# Survey on Information Retrieval in Semi Structured Data

Dayananda P
Assistant professor
Department of Information Science and Engg,
MSRIT, Bangalore-54

Dr. Rajashree Shettar
Associate professor
Department of Computer Science and Engg,
RVCE, Bangalore-59

## ABSTRACT
The rapid progress of network and storage technologies has led to a huge amount of electronic data such as webpages and XML data has been available on intra and internet. These electronic data are heterogeneous collection of ill-structured data that have no rigid structure, and are often called semi-structure data. These semi-structured data are stored in large repositories (XML databases) and stored as a graph internally in database with tuple as nodes and relationships as edges. As there is ever-growing availability of semi-structured information on web and digital libraries, there is a need of effective keyword search in order to fetch the correct and proximal result on Semi-Structured Data. This paper conducts a survey on how key word search can be performed on semi structure data, techniques involved in performing it, various result ranking strategies and result analysis techniques. It includes the analysis of various indexing schemes and different approaches for increase performance using caches for XML data in order to answer queries.

**Keywords** - XML, ranking, Indexing, keyword search

## 1. INTRODUCTION

The Web provides access to a large number of information sources of all kinds. The major models for semi-structured data exchange over distributed information sources are the Object Exchange Model, the Extensible Markup Language and the Resource Description Framework (RDF). Structured query language like XPath and XQuery is used to search XML data in

XML repository, the relevant keyword search on semi structured data is challenging task because each result fetched can have a multiple matches. The various techniques to retrieve exact match for query results [1, 2, 3, 4, 5, 6] are discussed in detail. There are

Various ranking schemas used for xml keyword search, in order to avoid confusion in search result fetched for xml query and result snippets are used to help for judging correct and relevant result. The speeding up of query processing is achieved using languages such asXQuery or XPath, to achieve improvements on efficiency in xml keyword search is done by using indexes and materialized views. The query processing speed can be increased by using cache, it stores the results of previously answered queries in order to answer succeeding queries faster by reusing these results. The different approaches for using caches are, approach checks whether or not a current query Q can be directly answered from the result of a previously answered query Qi stored in the cache. The new query is otherwise submitted to the source (xml database) [7]. The paper focus at data-centric XML with rich structured information and discuss some basic knowledge about data models, query model and query result.

## 2. BACKGROUND

This section provides an overview on XML data models, query models and the definitions of query results. The XML documents represent hierarchically structured information and are generally modeled as Ordered Labeled Trees. XML document can be modeled as a tree, which is labeled and directed. Each element, attribute and text value in the XML document is a node in the XML tree. Nodes represent XML elements and are labeled with corresponding element tag names, organized following their order of appearance in the document. Each edge in the XML tree represents the membership of the element corresponding to the child node, under the element corresponding to the parent node in the XML document. Graph model models an XML document as a graph. The representation of XML document is shown in figure 1.

**Figure 1 Representation of an XML document**

```
<bookstore>
<book category="COOKING">
<title lang="en">Everyday Italian</title>
<author>Giada De Laurentiis</author>
<year>2005</year>
<price>30.00</price>
</book>
<book category="CHILDREN">
<title lang="en">Harry Potter</title>
<author>J K. Rowling</author>
<year>2005</year>
<price>29.99</price>
</book>
<book category="WEB">
<title lang="en">Learning XML</title>
<author>Erik T. Ray</author>
<year>2003</year>
```

The query models, XSEarch[8] is structured query format, each query term has the format of l: k where l and k are keywords. There is another way to help non-expert users for querying XML documents is graphical query environment. Visual query processing DVQ[9] for XML database systems is used to displays the structure of the XML data to the user and XQBE [10] graphical environment to query XML data on web repository. A query result [11] of a keyword search on an XML document or search on XML repository retrieves a sub tree or subgraph as a result for keyword search, which contains both relevant keyword matches i.e., the XML nodes that match the keywords and other nodes that are relevant by the search engine and XSeek is the first system that automatically infers relevant non-matches i.e XSeek outputs the subtree rooted at each return node as relevant non-matches.

The information retrieval models [12] are divided into set theoretic models, algebraic models and probabilistic models. The Boolean model, case-based reasoning model, fuzzy set model and extended Boolean model are four main types in Set theoretic models. An algebraic model contains vector space model, generalized vector space model, latent semantic indexing model and neural network model. A probabilistic model includes probabilistic model, inference network model and brief network model. The issues are lack of semantics mainly in the Boolean model and the issues in algebraic models are maintaining difficulty, computational cost and lack of validation. The primary issues in probabilistic models are difficult to implement and computing cost.

# 3. APPROACHES FOR KEYWORD SEARCH

In a keyword search, for a given keyword, there may be a many matches found in the data. Any of the matches can be or cannot be necessarily relevant to the query as expected. To solve this problem, many approaches are proposed to identify relevant keyword matches.The three effective approaches are discussed in this paper. LCA (Lowest Common Ancestor) [24] based approaches such as XSEarch[8], MLCA [24] and SLCA [3].
In XML trees, nodes with low LCA are the nodes which are meaningfully related than nodes with higher LCA. Lower nodes in the XML trees will be having more specific semantic meaning. LCA based approaches have proposed different strategies to identify the relevant matches for a search. This approach connects keyword matches and identifies relevant matches using variants of LCA. There are two types of methods in identifying relevant matches. Multi match semantics, where multiple matches to keyword produced in the result. Single match semantics, each result contains only one match keyword.

## 3.1 XSearch

XSEarchis a semantic search engine for XML. It has simple query language, suitable for a naive user. As a result of search, it returns semantically related document fragments. Extended information retrieval techniques are used to rank query answers. For efficient implementation, advanced indexing techniques are developed. It includes full-text search features and ranking to XQuery.

For example, if a paper element has title as its first child and author as its last child, with all the section elements in between, then that paper element will get a low rank, even if query has a keyword from the title and a keyword from author's name. In XSEarch[8], proximity is included in terms of the size of relationship tree in the ranking formula and it ia not affected by the order of children. XSEarchemploys more information-retrieval techniques compared to XRANK. The element ranking used in XEANK can be incorporated in XSEarchas well, but its utility is not clear.

## 3.2 MLCAs

MLCA [24] concept of MLCA was proposed with schema-free XQuery which allow users to mix keyword search and structured query as it is beneficial to find relevant matches. MLCAs evaluated as a composition of standard access methods which are available in most XQuery engines. Li et al. [24] uses a stack based algorithm to compute MLCA nodes. First it retrieves list of all the matches to each keyword. Then it visits all the keyword matches in the document order and maintains a stack in which each node is a descendent of the node below it. I f node contains all the keywords in its sub tree, it is identified as a potential MLCA. To determine the pattern matches, it examines meaningfully relatedness of keyword matches.
In a Schema-Free XQuery with an embedded function MLCA's (e1, e2,. . . , em), where ei are the elements involved in the MLCA'S, we use IList[i] = {a11, a12, . . . , a1ni } belongs to N to represent a list of nodes matching ei (1 • i • m) in the XML data.

### *3.2.1 Basic Implementation of MLCA:*

MLCA [24] computation can be easily implemented using existing query standard operators. The basic idea is to find all the ancestors for each node in the IList, and join nodes sharing common ancestors into trees such that the leaf level contains exactly one node from each IList, and each leaf node has descendant-or-self relationship with the root. For any pair of trees, need to eliminate the one whose root is an ancestor (in database tree) of the root of the other. The remained trees are returned as MLCAs.

## 3.3 SLCA

In SLCA [3], there are two efficient algorithms proposed for keyword search in XML documents according to the SLCA semantics are Indexed Lookup Eager and Scan Eager. These algorithms work quickly and produce parts of answers so that, user may not wait for long to see first few answers.The Indexed Lookup Eager algorithm outperforms the known algorithms and Scan Eager by orders of magnitude when the keyword search includes at least one low frequency keyword along with high frequency keywords.

The performance of the algorithm mainly depends on the number of occurrences of least frequent keyword and the number of keywords in the query. But, it does not depend on the frequencies of more frequent keywords of the query. The Indexed Lookup Eager algorithm is important in practice since the frequencies of keywords typically vary significantly. In contrast, Scan Eager is tuned for the case where the occurrences of the query's keywords do not vary significantly.

# 4. RESULT RANKING

To resolve the ambiguity on fetched results for key word search, various ranking strategies has been proposed for keyword search on XML. The main types of ranking factor for evaluating query results are:

## 4.1Term frequency and inverse document frequency

The term frequency–inverse document frequency is a weight used in information retrieval, it is a statistical measure used to evaluate how important a word is to a document.The importance increases proportionally to the number of times a word appears in the document but is offset by the frequency of the word in the collection. It is used in many search engines for ranking a document's relevance for a user query.

The term frequency of term $t_i$ in document $d_j$ is computed as:

$$\mathbf{tf}_{i,j}=\frac{n_{i,j}}{\sum_k n_{k,j}} \quad (1)$$

where $n_{i,j}$ is the number of occurrences of $t_i$ in document d, and the denominator is the number of occurrences of all terms in document $d_j$.

The inverse document frequency [13] is obtained by dividing the total number of documents by the number of documents containing the term, and then taking the logarithm of that quotient as shown in (2).

$$\mathbf{idf(t)}=\log\frac{|D|}{|\{d:t\in d\}|} \quad (2)$$

$|\mathbf{D}|$ : cardinality of D, or the total number of documents in the collection.

$|\{\mathbf{d}: \mathbf{t} \in \mathbf{d}\}|$ : number of documents, where the term t appears (i.e tf(t,d) ≠0). If the term is not in the collection, this will lead to a division-by-zero. It is common to adjust the formula to $\mathbf{1}+|\{\mathbf{d}: \mathbf{t} \in \mathbf{d}\}|$.

Then $\mathbf{tf - idf(t,d) = tf(t,d) \times idf(t)}$ (3)

A weight in Term frequency (tf) and inverse document frequency (idf) is reached by a high term frequency (in the given document) and a low document frequency of the term in the whole collection of documents; the weights hence tend to filter out common terms.

## 4.2 Vector space model ranking factors

Vector space model[11] is an algebraic model for representing text documents. Word in document is the important component of vector. The document vector and the query vector are used to measure the relevance of a document with respect to a query. XSEarch[6] is implemented same as vector model, it represents each node in the result and each term in the query as a vector, and use vector similarity to determine the relevance of the result. The vector space model has the advantages over the Standard Boolean model are Simple model based on linear algebra, Term weights not binary, allows computing a continuous degree of similarity between queries and documents, allows ranking documents according to their possible relevance and allows partial matching. The vector space model is having some limitation like Search keywords must precisely match document terms, word substrings might result in a false positive match.

## 4.3 PageRank style ranking factors

PageRank is a link analysis algorithm used by Web search engines to analyze the importance of a web page. It assigns a numerical weighing to each link of documents in the web repository for the purpose of measuring its relative importance. XRank [15] adopts a similar idea in computing the importance of a node in an XML search result. It is first system that takes into account (a) the hierarchical and hyperlinked structure of XML documents, and (b) a two-dimensional notion of keyword proximity, when computing the ranking for XML keyword search queries.In XRank the importance of a node is named ElemRank, ElemRank of a node n is computed iteratively based on the number of edges of n and the ElemRank of the nodes that connect to n. the PageRank propagates in a single direction, i.e., from a web page to another web page it links, but not vice versa. The parent (e.g. cooking) and children (e.g. web) are considered as equal importance. ElemRank flows in both forward and backward edges may be weighted differently. In XML, parent-child edges and referential edges are two types of edges. The different probabilities will be assigned to both types of edges. To calculate result score, XRank combines the ElemRank and proximity of keyword nodes in the result, which will be discussed in Section 4.4.

## 4.4 Proximity ranking factors

The term proximity denotes the degree to which a query and a document match, based on the distance between the query terms within a document. The proximity value after calculation contributes to the overall ranking value of a document within a result set. In XRank[6], rank is computed for the root node of each result. The rank of a result root v with respect to a keyword ki, denoted by r(v, ki)

$$\mathbf{r(v,ki)=ElemRank(v_t)\times decay^{(t-1)}} \quad (4)$$

Where decay measures the proximity of keyword matches and ElemRank is discussed in previous Section 4.3. $v_t$ is a descendant of v whose depth is t-1 bigger than that of v; $v_t$directly contains keyword$k_i$v is from $v_t$ i.e a keyword. If there are many matches to a keyword search, in XBridge[14] presents distinct result to the uses and returns the most relevant from a group of SLCA nodes. In groups of SLCA, SLCA with the highest score is considered as the preferred result type of the query.

# 5. RESULT ANALYSIS

The result analysis techniques include result snippets, result clustering, There technique helps the users to search and analyze relevant results. A survey on these topics is discussed below:

## 5.1 Result snippets

In order to resolve the ambiguity of keyword search, various ranking strategies have beenproposed and studied for keyword search on unstructured, semi-structured and structureddata. It is impossible to design a ranking scheme that perfectly fetches the

search result same as users intentions. In order to manage the inaccuracy of ranking functions, web search engines uses snippets for helping user to find relevant answers for a query.The result snippets in XMLkeyword search has been discussed in eXtract [15, 16, and 17], Snippets are built based on four properties: self-contained, distinguishable, representative and small. Information for snippet is built for each result, which contains the informationof the result to be shown in its snippets. The snippet information should be chosen that, maximumnumbers of items areincluded in the snippet subjected to a limit of snippet size.Once the snippet information list of a result is constructed, the items in the list areadded to the snippet in the order they appear in the list. A snippet should distinguish a one result from other results and helps in finding the results quickly, result snippet includes the document title, which helps users for differentiating different results.

## 5.2 Result clustering

Clustering is used as approach to improve the effectiveness of information retrieval. The documents are clustered either before or after retrieval. The related documents in the repository leads to be relevant to the same search, it retrieves more relevant documents by clustering. Lot of ambiguity involved in keyword search,multiple clusters are formed to separates keyword queries or search. The clusters are grouped together based on the same equivalence class in results. Fine grain clustering is explained in Liu and Chen [18], clustering in formed using keyword category based on ancestor nodes of matched keyword. In paper, Liu et al. [19] has experimented that efficiency is increased by using snippets in result clustering, than using query results.

# 6. INDEXING IN INFORMATION RETRIEVAL

To optimize speed and performance in storage and to find the relevant documents for a search, the concept of indexing is introduced. Without an index, the search engine would searches every document in the collection, which would require considerable time and computing power. For example, while an index of 5000 documents can be queried within milliseconds, a sequential search of every word in 5000 large documents could take hours. The XML keyword search in indexes includes inverted indexes and B-tree indexes. The entry in an inverted index consists of a keyword and IDs of the nodes that match thekeyword. Entry in B-tree indexes contain information of each node, clustered by node IDs,Accessing to the children, parent and siblings is done using node IDs.

The basic Dewey labeling scheme[11] is applied a new node is inserted betweentwo sibling nodes, when any operation is performed on XML nodes Dewey labels needs to be updated. Yu et al. [20] extend the different labeling schemes and proposes a prefix-basedPBiTree (Perfect Binary Tree) labeling scheme.The prefix-based PBiTree provide a labeling scheme between twonodes. When a new XML node is inserted, there is no need to update the labels of other XML nodes, if preserved labels are available else re-labeling will be needed.O'Neil et al. [23], Wu et al. [21] and Xu et al. [22] allguarantee that re-labeling is not needed, any number of nodes are inserted.

# 7. RESEARCH CHALLENGES

There are many important problems that callfor further investigation.

## 7.1 Top-k result computation

Applying search on semi structured data results in large number of results, there is a need for calculating top-K results for increasing efficiency on key search. Calculation of top-K result has been discussed for minimal tree semantics [27] and ELCA semantic [28]. The ranking function for weighted tree size is explained in minimal tree semantics and ranking using monotonic aggregation of individual node scores is explained in ELCA semantics. The top result calculation for other result semantics and ranking function is open challenge.

## 7.2 Evaluation of XML Keyword Search

The results for XML key word search can be generated and presented in many number of ways. There are lots of approaches for generating and presenting results for XML keyword search, quality of an XML keyword search is estimated with respect to the ground truth over a large set of test data and queries. There is a bench mark to evaluate key word on XML data i.e INEX[26]. There is a need for comprehensive framework for evaluating XML key word search.

## 7.3 Diverse Data Models

The existing work deals with searching relational databases and data-centric XML data. The keyword search can apply to other types of data, including data extracted from parse tree databases, data warehouses [31], spatial and multimedia databases [32] and probabilistic databases. The techniques that enable users for accessing collections of heterogeneous data sources are in great demand.

## 7.4 Implementing of materialized views

The materialized views are used to improve the efficiency of critical search engine. The XML keyword searches using materialized views using SLCA semantics is disused by Liu and Chen [29]. The challenge is to use the materialized views to improve the query processing time and to apply the materialized view for computation of whole result under SLCA semantic.

## 7.5 Using relevance feedbacks in information retrieval

Exploration for deducing user's interest the user relevance feedback is widely used for achieving success in search quality. Exploring the structural relationships among keywords using explicit feedbacks for XML search is discussed in the paper structured feedback for key word based xml retrieval [30]. Using of relevance feedback, implicit feedback in different aspects of XML keyword search is the biggest challenge. There is a lot of opportunity for utilizing user feedback to improve search quality with reference to the structure of XML data. The implicit feedback can be used in result generation, ranking and result analysis.

# 8. CONCLUSION

A Survey on information retrieval in semi structured data done through several aspects, which includes approaches for keyword search, result ranking, result analysis and indexing in XML keyword search. There are several problems that can be listed for further investigation.Future directions for research include ontology-driven term expansion and further optimization of

queryprocessing and also to investigate techniques for applying MLCAS toqueries involving attributes and references. Finally, exploring moresophisticated IR techniques whereappropriate in schema-free queries is needed much. In ranking function, top-k result computation methods for semantics and ranking functions are still unknown and await discovery. To find the structured relationship among keywords, users feedbacks is used in existing work. Making use of relevance feedback for other aspects is the biggest challenges.

## 9. REFERENCES

[1]. Liu, Z., Chen, Y.: Reasoning and identifying relevant matches for XML keyword search. PVLDB 1(1), 921–932 (2008).

[2]. Li, G., Feng, J., Wang, J., Zhou, L.: Effective keyword search for valuable LCAs over XML documents. In: CIKM, pp. 31–40 (2007).

[3]. Xu, Y., Papakonstantinou, Y.: Efficient keyword search for smallest LCAs in XML databases.In: SIGMOD Conference, pp. 537–538 (2005).

[4] . Li, Y., Yu, C., Jagadish, H.V.: Schema-free XQuery. In: VLDB, pp. 72–83 (2004).

[5]. Guo, L., Shao, F., Botev, C., Shanmugasundaram, J.: XRANK: ranked keyword search over XML documents. In: SIGMOD Conference, pp. 16–27 (2003).

[6]. Cohen, S., Mamou, J., Kanza, Y., Sagiv, Y.: XSEarch: a semantic search engine for XML.In: VLDB, pp. 45–56 (2003).

[7]. Sven Groppe, Jinghua Groppe, and Dirk Müller, "Result Merging Technique for Answering XPath Query over XSLT Transformed Data",In: IEEE TRANSACTION(2009).

[8]. Cohen, S., Mamou, J., Kanza, Y., Sagiv, Y.: XSEarch: a semantic search engine for XML.In: VLDB, pp. 45–56 (2003).

[9]. Zheng, S., Zhou, A., Zhang, L., Lu, H.: DVQ: towards visual query processing of XML database systems. World Wide Web 6(2), 233–253 (2003).

[10]. Braga, D., Campi, A.: XQBE: a graphical environment to query XML data. World Wide Web 8(3), 287–316 (2005).

[11]. Ziyang Liu · Yi Chen , "Processing keyword search on XML: a survey", Springer Science+Business Media, LLC 2011.

[12].HaiDong,Farookh Khadeer Hussain, Elizabeth Chang," A Survey in Traditional Information Retrieval Models", IEEE 2008.

[13]. Salton, G., McGill, M.J.: Introduction to Modern Information Retrieval. McGraw-Hill (1983).

[14]. Li, J., Liu, C., Zhou, R.,Wang, W.: Suggestion of promising result types forXMLkeyword search. In: EDBT, pp. 561–572 (2010).

[15]. Huang, Y., Liu, Z., Chen, Y.: eXtract: a snippet generation system for XML search. PVLDB 1(2),1392–1395 (2008).

[16]. Huang, Y., Liu, Z., Chen, Y.: Query biased snippet generation in XML search. In: SIGMODConference, pp. 315–326 (2008).

[17]. Liu, Z., Huang, Y., Chen, Y.: Improving XML search by generating and utilizing informativeresult snippets. ACM Trans. Database Syst. 35(3), 19:1–19:45 (2010).

[18]. Liu, Z., Chen, Y.: Return specification inference and result clustering for keyword search onXML. ACM Trans. Database Syst. 35(2), 10:1–10:47 (2010).

[19]. Liu, Z., Huang, Y., Chen, Y.: Improving XML search by generating and utilizing informativeresult snippets. ACM Trans. Database Syst. 35(3), 19:1–19:45 (2010).

[20]. Yu, J.X., Luo, D., Meng, X., Lu, H.: Dynamically updating XML data: numbering schemerevisited. World WideWeb 8(1), 5–26 (2005).

[21] Wu, X., Lee, M.-L., Hsu, W.: A prime number labeling scheme for dynamic ordered XML trees.In: ICDE, pp. 66–78 (2004).

[22]. Xu, L., Ling, T.W., Wu, H., Bao, Z.: DDE: from Dewey to a fully dynamic XML labeling scheme.In: SIGMOD Conference, pp. 719–730 (2009).

[23]. O'Neil, P.E., O'Neil, E.J., Pal, S., Cseri, I., Schaller, G., Westbury, N.: ORDPATHs: insertfriendlyXML node labels. In: SIGMOD Conference, pp. 903–908 (2004).

[24]. Schmidt, A., Kersten, M.L., Windhouwer, M.: Querying XML documents made easy: nearestconcept queries. In: ICDE, pp. 321–329 (2001).

[25]. Li, Y., Yu, C., Jagadish, H.V.: Schema-free XQuery. In: VLDB, pp. 72–83 (2004).

[26]. INEX. Initiative for the evaluation of xml retrieval.

http://inex.is.informatik.uni-duisburg.de.

[27]. Golenberg, K., Kimelfeld, B., Sagiv, Y.: Supporting top-K keyword search in XML databasessearch in complex data graphs.In: SIGMOD Conference, pp. 927–940 (2008).

[28]. Chen, L.J., Papakonstantinou, Y.: Supporting top-K keyword search in XML databases. In:ICDE, pp. 689–700 (2010).

[29]. Liu, Z., Chen, Y.: Answering keyword queries on XML using materialized views. In: ICDE,pp. 1501–1503 (2008).

[30]. Schenkel, R., Theobald, M.: Structural feedback for keyword-based XML retrieval. In: ECIR,pp. 326–337 (2006).

[31]. S. Tata and G. M. Lohman. SQAK: doing more withkeywords. In SIGMOD, 2008.

[32]. I. De Felipe, V. Hristidis, and N. Rishe. Keywordsearch on spatial databases. In ICDE, 2008.