Towards developing an Efficient Approach to Keyword Search for XML Documents

S. Selvaganesan[†], Su-Cheng Haw and Lay-Ki Soon

Faculty of Computing and Informatics, Multimedia University, Cyberjaya, Malaysia

[†]sselvaganesan@gmail.com

Abstract. Research on keyword search in XML database is on the increase, owing to its convenient and extensive use in information retrieval (IR) from XML data. Bao et al.[11] proposed an IR-style approach utilizing the statistics of underlying XML data, to resolve keyword ambiguity problems. We find that this approach suffers from the problems of not dealing with tag and data node separately and inefficient utilization of tag and data node frequency information. Based on our findings, we propose a new approach for keyword search for XML document, based on two-level indexing, to resolve these problems. The two-level indexing builds two indices viz. tag information table and data node information table, for structural nodes and data nodes respectively, and deals with each tag and data node separately in XML document. To efficiently use the frequency information, we propose a new formula based on mutual information between selected tags with respect to XML query keywords, and thereby reduce the uncertainty in finding an exact *T*-typed node. Also, we propose an entropy formula to find the exact data value through the selected *T*-typed node.

Keywords: XML, keyword Search, XML document, mutual information, entropy

1. Introduction

In recent years, XML is being used as a de facto standard for information representation and exchange on the Web. As a result, a huge amount of information is stored and represented in XML, and research on keyword search in XML documents is on the increase because it allows users to find information they are interested in without having to know the underlying database schema or complex query language. A lot of research has been conducted in XML keyword search[3,4,6,7] and in these approaches, the returned answers may be meaningful but they may be irrelevant to user search intention. The main problem of effective XML keyword search is to identify the user search intention accurately in the presence of keyword ambiguities.[11]

Bao et al.[11] introduced an IR-style approach utilizing the statistics of XML database to address the problem of XML keyword search viz. search intention identification, result retrieval and relevance oriented ranking. This approach builds two indices viz. keyword inverted list and frequency table. Of these indices, the keyword inverted list retrieves a list of data nodes in document order whose values contain the input keyword. Each inverted list has an index (eg. B+-tree) on the top. For each keyword in the query, the inverted list returns a set of nodes *a* in the document order. Each inverted list containing the input keyword is in the form of a tuple <*DeweyID, prefixPath*, $f_{a,k}$, W_a>, where $f_{a,k}$ is the number of occurrences of a keyword *k* in data node *a* and W_a represents the weight of *a*. In the second index built viz. frequency table, only the frequency f_k^T i.e. number of *T*-typed nodes that contain keyword *k* in their subtrees in XML data, is stored for each combination of keyword *k* and node type *T* in XML document. For each keyword in the given query, the approach gets the value of f_k^T without specifying whether the keyword is a tag or a data value. Moreover, the approach does not deal with each tag and data node separately and thereby makes query processing more complex. XML keyword search based on this will be time consuming. In fact, frequencies $f_{a,k}$ and f_k^T stored in both indices built (inverted list and frequency table) in the approach are not dependent and do not share the information. There will be uncertainty of information between the two frequencies. To resolve the problem of not dealing with tag and data node separately, we propose a new two-level indexing that builds two indices viz. *tag information table* and *data node information table*, for structural nodes and data nodes respectively in XML document. The proposed two-level indexing deals with each tag and data node separately in XML document and thereby speeds up the query processing. To efficiently use the frequency information, we define a new formula for mutual information between selected tags with respect to XML query keywords, and thereby reduce the uncertainty in finding an exact *T*-typed node. By utilizing the concept of entropy, we propose a formula to compute similarity between leaf node of XML document and query keyword so as to find the exact data value through the selected *T*-typed node.

This paper is organized as follows: We briefly present the related work in Section 2. In Section 3, we explain our new approach. Details of implementation consideration and future works are described in Section 4. Finally, in Section 5, we conclude the paper.

2. Related Work

Among the various techniques in recent times, several research works focus on keyword search to effectively retrieve information from the XML documents. In this section, we give concise review of related research works for keyword search on XML data. Bao et al.[8] propose an IR-style approach utilizing the statistics of underlying XML data to address the challenges viz. identification of user search intention, resolving keyword ambiguity problems and estimation of result relevance to a given query. Based on three guidelines proposed in the approach, they design formulae to identify the search for nodes and search via nodes of a query, and present a XML TF*IDF ranking strategy to rank the individual matches of all possible search intentions. The proposed techniques are implemented in an XML keyword search engine called XReal, and extensive experiments show the effectiveness of our approach. Bao et al.[10] model XML document as the interconnected object-trees, based on which they propose two main matching semantics called Interested Single Object (ISO) and Interested Related Object (IRO), to capture different user search concerns. A customized ranking scheme is proposed by taking both the structure and content of the results into account. They propose efficient algorithms to compute and rank the query results in one phase. Bao et al.[11] make several updates to Ref.8 as an extension. To complement the result ranking framework, they take the popularity into consideration for the results that have comparable relevance scores. New index and efficient algorithm are designed to compute the popularity score and more experiments are conducted to show the effectiveness of the approach. Inspired by Ref.11, we propose a new approach for efficient XML keyword search.

3. Proposed Approach

This section presents our approach for keyword search for XML documents describing two-level indexing, selection of *T*-typed nodes using two-level matching, design of mutual information measure to find the exact *T*-typed node and design of entropy measure to find the exact data value through the selected *T*-typed node. Also the approach will include the design of popularity of query results that have comparable relevance scores and evaluation to prove the efficiency of the approach in keyword search.

3.1 Two-level indexing

With a single frequency table,[11] ambiguity exists in whether a query keyword is a tag or a data, making the query process more complex. This could be overcome using the proposed two-level indexing. After pre-processing the XML document, the two-level indexing builds two indices viz. tag information table and data node information table, for structural nodes and data nodes respectively. Unlike the indices of Ref.11, for each tag in XML document, the proposed approach stores tag name, frequency of occurrences of tag in *T*-typed nodes and their subtrees, and prefix path of the corresponding *T*-typed node, in the tag information table. Similarly, for each data node, it stores data value, names of leaf tag of the data node and frequency of occurrences of data node contained in the corresponding leaf tag in XML document, in the data information table. The data node information table is dependent on the tag information table in relation with the tag name. Hence, the proposed two level indexing deals with each tag and data node separately in XML document and thereby speeds up the query processing.

3.2 Selection of T-typed nodes

By default, there exists a unique desired node type to search for, in the search intention of each query. Using the two-level matching between the two indices, all the possible *T*-typed nodes will be selected for a query. When searching for an input query, each query keyword is initially searched in the tag information table. If the query keyword matches the tag in the tag information table, it is considered as a tag and, subsequently, tag name, frequency of occurrence of the tag in *T*-typed nodes and their subtrees and prefix path of *T*-typed nodes containing the tag will be retrieved from the tag information table. When the query keyword is not a tag, it is searched in the data node information table until a match is obtained. The query keyword is considered as a data value and the name of leaf tag of the corresponding data nodes from the data node information table will then be searched in the tag information table and the corresponding leaf tag name, frequency of occurrence of the tag in the subtrees of *T*-typed nodes, and prefix path of *T*-typed nodes will be retrieved from the tag information table.

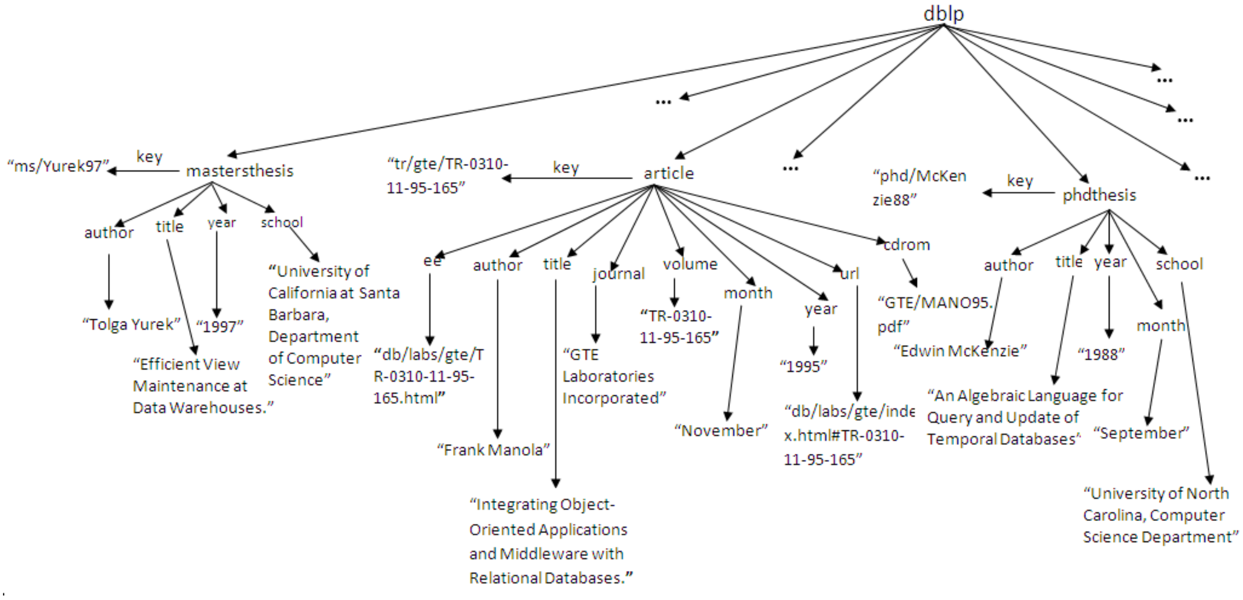
dblp
...
...
...
...
...
"ms/Yurek97"
key
mastersthesis
author
title
year
school
"Tolga Yurek"
"1997"
"Efficient View Maintenance at Data Warehouses."
"University of California at Santa Barbara, Department of Computer Science"
"tr/gte/TR-0310-11-95-165"
key
article
ee
author
title
journal
volume
month
year
url
cdrom
"db/labs/gte/TR-0310-11-95-165.html"
"Frank Manola"
"Integrating Object-Oriented Applications and Middleware with Relational Databases."
"GTE Laboratories Incorporated"
"TR-0310-11-95-165"
"November"
"1995"
"db/labs/gte/index.html#TR-0310-11-95-165"
"GTE/MANO95.pdf"
"phd/McKenzie88"
key
phdthesis
author
title
year
school
month
"Edwin McKenzie"
"1988"
"An Algebraic Language for Query and Update of Temporal Databases"
"September"
"University of North Carolina, Computer Science Department"

Figure 1. Portion of data tree for dblp XML document

A query keyword may appear in different *T*-typed nodes and their subtrees. Consider a XML keyword query "*month*, *November*" issued on the dblp XML document in Figure 1. The proposed approach finds the match for the first keyword *month* in the tag information table and returns in the format *<tag name#frequency of occurrence of the tag in T-typed nodes and their subtrees#prefix path of T-typed nodes containing the tag>* i.e. *month#22#dblp,article, month#3#dblp,phdthesis*. The second keyword *November* matches with data value in the information table and the name of leaf tag containing data value *November* is obviously *month*. Based on the leaf tag *month* from the data node information table, the proposed approach returns *month#22#dblp,article, month#3#dblp,phdthesis*. Hence, the query keywords occur in node types article and phdthesis and the corresponding prefix paths *dblp,article* and *dblp,phdthesis* will be extracted.

3.3 Selection of desired T-typed node

T-typed nodes for a given keyword query and the corresponding prefix paths will be selected based on the two-level indexing described in Section 3.2. The keyword matching tag may occur once or many times in different *T*-typed nodes and their subtrees. In case of a query keyword with more number of matches, query processing will be complex and an arithmetic formula is to be formulated to filter out the optimum *T*-typed node. In this section, we review the basics of mutual information and then apply it for keyword search in XML document.

3.3.1 Mutual Information for every prefix path extracted

The mutual dependence between two random variables can be measured using mutual information.[5] Pointwise mutual information (PMI) is defined over values of random variables.[9] If X and Y are discrete random variables with the joint distribution *f(x,y)* and the marginal distributions *f(x)* and *f (y)*, then

$$I(x,y) = \log\frac{f(x,y)}{f(x)f(y)} \quad (1)$$

is the pointwise mutual information at *(x,y)*. Intuitively, PMI is the amount of information provided by the occurrence of event *y* about the occurrence of event *x*. To make this definition unambiguous, a base of 2 to the log function could be used. Mutual information, intuitively, reduces the uncertainty of one random variable with the information obtained from another variable. Higher the mutual information, larger is the reduction in uncertainty and vice-versa. When the two random variables are independent, mutual information between them is zero.

In our approach, the tag information table index and data node information table index are mutually dependent and share information. Consequently, the selected tags namely query keyword matching tags and leaf tags containing keyword matching data values are dependent. Intuitively, pointwise MI is the amount of information provided by the occurrence of query keyword matching tags about the occurrence of leaf tags containing keyword matching data values. By incorporating the concept of mutual information, we define $I(t,t_d)$, which is the pointwise mutual information (PMI) between selected tags i.e. query keyword matching tags and leaf tags containing keyword matching data values as follows:

$$I(t,t_d) = \log_2\frac{f(t,t_d)}{f(t)f(t_d)} * r^{depth\ (T)} \quad (2)$$

where t represents keyword matching tags, t_d represents leaf tags containing keyword matching data values, $f(t,t_d)$ is the combined frequency of t and t_d, $f(t)$ is the sum of frequencies of keyword matching tags, $f(t_d)$ is the sum of frequencies of leaf tags containing keyword matching data values, r is a reduction factor with range (0,1) and is normally chosen to be 0.8, and *depth(T)* is the depth of *T*-typed nodes in document. The reduction factor $r^{depth(T)}$ in Formula 2 is used to reduce mutual information of the node types that are deeply nested in the XML document.

If the given query keyword matches with tag(s) in the tag information table, the approach will retrieve tag names, frequency of occurrences of the tag in *T*-typed nodes and their subtrees and prefix path of *T*-typed nodes containing the tag from the tag information table. Frequency of each keyword matching tags contained in *T*-typed nodes and their subtrees will be added together to get $f(t)$. Similarly, for each keyword matching data value, the name of leaf tag of the corresponding data value from the data node information table will then be searched in the tag information table, and the corresponding tag name, frequency of occurrence of the tag in *T*-typed nodes and their subtrees, and prefix path of *T*-typed nodes will be retrieved from the tag information table. Frequency of leaf tags of each keyword matching data value will be added together to get $f(t_d)$. Based on the combination of keywords in a given query, for every prefix path extracted, combined frequency $f(t,t_d)$ will be calculated from the tag information table. Mutual information between selected tags will then be computed for every prefix path extracted. In the event of each node type being the desired type, the node with the highest mutual information is chosen as the appropriate *T*-typed node.

For the query "*month*, *November"* as explained in Section 3.2, the query keywords occur in node types article and phdthesis and the corresponding prefix paths *dblp,article* and *dblp,phdthesis* will be extracted. For every prefix path extracted, the approach will find mutual information in order to select exact *T*-typed node. As such, mutual information for the extracted prefix paths *dblp,article* and *dblp,phdthesis* will be -3.090 and -4.930 respectively. Node type article will be obviously the exact *T*-typed node for the given XML keyword query, as mutual information value for prefix path *dblp,article* is higher.

3.4 Similarity measure for every prefix path with respect to its keyword

After finding the exact *T*-typed node for a query, it is necessary to find the exact data value through the selected *T*-typed node. For each data in the given input query, the proposed approach will search for all the matching similar data values in the data node information table, find the relevant leaf tag containing the data value and subsequently compare each relevant tag with every prefix path extracted, so as to return similar paths in the format <*tag name#frequency of occurrence of the tag in T-typed nodes and their subtrees#prefix path of T-typed nodes containing the tag*>. Moreover, our approach proposes an entropy formula to compute the similarity between leaf node and keyword for every prefix path with respect to its keyword.

3.4.1 Entropy Measure

Entropy[1,2,5] $H(X)$ of a random variable X with n outcomes $\{x_i: i = 1,...,n\}$, a measure of uncertainty, is defined as

$$H(X) = -\sum_{i=1}^{n} p(x_i) \ \log_b p(x_i) \tag{3}$$

where p is the probability mass function of x_i. In this definition, the convention 0 log $0 = 0$ is adopted. Based on the definition of entropy, we propose a formula to find the exact data through the selected T-typed node as follows:

$$Entropy = -\sum_{i=1}^{n} \frac{f_{q_i}^{T_a}}{f^{T_a}} \log_2 \frac{f_{q_i}^{T_a}}{f^{T_a}} \tag{4}$$

where q represents a keyword query; n is the number of query keywords; a is data node; T_a is the type of a's parent node, f^{T_a} is the number of T_a-typed nodes and $f_{q_i}^{T_a}$ is the number of T_a-typed nodes containing query keyword. A more complex, and less predictable parameter carries higher entropy, and vice versa. With the entropy of each prefix path extracted, the path with the lowest entropy will be chosen as the appropriate prefix path.

For each data in the given input query *"month, November"*, the proposed approach will return similar data *november*, and find the relevant tag with the largest frequency of those data i.e. *month*. Also it will return *month#22#dblp,article* and *month#3#dblp,phdthesis* with similar paths and for these paths, our designed entropy formula will return 0.203 and zero. Zero entropy indicates non occurrence of data value *november* in the prefix path *dblp,phdthesis*. Obviously, prefix path *dblp,article* will be the exact path.

4. Implementation Consideration and Future Work

The initial results reported in Section 3 confirm that the proposed approach utilizes the frequency information effectively and efficiently. Evidently, we will pursue our research with the full implementation of the proposed approach for effective XML keyword search that has been presented in Section 3. Upon considering the popularity of query results, we will design the popularity of query results that have comparable relevance scores. The results will be extensively evaluated to prove the efficiency of the proposed approach for XML keyword search.

5. Conclusion

We propose a new approach for keyword search in XML document based on two indices viz. tag information table and data node information table in XML document. We develop a searching technique of selecting all possible T-typed nodes for a given query using the two-level matching between two indices. By incorporating the concept of mutual information and dependence of two indices, we define the mutual information between selected tags and query keywords to find the exact T-typed node. By adopting the concept of entropy, we design a formula to compute similarity between the leaf nodes of XML document and the query keywords so as to find the exact data value through the selected T-typed node. To continue with our approach, our future works involve designing the popularity of query results that have comparable relevance scores and evaluating the results to prove the efficiency of the proposed approach in XML keyword search. Our final goal is to analyze the designed approach on various XML databases and evaluate the approach with the different keyword search algorithms.

6. References

[1] Shannon, C.E., "A Mathematical Theory of Communication," Bell Syst. Tech. J. 27, 379-423 and 623-656 (1948).

[2] MacKay, D.J.C., [Information Theory, Inference, and Learning Algorithms], Cambridge University Press, Cambridge, 22-46 (2003).

[3] Guo, L., Shao. F., Botev, C. and Shanmugasundaram, J., “XRANK: Ranked Keyword Search over XML Documents,” Proc. SIGMOD, 16-27 (2003).

[4] Xu, Y. and Papakonstantinou, Y., “Efficient Keyword Search for Smallest LCAs in XML databases,” Proc. SIGMOD, 537-538 (2005).

[5] Cover, T.M. and Thomas, J.A., [Elements of Information Theory], John Wiley and Sons, Hoboken, New Jersey, 13-23 (2006).

[6] Liu, Z. and Chen, Y., “Identifying Meaningful Return Information for XML Keyword Search,” Proc. SIGMOD, 329-340 (2007).

[7] Liu, Z. and Chen, Y., “Reasoning and Identifying Relevant Matches for XML Keyword Search,” Proc. VLDB Endowment 1(1), 921-932 (2008).

[8] Bao, Z., Ling, T.W., Chen, B. and Lu, J., “Effective XML Keyword Search with Relevance Oriented Ranking,” Proc. IEEE Int. Conf. Data Eng., 517-528 (2009).

[9] Bouma, G., “Normalized (Pointwise) Mutual Information in Collocation Extraction,” Proc. The Biennial GSCL Conf., 31-40 (2009).

[10] Bao, Z., Lu, J., Ling, T.W., Xu, L. and Wu, H., “An Effective Object-level XML Keyword Search,” Proc. DASFAA 1, 93-109 (2010).

[11] Bao, Z., Lu, J., Ling, T.W. and Chen, B., “Towards an Effective XML Keyword Search,” IEEE Trans. Knowl. Data Eng. 22(8), 1077-1092 (2010).