**International Journal of Engineering Research and Development**
**e**-ISSN: 2278-067X, **p**-ISSN : 2278-800X, www.ijerd.com
Volume 5, Issue 9 (January 2013), PP. 01-06

# Schema and Design Free Keyword Search Interfaces for XML Databases

M.Rammohan Rao[1], M.Brahmaiah[2], E.Ramesh[3], V.Rajesh[4]
[1]Asst. Prof. Sri Vatsavai Krishnamraju college Of Engineering And Technology
[2,3]Asst. Prof. in RVR&JC College of Engineering
[4]Mtech scholar, Dept of cse, RVR&jc college of engineering

**Abstract:-** Now a days XML is becoming a standard in data representation and data exchanging. Due to its simplicity and lightweight most of the real time applications adopted XML as a DBto store the frequently updating enterprise data, which have complex schema and design with thousands of xml elements. To retrieve the data from XML databases effectively, most previous information retrieval(IR) approaches used XQuery and XSEarch, which are schema and design based mechanisms and allows only the DB experts for querying data. Lack of knowledgeon the database schema and design can be a significant obstacle for common users who want to access the information in an XML database.Inspired by the great success of information retrieval (IR) style keyword search on the web, keyword search on XML has emerged recently, which is schema independent approach and allows all the user to query XML databases by using keyword combination without the knowledge of complex query languages and the database schema. In this paper we propose the Schema and Design Free Keyword search Interfaces (SDFKI) to address the above dependency problems and achieves the higher result relevance and scalability. SDFKI not only return relevant results, it will overcome the dataredundancy problem in XML databases and uses a novel XRank ranking strategy to display the results in ranked order. Lastlyextensive experiments have been conducted to show the effectiveness of our approach.

**Keywords:-** Schema and Design Free Keyword search Interfaces(SDFKI), Information-Retrieval(IR), Database Schema, Database Design, XRank.

## I. INTRODUCTION

In recent years XML databases gradually becoming a standard to store and represent the real world applications data.An XML database is a data persistence software system that allows data to be stored in XML format. These data can then be queried, exported and serialized into the desired format. XML databases are usually associated with document-oriented databases.To retrieve the data from XML databases efficiently some XML querying mechanisms [1] also introduced along with the XML databases. Traditionally there are two main approaches to retrieve the data from XML databases: the structured query approach and the keyword-based approach. These both can have their own advantages and disadvantages.Fully structured query approach (e.g., XQuery [3]) works effectively with the structure information, can convey complex semantic meaning in the query, and therefore can retrieve precisely thedesired results. However, if the user does not know the schema of database, it is difficult to write the right query. Even if the user does know the schemas, when data isto be amalgamated from multiple sources with differentschemas, it typically will not be possible to write a singlequery applicable to all sources; rather, multiple querieswill have to be written, a process that is complex and error-prone. Most of the common users they don't know the XML database schema, design and how to query it because of their lack knowledge on databases. Many databases are equipped with the little documentation which is not enough to understand the database design and users don't like to read documentation and prepare the schema dependent structured queries.

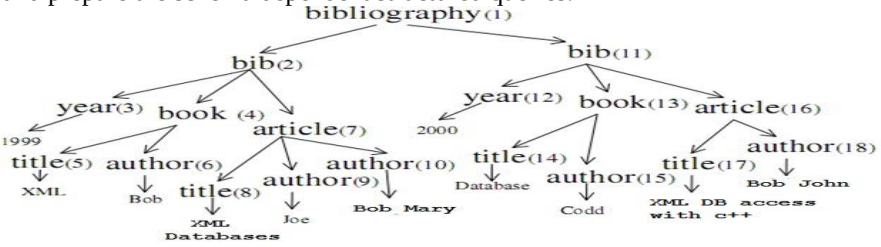


**Fig 1.** A Sample normalized Bibliography XML DataBase.

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[2],[4],[7] 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. Schema and Design Free Keyword search Interfaces (SDFKI) have been proposed as solutions to the above search problems for XML DBs. With SDFKIs, users do not need to knowschema details or a query language. For example, suppose auser wants to find the papers that Bob published aboutXML in the DB fragment in Fig. 1. The user submitsquery Bob XML. The SDFKI returns the answer, whichis the paper at node 6.Database administrators always use various normalization techniques to eliminate the redundancy in databases. Our Keyword search approach can be performed on the normalized database structures in order to overcome the redundancy problem and to retrieve the most relevant elements as results. Because of this SDFKI is schema independent approach we can adopt the same keyword search interface for any XML database to retrieve the results regardless of their schema implementations and updating. In this paper this feature is called as Schema and Design Independent approach. So our proposed interfaces not only schema but also design independent approaches. Consequently, an important requirement for keyword search is to rank the query results so that the most relevant results appear first. We propose a novel relevance oriented ranking scheme called XRank similarity which can capture the hierarchical structure of XML and resolve ambiguity in a heuristic way. Besides, the popularity of query results is designed to distinguish the results with comparable relevance scores. At last the final ranked list of results will be displayed to the user.

## II. RELATEDWORK

Extensive research efforts have been conducted in XML keyword search to find the smallest sub-structures in XML data that each contains all query keywords in either the tree data model or the directed graph (i.e. digraph) data model. In tree data model, LCA (lowest common ancestor) semantics is first proposed and studied in [12], [10] to find XML nodes, each of which contains all query keywords within its subtrees. Subsequently, SLCA (smallest LCA [11]) is proposed to find the smallest LCAs that do not contain other LCAs in their subtrees. Similar approach has also been taken to apply keyword search in XML documents (e.g., XKeyword [13] and XRANK [14]).Ranking mechanisms have been applied to the search results such that results with received higher relevance are returned to the user first. All such keyword search approaches suffer from two drawbacks: (1) they do not distinguish tag name from textual content; (2) they cannot express complex query semantics.

A number of attempts have also been made to support information retrieval style search by expanding XQuery [3] or other structured query languages (e.g.,XXL[5] , XIRQL[8] ). These approaches require a user to learn the query semantics and in cases where a user is unaware of the document structure, they do not exploit any document structure. Other approaches (e.g., LOREL and Meet ) created query languages to enable keyword search in XML documents and exploit some structural information that is not specified in the query. The differences between those approaches and ours are that we eliminate any requirement for path expressions, and we exploit the document structure better to identify results that are more meaningful.A recent closely related work is XSEarch [6], which attempts to return meaningful results based on query as well as document structure using a heuristic called interconnection relationship. In XSEarch, two nodes are considered to be semantically related if and only if there are no two distinct nodes with the same tag name on the path between these two nodes (excluding the two nodes themselves). Queries are allowed to specify tag names and attribute value pairs. However, interconnection does not work when two unrelated entities are present in entities of different types. For example, two author nodes may be considered as interconnected, even though one of them belongs to an article node and the other belongs to a book node. Moreover, due to the simple query semantics used, XSEarch suffers from drawbacks similar to keyword search methods: difficulty to express complex knowledge semantics. The MLCAS[9] operator, on the other hand, takes full advantage of well-defined XQuery,and enables the user to take more control of the search results without knowing the document structure.

Keyword-based query can overcome the problems with unknown schema or multiple schemas because knowledge of structure is not required for the query. However, this absence of structure leads to two serious drawbacks. First, it is often difficult and sometimes impossible to convey semantic knowledge in pure keyword queries. Second, the user cannot specify exactly how much of the database should be included in the result. Keyword search enables users to query XML data exploiting whatever partial knowledge of the schema they have. If they know the full schema, they can write regular XQuery. If they do not know the schema at all, they can just specify keywords.In previous work [6], we defined the property of structure independence, analytically determined the worst case structure independence for current QIs, and introduced QIs DA-CR. In this paper, we present a much more extensivestudy of schema and design independence for QIs, overlarger data sets and workloads than our previous work.We also explore the information preservation characteristicsof weak value

structure preserving transformations in thispaper, and explore transformations that slightly change DBvalues. Further, we fully justify XRank ranking approach,and compare SDFKIs to the XQuery approach.

## III. SCHEMA AND DESIGN FREE KEYWORD SEARCH INTERFACES

In this paper, we developed Schema and Design Free Keyword search Interfaces (SDFKI) that enables users to query XML data exploiting whatever partial knowledge(or zero knowledge) of the schema they have. If they know the full schema, they can write regular XQuery. If they do not know the schema at all, they can just specify keywords as intention to search XML data by using the SDFKI approach.
In this section, we discuss the XML data model, some basic definitions, SDFKI and how to retrieve the search intentions of keyword query according to the statistics in XML data.

### 3.1 XML DB model

The eXtensible Markup Language(XML)is a hierarchicalformat for data representation and exchange. An XML document consists of nested XMLelementsstartingwiththerootelement. Each element can have attributes and values,in additionto nested sub elements. We model XML database as a rooted, labeled tree plus a set of edges between XML nodes, such as the one in Figure 1. Our approach exploits the prefix path of a node and its tag name for result retrieval and ranking. We represented our XML database as tree XD = (r, V, E, L, C , A) where V is the set of nodes(XML elements) in the tree, r $\epsilon$ V is the root, E is the set of parent-child edges between members of V , Cis a subset of the leaf nodes of the tree called data(value) nodes, L is a label to each member of V - C and A assigns aString value to each data node. Our DB tree assumes that each node has at most one leaf child node and the siblings are not following any order. Each subtree S1= (rS, VS, ES, Ls, CS , AS)is a tree which formation is like the rS $\epsilon$ r, VS $\epsilon$ V, ES $\epsilon$ E, Ls $\epsilon$ L, CS $\epsilon$ C and AS $\epsilon$ A. we assume every multi-valued node has a grouping node as its parent, as we can easily introduce a dummy grouping node in indexing without altering the data. Note that the existing works [11], [3] rely on DTD while our approach works without any XML schema information.

### 3.2 Basic Definitions

In order to infer the Schema Free keyword search on XML databases our DB tree has the following basic definitions.

Definition 3.2.1 (Node Type)The 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. In our model node (4) and (16) both can have the same prefix path from root node, so the both have same node type. We can use this node type to found different relevant results as per user intention(keywords combination).
Definition 3.2.2: (Data Node) The String values that are contained in the leaf node of XML data and have no tag name isdefined as a data node.
Definition 3.2.3: (Structural Node) 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.

Definition 3.2.4:The pattern concisely represents a maximal set of isomorphic trees (its instances). The pattern can be obtained from the prefix string of any member of the set of instances, by removing the content(value). Pattern P1 is a sub pattern of pattern P2 if each of P1's instances is asubtree of one of P2's instances.

### 3.3Schema and Design Free Keyword search Interfaces

We now define the structure of schema free query model (SDFKI for short) over XML data trees (databases).In Keyword search query user gives a bag of terms (key words) Q=<t1 ….. tn> where each term ti,1 $\leq$ i $\leq$ q, is the label of an attribute (label term) or a keyword (keyword term). Because of users are unaware about the schema of database they will give their intention in the format of keyword query. For example to find the C++ book information which is written by theauthor john, they may give query like"bookjohn C++" , "author john book C++" or some people may give "C++ book by author john". Different people in the world have different formats of intention representation based on their knowledge. We have to develop a robust keyword query interface to infer the intentions of all types of users in the world. Finding the user intention by given keywords is a key problem in keyword search.Inspired by the important role of data statistics in IR ranking, we try to utilize it to resolve ambiguities for XML keyword search in SDFKI, as it usually provides an intuitionistic and convincing way to model and capture human intuitions.

Our SDFKI approach is not depends on any .dtd file information of an XML database in order to retrieve the relevant results, so it called as schema free keyword search interface and this keyword search interface is applicable to all XML databases by converting and normalizing them into XML tree structure to obtain the Design independence. This schema and design free approach first transforms the given XML database

into a normalized XML tree. To normalize XML DB we are using WXS[14] normalization forms which are provided by W3C to Minimize redundancy, Eliminate ambiguity, Facilitate preservation and Allows the rational maintenance of data. On this normalized database (NDB) we can perform the keyword search by using the SDFKI in order to overcome all above keyword search problems.To avoid the ambiguity and to infer the user intention correctlyin keyword search our SDFKI approach internally following three guidelines:
Guideline 1: T is intuitively related to every query keyword in q, i.e. for each keyword k, there should be some (if not many) T-typed nodes containing k in their subtrees.
Guideline 2: XML nodes of type T should be informative enough to contain enough relevant information.
Guideline 3: XML nodes of type T should not be overwhelming to contain too much irrelevant information.

The desired node type to search for is the first issue that a search engine needs to address in order to retrieve the relevant answers, as the search target in a keyword query may not be specified explicitly like in structured query language. Given a keyword query q, a node type T is considered as the desired node to search for based on the above three guidelines.From the first and second guidelines, our search engine matches the user given keyword query Q against the XML database XD and it founds some subtrees(<T1 …Tn> ϵ XD) which have all the keywords ofQ. However the desired sub trees may contain each keyword K of Q, but all the sub trees are not relevant results. For example the given keyword query "Author Bob Title XML" ,our first guideline will result three sub trees T1(book(4)) , T2(article(7)) and T3(article(16)) because of these sub trees can have all the keywords(k) of Q. We know that T1,T2 and T3 may have all keywords of Q, T1 is the user intention because T2 and T3 have more noise data than T1. Displaying results with these noise data will reduce the accuracy, efficiency and reliability of our XML keyword search approach. So now we have to eliminate or reduce the priority of noisy data in search results. To overcome this problem our approach (SDFKI) following the third guideline and identifies the overwhelmed too much irrelevant data from the resulted sub trees by using the below formula:

$$C_{for}(T,q) = \log_e(1 + \prod_{k \in q} f_k^T) * r^{depth(T)}$$

Where C represents the confidence of resulted sub tree T and q is a given keyword query to infer the user intention. The value of k represents a keyword in query q; fTk is the number of T-typed sub trees that contain k as either values or tag names in their structure; r is a reduction factor with range [0,1] and normally chosen to be 0.8, and depth(T) represents the depth of T-typed subtrees in document. Iteration over above formula will give the relevance confidence of each resulted sub tree of our query Q. Based on the confidence value SDFKI eliminate the noisy data results or otherwise reduces the priority of them in resultset. Because of T2 and T3 have more noisy data, T1 will get the first priority than T2 and T3 in our SDFKI approach.

**3.4 Ranking the Result trees**

Ranking is an important factor while displaying the results to user in a meaningful order. Butproviding the sufficient ranks for XML search results is not an easy task because they are different from other document based results in two ways. First, XMLkeywordsearch queriesdonotalwaysreturnentiredocuments,butcanreturn deeplynestedXMLelementsthatcontainthedesiredkeywords.Second,thenestedstructureofXMLimpliesthatthenotio nofrankingisnolongeratthegranularityofadocument,butatthe

granularityofanXMLelement.Finally,thenotionofkeyword proximity is more complex in the hierarchical XML data model.To overcome all the above problems and to provide better ranking for XML keyword search results SDFKI appointed the XRank system. XRank will considerakeywordsearchqueryQ=(k1,k2,…,kn)andthe correspondingresult set R. Now consider a result element v1 ϵ R. We first define the ranking of v1 with respect to one query keyword ki, r(v1, ki), before defining the overall rank, rank(v1, Q). BythedefinitionofR, weknowthatcontains(v1,ki)istruefor everyki, Hence,thereisasequenceofcontainmentedgesofthe form (v1, v2), (v2, v3),… , (vn-1, vn) suchthat vn directly contains ki.

**r(v1,ki) = ElementRank (vn) × decayn-1**

Intuitively, the rankofv1withrespecttoakeywordkiis Element Rank (vn), where vn directly contains ki, scaled appropriately to account for the specificity of the result. When the result element v1 directlycontainsthekeyword(i.e.,v1=vn),therankisjustthe Element Rank of the result element. When the result element indirectly contains the keyword (i.e., v1 ≠vn),therankisscaled down by a factor decay for each level. decay is a parameter that can be set to a value in the range 0 to 1. After finding the ranking for subtree based on each query keyword k we can compute the overall ranking for a query Q which have a set of keywords k1 to kn as follows:

$$R(v_1, Q) = \left( \sum_{1 \le i \le n} \hat{r}(v_1, k_i) \right) \times p(v_1, k_1, k_2, \ldots, k_n)$$

Theoverallrankingisthesumoftheranks withrespecttoeach querykeyword, multipliedbyameasureofkeywordproximity p(v1, k1, k2, …, kn). We currently setthe keyword proximityto be inversely proportionaltothe minimum windowsizethat contains allthequerykeywordsinv1(themaximumvalueofkeyword proximityis1andminimumvalueis0.2).Clearly,other combinationfunctionstoproducetheoverallrankarealso possible with XRank.

## IV. EXPERIMENTAL RESULTS

We performed extensive experimentation with the SDFKI system, which was implemented in Java. The experiments were carried out on a Pentium 4, with a CPU of 1.6GHZ and 2GB of RAM, running the Windows XP operating system. Later we implemented SDFKI using the Research articles XML database [14] and evaluated the system on two aspects: 1) search quality, which is evaluated using both a standard XML benchmark and a heterogeneous data collection; 2) search performance, where we measure the overhead caused by evaluating schema-free query versus the schema-aware query. The quality of a search technique was measured in terms of accuracy and completeness using standard precision and recall metrics, where the correct results are the answers returned by the corresponding schema-aware XQuery. Precision measures accuracy, indicating the fraction of results in the approximate answer that are correct, while recall measures completeness, indicating the fraction of all correct results actually captured in the approximate answer.

### 4.1 Precision, Recall & F-measure

To measure the search quality, we evaluate all queries and summarize two metrics borrowed from IR field: precision and recall Precision measures the percentage of the output subtrees that are desired; recall measures the percentage of the desired sub trees that are output. we obtain the correct answers by running the schema-aware XQuery with an additional manual verification. Form that results we measured the top-100 results to find the precision and recall. We obtained the precision value from search results by using below formula: Precision(p) = [ (no of relevant results ∩ no of returned results) / no of returned results ]
Recall(r) = [ (no of relevant results ∩ no of returned results) / no of relevant results ]
F-Measure = [ 2* (Precision . Recall)/(Precision + Recall) ]

### 4.2 Query Execution Time

In order to check how query execution time is affectedby the semantics of the query and the type of interconnection index used, we generated 1000 random queries for the Research Article Record document. These queries had at most 3 required searchterms and at most 3 optional search terms. The keywords and labels in the queries were drawn randomlyfrom the set of keywords and labels in the Research Articles Record. We executed the queries to determine execution time using either a hashtable or a matrix as theinterconnection index. We ran the queries under bothall-pairs and star semantics. A histogram of the number of milliseconds needed to process a query is presented.However, in all cases, over 80% of the queries ran in under10 milliseconds and over 97% of queries ran in under 100 milliseconds. The average run time for querieswith all-pairs semantics was about 35 milliseconds andwith star semantics was about 634 milliseconds.

**Table 1.** Precision, Recall, F-Measure, Execution Time comparison of SDFKI with other approaches

| Search Technique | Precision | Recall | F-Measure | Execution Time |
|---|---|---|---|---|
| SDFKI | 0.79 | 0.85 | 0.81 | 35 ms |
| MLCA's | 0.71 | 0.68 | 0.69 | 42 ms |
| XQuery | 0.82 | 0.88 | 0.84 | 28 ms |

## V. CONCLUSION

In this paper, we study the problem of Schema and Design free XML keyword search (SDFKI) which includes the identification of user search intention and result ranking in the presence of keyword search ambiguities and lack of schema knowledge to user. We formalized this approach with three guidelines and analyzed and compared the design independence of current keyword search and schema freequery interfaces. We utilize various XML statistics to infer user search intention and used XRank to rank the schema free keyword search query results. SDFKI can enable users to take full advantage against XQuery in querying XML data precisely and efficiently without requiring full or less knowledge of the document schema. At the same time, any partial knowledge available to the user can be exploited to advantage. We have shown that it is possible to express a wide variety of queries in a schema-free manner and have them return correct results over a broad diversity of schema. At last we compared our results against the various search approaches and the results shown that our approach has higher precision and recall than the previous approaches.

## REFERENCES


[1]. http://www.xml-benchmark.org/.
[2]. S. Cohen, Y. Kanza, B. Kimelfeld, and Y. Sagiv. Interconnection semantics for keyword search in xml. In CIKM, pages 389–396, 2005.
[3]. D. Chamberlin. XQuery: An XML query language. IBM System Journal, 41:597{615, 2003.
[4]. V. Hristidis, N. Koudas, Y. Papakonstantinou, and D. Srivastava. Keyword proximity search in XML trees. In TKDE, pages 525–539, 2006.
[5]. A. Theobald and G. Weikum. The index-based XXL search engine for querying XML data with relevance ranking. In Proc. 8th International Conference on Extending Database Technology, pages 477–495, Prague (Czech Republic), March 2002. Springer-Verlag.
[6]. S. Cohen et al. XSEarch: A semantic search engine for XML. In VLDB, 2003.
[7]. Y. Li, C. Yu, and H. V. Jagadish. Schema-free XQuery. In VLDB, 2004.
[8]. N. Fuhr and K. Großjohann. Xirql: A query language for information retrieval in xml documents. In SIGIR, pages 172–180, 2001.
[9]. W. Fan and P. Bohannon, "Information Preserving XML Schema Embedding," TODS, vol. 33, no. 1, 2008.
[10]. E. Elmacioglu and D. Lee, "On Six Degrees of Separation in DBLP-DB and More," SIGMOD Record, 2005.
[11]. Y. Luo, X. Lin, W. Wang, and X. Zhou, "SPARK: Top-k Keyword Query in Relational Databases," in SIGMOD 2007.
[12]. M. Arenas and L. Libkin, "A Normal Form for XML Documents," TODS, vol. 29, no. 1, pp. 195–232, 2004.
[13]. Y. Xu and Y. Papakonstantinou, "Efficient Keyword Search for Smallest LCAs in XML Databases," in SIGMOD, 2005.
[14]. Z. Liu and Y. Chen, "Reasoning and Identifying Relevant Matches for XML Keyword Search," in VLDB, 2008.


### Authors

M.RAMMOHAN RAO completed M.Tech in Computer Science & Engineering from ANU.Heis presently working as an Asst. Prof. SRI VATSAVAI KRISHNAM RAJU COLLEGE OF ENGINEERING AND TECHNOLOGY, GOLLALAKODERU,PAALAKODERU MANDAL BHIMAVARAM , India. He is having about 3 years of teaching experience and an associate member of CSI and Life member of ISTE, Member in IAENG, published 2 international journals in the area of Data Mining. E-Mail id:rammohanraomt@gmail.com

M.Brahmaiah completed Mtech from ANU.He is presently working as an Asst. Prof. in RVR&JC College of Engineering,chowdavaram, Guntur, India. He is having about 2 years of teaching experience and also worked almost 14 years as programmer in addition to associate member of CSI and Life member of ISTE, Member in IAENG. E-Mail id: Brahmaiah_m@yahoo.com

E.Ramesh completed M.Tech in Computer Science & Engineering from ANU.Heis presently working as an Asst. Prof. in RVR&JC College of Engineering,chowdavaram, Guntur, India. He is having about 7years of teaching experience and an associate member of CSI and Life member of ISTE, Member in IAENG, published 2 international journals in the area of Data Mining. Email id:eluri.r@gmail.com

V.Rajesh completed M.Tech in Computer Science & Engineering from ANU.. He is having about 1 year of teaching experience and an associate member of CSI and Life member of ISTE, Member in IAENG., published 1 international journals in the area of Data Mining.
Email id: Vemula1050@gmail.com