IGI PUBLISHING

701 E. Chocolate Avenue, Hershey PA 17033-1240, USA
Tel: 717/533-8845; Fax 717/533-8661; URL-http://www.igi-global.com

ITJ 5157

This paper appears in the publication, Journal of Database Management, Volume 20, Issue 3
edited by **Keng Siau © 2009, IGI Global**

OOXKSearch:
A Search Engine for Answering XML Keyword and Loosely Structured Queries Using OO Techniques

Kamal Taha, The University of Texas at Arlington, USA

Ramez Elmasri, The University of Texas at Arlington, USA

ABSTRACT

OOXKSearch is a semantic search engine that answers XML keyword-based queries as well as loosely structured queries using Object Oriented techniques. There has been extensive research in XML keyword-based and loosely structured querying. Some frameworks work well for certain types of XML data models while fail in others. The reason is that the proposed techniques are based solely on establishing relationships between individual elements while overlooking the context of these elements. The context of a data element is determined by its parent, because it specifies one of the characteristics of the parent. Since data elements are nothing but characteristics of their parents, we observe that we could treat each parent-children set of elements as one unified entity. We then find semantic relationships between the different unified entities. If two distinct unified entities are semantically related, their data elements are also semantically related. The search performance and quality of OOXKSearch were evaluated experimentally and compared with three recent proposed systems. The results showed marked improvement. [Article copies are available for purchase from InfoSci-on-Demand.com]

Keywords: *Canonical Tree; Keyword Search; Keyword Query; Loosely Structured Query; XML; XML Search Engine*

INTRODUCTION

With the emergence of World Wide Web, some businesses' databases are being queried by customers. The customers may not be aware of the underlying data and its structure, and might never learn a query language that enables them to issue a structured query. Some of the employees of these businesses who query the databases may also not be aware of the structure of the data, but they are likely to be aware of some of the elements and their labels. We believe that there is a need for a search engine that answers each user based on his *degree* of knowledge of the underlying data. The engine is a dual search engine, which is composed of a keyword-based search engine (e.g., for answering customers' queries), and a loosely structured search engine (e.g., for answering employees' queries). Motivated by the marked experimental results of our earlier proposed search engine OOXSearch (Taha & Elmasri, 2007), which answers only XML loosely structured queries, we now propose a dual search engine called OOXKSearch, which expands OOXSearch and answers both XML keyword-based and loosely structured queries. While both keyword-based and loosely structured queries do not require the user to be aware of the structure of the data, loosely structured queries require the user to know the labels of the elements containing the keywords and the requested data. Consider that a user wants to know the data *D*, which is contained in an element labeled *E*. If the user knows only the keywords k_1, k_2, .., k_n, which are relevant to *D*, he can submit a keyword-based query in the form: Q (“k_1”, “k_2”, .., “k_n”). If however he knows the label *E* and also the labels of the elements containing the keywords (the labels l_{k_1}, l_{k_2}, ...l_{k_n}), he can submit a loosely structured query in the form: Q*(*l_{k_1}*= “*k_1*”, ...,* l_{k_n} *= “*k_n *“ , E?).*

Extensive research in XML keyword-based querying has been done. These studies could be categorized into four groups. The first expands structured query languages. The second uses keyword-based search techniques for ranking results based on importance and relevance (Balmin, Hristidis, & Papakonstantinon, 2003, 2004; Botev, Shao, & Guo, 2003). The key drawback of those ranking techniques is that they do not consider search semantics. The third proposes modeling the XML document as a graph and processing the graph based on driven schema (Balmin et al., 2003; Balmin et al., 2004; Botev et al., 2003; Cohen & Kanza, 2005). The fourth employs semantic search over XML documents modeled as trees, which makes them the closest to our work (Cohen et al., 2003; Li et al., 2004; Xu & Papakonstantinou, 2005). Despite their success, however, they suffer *recall* and *precision* limitations as a result of basing their techniques on building relationships between data nodes based solely on their labels and proximity to one another while overlooking their contexts. As a result, the proposed search engines may return faulty answers, especially if the XML document contains more than one node having the same label but representing different types, or having different labels but belonging to the same type. Consider for example a node is labeled “title.” We cannot determine whether it refers to a book title or a job title without referring to its parent. We compared OOXKSearch experimentally with the systems proposed by (Cohen et al., 2003; Li et al., 2004; Xu & Papakonstantinou, 2005).

The framework employed by OOXKSearch is inspired by the following observations. If we fragment an XML data model to the simplest semantically meaningful fragments, we will find that each fragment will consist of a parent node and its leaf children data nodes. We call each of these fragments a *Canonical Tree.* Thus, a Canonical Tree is a union of a parent node and its leaf children data nodes (see Figure 3). Leaf children data nodes represent the characteristics of their parent nodes. For instance the data nodes “title” and “year” of publication represent some of the characteristics of their parent node “book.” Therefore, the “book,” “title,” and “year” nodes represent a *meaningful* union. We can treat each Canonical Tree as one entity. A data model is a metaphor of real-word entities. Two real-word entities may have different names but belong to the same type (e.g., a book and an article belong to the same publication type), or may have the same names but refer to two different types (e.g., a “name” of a student and a “name” of a school). To overcome that labeling ***ambiguity***, we observe that if we cluster Canonical Trees based on the reduced essential characteristics and cognitive qualities of their parent nodes, we will identify a number of *clusters*. Each cluster contains Canonical Trees whose parent nodes components belong to the same ontological concept (Kim, Sengupta, Fox, & Dalkilic, 2007). For example, we can have a cluster that contains a Canonical Tree whose parent node component is “book” and also a Canonical Tree whose parent node component is “article,” since both “book” and “article” fall under the same “publication” ontology concept. Thus, though “book” and “article” have different labels, they belong to the same type. On the other hand, a Canonical Tree, whose parent node component is “student” falls under the “person” cluster