SAILER: An Effective Search Engine for Unified Retrieval of Heterogeneous XML and Web Documents

Guoliang Li, Jianhua Feng, Jianyong Wang, Xiaoming Song, and Lizhu Zhou
Department of Computer Science and Technology, Tsinghua University, Beijing 100084, P. R. China
{liguoliang,fengjh,jianyong,dcszlz}@tsinghua.edu.cn;songxm07@mails.tsinghua.edu.cn

ABSTRACT

This paper studies the problem of unified ranked retrieval of heterogeneous XML documents and Web data. We propose an effective search engine called SAILER to adaptively and versatilely answer keyword queries over the heterogenous data. We model the Web pages and XML documents as graphs. We propose the concept of *pivotal trees* to effectively answer keyword queries and present an effective method to identify the *top-k* pivotal trees with the highest ranks from the graphs. Moreover, we propose effective indexes to facilitate the effective unified ranked retrieval. We have conducted an extensive experimental study using real datasets, and the experimental results show that SAILER achieves both high search efficiency and accuracy, and outperforms the existing approaches significantly.

Categories and Subject Descriptors

H.2.8 [**Database Applications**]: Miscellaneous

General Terms

Algorithms, Performance, Languages

Keywords

Keyword Search, XML, Web Pages, Unified Keyword Search

1. INTRODUCTION

Existing web search engines cannot integrate the information from multiple interrelated pages to answer keyword queries meaningfully. The next-generation Web search engines require *link-awareness*, or more generally, the capability of integrating the correlative information that are linked through hyperlinks. For example, to search for the conferences including the topic of "`Information Retrieval`" and held in "`Beijing` 2008", users issue a keyword query of `"Conference 2008 Beijing Information Retrieval"` to a search engine like `GOOGLE`. As we all know, `"WWW 2008"` is held in Beijing and "`Information Retrieval`" is one of its major research topics, but surprisingly, the homepage of `"WWW 2008"` is not in the top-10 results and not even in the first one hundred answers either. This is because `"WWW 2008"` splits its information into several pages methodically. The page of IMPORTANT DATE contains keywords `"2008,Conference"`, `"Information Retrieval"` is contained in the page of CALL-FOR-PAPER and `"Beijing"` is included in the homepage. Consequently, existing search engines often include a number of false negatives due to the limitation of their models which take only a list of individual pages as the result but neglect the fact that interrelated pages linked by hyperlinks may be more meaningful. However, this is not an *ad hoc* problem but ubiquitous over the Internet.

As XML is widely recognized as the data interchange standard over the Internet, the research community has been introducing keyword search capability into XML documents[3, 4, 5]. To the best of our knowledge, few existing works could be universally applied to Web pages and XML documents. Therefore, providing both effective and efficient search ability over such heterogeneous collections within a single search engine remains a big challenge. This calls for a framework for indexing and querying over large collections of heterogeneous data. To address these problems, we propose an effective search engine SAILER based on Structure-Aware Indexing for unified retrievaL of hetERogeneous XML and web documents. As opposed to the traditional search engines, which return a list of individual pages as the results, SAILER extracts a set of relevant pages, which are highly interrelated and related to queries.

2. SAILER

2.1 Graph Modeling

We model the Web pages and XML documents as graphs, where the nodes are respectively pages and elements and links are hyperlinks between pages and parent-child relationships (or IDREF) in XML documents. We can translate the problem of keyword search over the heterogeneous data to the problem of finding the connected trees with minimal cost over the graphs, which contain all or a part of input keywords, called *Steiner trees*. However, it is fairly difficult to extract the Steiner trees in a large graph, which is NP-hard [1]. Alternatively, we devise indices for facilitating keyword-based search over large graphs.

2.2 Pivotal Trees

DEFINITION 1. (PIVOTAL NODE) *Given a graph $\mathcal{G}$, a keyword k_i, and a node $n \in \mathcal{G}$ that directly or indirectly contains k_i, the node $pn_{(k_i,n)}$, which directly contains k_i and has the minimal distance with n, is called a pivotal node. That is,*

$$pn_{(k_i,n)} = argmin_{n_r}\{\delta(n_r,n) | n_r \in \mathcal{G}\},$$

where n_r directly contains k_i and $\delta(n_r,n)$ denotes the distance between n_r and n.

WWW 2008, April 21–25, 2008, Beijing, China.
ACM 978-1-60558-085-2/08/04.

DEFINITION 2. (PIVOTAL TREE) *Given a keyword query* $\mathcal{K}=\{k_1, k_2, \cdots, k_m\}$*, and a graph* $\mathcal{G}$*, consider node* $n \in \mathcal{G}$*, the subtree rooted at* n *and containing the pivotal paths from* n *to every pivotal node* $pn_{(k_i,n)}$ *is called a Pivotal Tree.*

Pivotal trees are compact connected trees in the graph, which contain all the input keywords, and therefore they can be taken as the answers of keyword queries.

2.3 Ranking

We present how to effectively rank the pivotal trees. Given a pivotal tree $\mathcal{PT}$ and a keyword query $\mathcal{K}=\{k_1, k_2, \cdots, k_m\}$, we present Equation 1 to rank the pivotal tree $\mathcal{PT}$.

$$\text{SCORE}(\mathcal{K}, \mathcal{PT}) = \sum_{k=1}^{m} \text{SCORE}(\texttt{root}(\mathcal{PT}), k_i) \quad (1)$$

where $\texttt{root}(\mathcal{PT})$ denotes the root of $\mathcal{PT}$. $\text{SCORE}(\texttt{root}(\mathcal{PT}, k_i))$ denotes the score of k_i in $\mathcal{PT}$. Given any node n and a keyword k_i, we present how to assign the score of k_i in n (i.e., $\text{SCORE}(n, k_i)$) as follows. If n directly contains k_i, we propose Equation 2 to compute $\text{SCORE}(n, k_i)$.

$$\text{SCORE}(n, k_i) = \frac{ln(1 + tf(k_i, n)) * ln(idf(k_i))}{(1 - s) + s * ntl(n)} \quad (2)$$

where $tf(k_i,n)$ denotes the term frequency of k_i in n; $idf(k_i)$ denotes the inverse document frequency of k_i; $ntl(n)$ denotes the normalized term length of n and $ntl(n)=\frac{|n|}{\frac{\sum_{n' \in \mathcal{G}} |n'|}{|\mathcal{G}|}}$, where $|n|$ denotes the number of terms in n and $|\mathcal{G}|$ denotes the number of nodes in $\mathcal{G}$; s is a constant and usually set to 0.2.

If n indirectly contains k_i, we present Equation 3 to compute $\text{SCORE}(n, k_i)$.

$$\text{SCORE}(n, k_i) = \frac{\text{SCORE}(pn_{(k_i,n)}, k_i)}{\sigma^{\delta(pn_{(k_i,n)}, n)}} \quad (3)$$

where σ is an attenuation factor. Obviously, the larger distance between k_i and n, the less relevant between them. We experimentally prove that σ is usually set to 0.8.

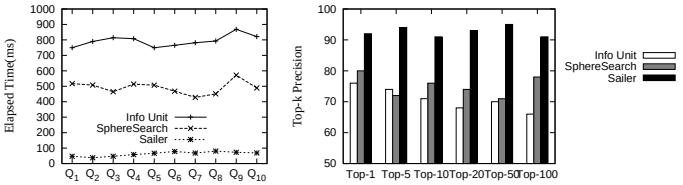
Note that, $\text{SCORE}(pn_{(k_i,n)}, k_i)$ can be computed based on Equation 2, as $pn_{(k_i,n)}$ directly contains k_i and $\delta(pn_{(k_i,n)}, n)$ can be pre-computed off-line. Accordingly, we can score the nodes that indirectly or directly contain the keywords based on Equation 2 and Equation 3.

2.4 Indexing

We note that $\text{SCORE}(n, k_i)$ in Equation 2 and Equation 3 can be pre-computed off-line and thus we can materialize such scores into the index. The entries of the index are the keywords that contained in the graph. Different from inverted indices which only maintain the nodes that directly contain the keyword, each entry of index preserves the nodes that *directly* or *indirectly* contain the keyword in the form of a triple <Node, Score, Pivotal Path>, where the Score is the assigned score of the keyword in the Node, and Pivotal Path preserves the path from Node to the corresponding *pivotal node*. Accordingly, the index captures the rich structural relationships as each entry preserves the paths from a given node to the corresponding pivotal node.

3. EXPERIMENTAL STUDY

We have designed and performed a comprehensive set of experiments to evaluate the performance of our approach. We crawled a huge amount of real data from the Internet.



(a) Search Efficiency (b) Search Quality

Figure 1: Search Efficiency and Quality

There are four types of data in our dataset: *i*) the homepages of top conferences, such as WWW, SIGIR, SIGMOD and so on; *ii*) the hompages of research groups; *iii*) the homepages of researchers; *iv*) XML, PDF, WORD and PPT documents. There were approximate 100,000,000 documents. The experiments were conducted on an Intel(R) Pentium(R) 2.4GHz computer with 1GB of RAM. The algorithms were implemented in Java. We compared SAILER with state-of-the-art methods, Information Unit [6], and SphereSearch [2]. We selected one hundred queries for the experiments. We gave the elapsed time of the first ten queries and the average precision of all the queries as illustrated in Figure 1.

4. CONCLUSION

In this paper, we have investigated the problem of unified retrieval over heterogeneous web pages and XML documents. We modeled the heterogeneous data as graphs and identified the pivotal trees to answer keyword queries. We proposed indexes for facilitating the identification of pivotal trees. We have conducted an extensive performance study to evaluate the search efficiency and quality of our method. The experimental results show that our approach achieves both high search efficiency and quality, and outperforms the existing approaches significantly.

5. ACKNOWLEDGEMENT

This work is partly supported by the National Natural Science Foundation of China under Grant No.60573094, the National High Technology Development 863 Program of China under Grant No.2007AA01Z152 and 2006AA01A101, the National Grand Fundamental Research 973 Program of China under Grant No.2006CB303103.

6. REFERENCES

[1] Gaurav Bhalotia, Arvind Hulgeri, Charuta Nakhe, Soumen Chakrabarti, and S. Sudarshan. Keyword searching and browsing in databases using banks. In *ICDE*, 2002.

[2] Jens Graupmann, Ralf Schenkel, and Gerhard Weikum. The spheresearch engine for unified ranked retrieval of heterogeneous xml and web documents. In *VLDB*, 2005.

[3] Guoliang Li, Jianhua Feng, Jianyong Wang, and Lizhu Zhou. Efficient keyword search for valuable lcas over xml documents. In *CIKM*, 2007.

[4] Guoliang Li, Beng Chin Ooi, Jianhua Feng, Jianyong Wang, and Lizhu Zhou. EASE: Efficient and Adaptive Keyword Search on Unstructured, Semi-structured and Structured Data. In *SIGMOD*, 2008.

[5] Guoliang Li, Jianhua Feng, Jianyong Wang, and Lizhu Zhou. RACE: Finding and Ranking Compact Connected Trees for Keyword Proximity Search over XML Documents. In *WWW*, 2008.

[6] Wen-Syan Li, K. Selcuk Candan, Quoc Vu, and Divyakant Agrawal. Retrieving and organizing web pages by 'information unit'. In *WWW*, 2001.