyword Search," Proc. ACM SIGMOD Conf., 2007.

[5] Cohen.S, Mamou.J, Kanza.Y, and Sagiv.Y, 'XSEarch: A semantic search engine for XML', In Proc. Of VLDB Conference, pp. 45–56. 2003

[6] Yunyao Li, Cong Yu, H. V. Jagadish, "Schema-Free XQuery" Proceedings of the 30th VLDB Conference, Toronto, Canada, 2004

[7] D. Chamberlin. XQuery: An XML query language. IBM System Journal, 41:597–615, 2003.

[8] V. Hristidis, Y. Papakonstantinou, and A. Balmin, "Keyword proximity search on XML graphs," in ICDE, 2003, pp. 367–378.

[9] Norbert Fuhr, Kai Großjohann, "XIRQL: A Query Language for Information Retrieval in XML Documents SIGIR'01, September 9–12, New Orleans, Louisiana, USA

[10] Rosie Jones, Benjamin Rey and Omid Madani, Wiley Greiner, "Generating Query Substitutions" WWW 2006, May 23–26, 2006, Edinburgh, Scotland ACM 1-59593-323-9/06/0005.

[11] Albrecht Schmidt, Martin Kersten, Menzo Windhouwer "Querying XML Documents Made Easy: Nearest Concept Queries" proceedings of 17th International conference on data engineering, 2001.