Generic and Effective Semi-Structured Keyword Search

Arash Termehchy and Marianne Winslett
Department of Computer Science, University of Illinois, Urbana IL USA
{termehch, winslett}@cs.uiuc.edu

ABSTRACT

Current semi-structured keyword search and natural language query processing systems use ad hoc approaches to take advantage of structural information. Although intuitive, they are ultimately ad hoc. We have developed the concept of *coherency ranking* based on the statistical properties of data to rank the keyword search results. We demonstrate how the coherency ranking works for two real-world XML databases and show its advantages over the previously proposed XML keyword search methods.

Categories and Subject Descriptors

H.2.4 [**Database Management**]: Query Processing

General Terms

Algorithms, Design, Language

1. INTRODUCTION

Keyword search is becoming a popular query interface for (semi-)structured databases [2, 3]. Since the keyword query is not described in a database query language, the challenge is to find the data elements in the database that are closely related to the input query. An ideal system must return all related portions of data (high recall) and do not return any non-relevant portion (high precision). More importantly, as relatedness is a matter of degree, the system must rank the candidate answers so that the most relevant answers appear at the beginning of the list. Current proposals use intuitively appealing but ad hoc heuristics to use the structural information in keyword search processing [2, 3, 5]. These approaches assume certain properties in the database schema which are frequently violated in practice, even in the highest-quality XML schemas. Therefore, current approaches suffer from low precision, low recall, or both. They either do not rank the candidate answers or use a naive ranking heuristic (e.g., smallest answer first). In this demo, we present *coherency ranking*, a generic ranking method for XML keyword search. Coherency ranking is based on a theoretical framework that defines the semantic relatedness of query terms to XML subtrees, based on *normalized total correlation* [6], an extended version of the concepts of data dependencies and mutual information. The strong statistical basis of coherency ranking allows us to avoid the pitfalls that lower the precision and recall of previous approaches, and provide a better ranking than the previous proposals.

We have built a system that deploys coherency ranking in XML keyword query processing, using a two-phase approach. The first phase is a preprocessing step that extracts the meaningful substructures from an XML DB, before the query interface is deployed. During normal query processing, we use the results of the preprocessing phase to rank the substructures in the DB that contain the query keywords. We have performed an empirical evaluation with two real-world XML data sets and user-supplied queries that shows that coherency ranking is efficient at run time and has much higher precision, recall, and ranking quality than previous approaches. We have also developed optimization and estimation techniques to minimize preprocessing costs.

The goal of this demo is to illustrate the pitfalls of the previous approaches and explain the importance of taking advantage of deep XML structural information to address these weak points. We will also introduce the concept of coherency ranking and explain how it provides desirable rankings for the candidate answers. More precisely, we will use the user interface for coherency ranking running side-by-side with the leading proposed methods for XML keyword search: XRANK [3], SLCA and MaxMatch [5], XSEarch [2], CVLCA [4], and XReal [1]. Using XML versions of IMDB and DBLP and queries supplied by users not performing this research, we will demonstrate that the precision and recall advantages that we have found for coherency ranking [6] translate into a **significantly more usable system** than competing approaches. Our queries will be drawn from the set of 65 queries in [6], plus audience-supplied queries over the versions of IMDB/DBLP that we bring with us.

In the remainder of this paper, we will outline the underpinnings of coherency ranking. More details can be found in our technical report [6].

2. COHERENCY RANKING

The consensus in XML keyword search is that the answers should include the most specific subtrees containing the query terms. The specificity of a subtree depends on the strength of the relationship between its nodes. For instance, if two nodes merely belong to the same bibliography,

KEYS'09, June 28, 2009, Providence, Rhode Island, USA.

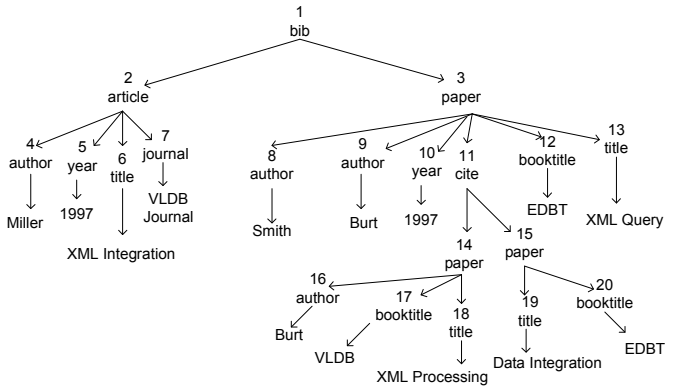

Figure 1: DBLP database fragment, with DBLP's native XML schema

such as titles of two different papers, then the user will not gain any insight from seeing them together in an answer. If the nodes belong to the same paper, such as the paper's title and author, the user will surely benefit from seeing them together. If the nodes represent titles of two different papers cited by one paper, the answer might be slightly helpful.

The *baseline method* for XML keyword search returns *every* candidate answer (with modest refinements in [3]). For instance, consider the DBLP fragment from http://dblp.uni-trier.de/xml shown in Fig. 1. The answer to query *Integration Miller* is (rooted at) node 2. This approach has high recall but very low precision. For example, for query Q_1 = *Integration EDBT*, the baseline approach returns the desired answer of node 15, but also the unhelpful root node. The SLCA approach [5] eliminates every candidate answer whose root is an ancestor of the root of another candidate answer. This approach relies on the intuitively appealing heuristic that far-apart nodes are not as tightly related as nodes that are closer together. For Q_1, the SLCA approach does not return node 1. However, it returns node 11 for Q_2 = *Integration VLDB*; as it does not rank its answers, the user will get a mix of unhelpful and desirable answers. SLCA's recall is less than the baseline's: for query Q_3 = *XML Burt*, nodes 3 and 14 are desirable; but since node 3 is an ancestor of node 14, node 3 will not be returned. XSEarch [2] removes every candidate answer having two non-leaf nodes with the same label. The idea is that non-leaf nodes are instances of the same entity type if they have duplicate labels, and there is no interesting relationship between entities of the same type. XSEarch ranks the remaining answers by the number of nodes they contain and their TF/IDF. This is not an ideal way to detect nodes of similar type. For example, nodes *article* and *paper* in Fig. 1 have different names but represent similar objects. As a result, for the query Q_4 = *Burt Miller*, DL returns the root node, which is undesirable. Other approaches have similar pitfalls [6].

The key shortcoming of all these methods is that they *filter out answers instead of ranking them* and/or they *rely on shallow structural properties to rank answers*. Current approaches implicitly assume that all relationships between nodes in a candidate answer have the same strength. For example, they assume that the relationship between *booktitle* and *author* is as important as the relationship between *title* and *author* in DBLP. However, a paper's author is more closely related to its title than to its proceedings title, in which case users will prefer answers whose *title* and *author* children match the query terms, rather than answers whose *booktitle* and *author* children match. Also, the relationship between the *title* of a paper and the *title* of the papers it cites is stronger than the relationship between the *t*itles of the cited papers themselves. Similarly, the relationship between the *title* of an article and the *title* of a paper is weak if their only connection is that they are in the same database. These observations suggest that the higher the correlation between schema elements, the more meaningful their relationship is. We have formalized this intuition as *coherency ranking*, which uses information theoretic concepts to capture the correlation between the schema elements in a subtree [6]. Coherency ranking ranks a subtree higher if its *normalized total correlation* (NTC) is higher than that of other candidate answers. NTC is normalized according to the number of the schema elements in the subtree and their statistical properties. Our implementation of coherency ranking allows the user to set a minimum value for the NTC of the subtrees it returns. In the demo, we will set the minimum NTC to zero, meaning that subtrees rooted at node 1 will not be returned for the example queries.

We have proposed an algorithm that efficiently computes the NTC for all schema elements in a preprocessing phase [6]. The result of the preprocessing phase is used at query time to rank candidate answers. We use a stack bases algorithm to efficiently find candidate answers. We have explored and implemented the ways that NTC can be combined with TF/IDF heuristic. We have performed a user study with DBLP and IMDB (http://www.imdb.com) [6]. Also, it improves the mean average precision of the previous approaches by up to 11%.

Although our focus in this demo is on XML keyword queries, coherency ranking is also appropriate for relational and graph DBs (e.g., RDF, OWL, XML with ID/IDREF), and for natural language queries.

3. REFERENCES

[1] Z. Bao, T. W. Ling, B. Chen, and J. Lu. Effective XML Keyword Search with Relevance Oriented Ranking. In *ICDE 2009*.

[2] S. Cohen, J. Mamou, Y. Kanza, and Y. Sagiv. XSearch: A Semantic Search Engine for XML. In *VLDB 2003*.

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

[4] G. Li, J. Feng, J. Wang, and L. Zhou. Effective Keyword Search for Valuable LCAs over XML Documents. In *CIKM 2007*.

[5] Z. Liu and Y. Chen. Reasoning and Identifying Relevant Matches for XML Keyword Search. In *VLDB 2008*.

[6] A. Termehchy and M. Winslett. Effective Ranking of XML Keyword Search Results (Extended Version), University of Illinois, UIUCDCS-R-2009-3043, 2009.