Chapter 7
Keyword Search on XML Data

Ziyang Liu
Arizona State University, USA

Yi Chen
Arizona State University, USA

Information search is an indispensable component of our lives. Due to the vast collections of XML data on the web and in enterprises, providing users with easy access to XML is highly desirable. The classical way of accessing XML data is through issuing structured queries, such as XPath/XQuery. However, in many applications it is inconvenient or impossible for users to learn these query languages. Besides, the requirement that the user needs to comprehend data schemas may well be overwhelming and infeasible, as the schemas are likely complex, fast-evolving, or unavailable. A natural question to ask is whether we can empower users to effectively access XML data simply using keyword queries.

Ideally the result of a keyword search on XML will automatically assemble relevant pieces of data that are in different locations but are inter-connected and collectively relevant to the query. There are several advantages of such an approach. First, it can relieve casual users from the steep learning curve of studying structured query languages and data schemas when accessing structured data. Second, it allows users to easily access heterogeneous databases. For instance, for websites with back-ends storing XML data, this approach provides a more flexible search method than the existing solution that uses a fixed set of pre-built template queries. Furthermore, this approach helps to reveal interesting or unexpected relationships among entities. Making XML data searchable will substantially increase the information volume that a user can access, have potential to provide search results with better quality compared with keyword search on textual documents, and thus increase the us-

DOI: 10.4018/978-1-61520-727-5.ch007

ability of XML and make significant impact to people's lives.

The objective of this chapter is to provide an overview of the state-of-the-art in supporting keyword search on XML data, outline the problem space in this area, introduce representative techniques that address different aspects of the problem, and discuss further challenges and promising directions for future work. The problem spectrum that will be covered in this chapter ranges from identifying relevant keyword matches and an axiomatic framework to evaluate different strategies, identifying other relevant data nodes, ranking, indexes and materialized views, to result snippet generation.

1.1 QUERY MODEL AND QUERY RESULT

1.1.1 Keyword Query

A keyword search, as the name indicates, is a query which consists of a set of keywords. A keyword may be required (it must appear in each result) or optional. Each keyword may match elements, attributes names and/or values in the XML document. Some search engines, such as XSEarch (Cohen, Jamou, Kanza, & Sagiv, 2003), has a slightly structured query format: each query term is one of the three formats: *l*:*k*, *l*: or:*k*, where *l* and *k* are keywords, *l* must match name nodes (element names and/or attribute names) and *k* must match value nodes.

1.1.2 Query Result

A result of a keyword search on textual documents is usually a whole document. On the other hand, due to the structure of XML, each result is not an entire XML document, but a fragment of it. When the user does not separate the keywords into required keywords and optional keywords, a system may opt to use AND semantics or OR semantics. As the names suggest, AND semantics requires a query result to have all keywords in the query, while OR semantics only requires a result to have some of the keywords.

XML search engines generate results as subtrees / subgraphs of the XML document. A query result should contain both relevant keyword matches (i.e., the XML nodes that match the keywords) and other relevant nodes that are deemed relevant. The details of how query results are constructed will be discussed later in this chapter.

In this chapter, "query" and "search" are used interchangeably.

1.2 WHAT ARE THE PROBLEMS INVOLVED?

Due to the structure of XML data, processing keyword search on XML requires fundamentally different techniques than on textual documents. In fact, the structure of XML data provides us with better opportunities than textual documents to generate meaningful results, as long as they can be properly exploited. We summarize processing XML keyword search as the following problems, including generating query results (including finding relevant matches and relevant non-matches), improving efficiency, ranking and generating result snippets.

1. **Identifying relevant keyword matches.** Each keyword can have multiple matches in the XML document. Not all of them are relevant to the query. Consider query "*Greg, position*", on the XML data in Figure 1. By issuing this query, the user would like to find the *position* of the player named *Greg*. In Figure 1, there are multiple "*position*" nodes, but only the one labeled (20) is relevant to the query.
2. **Identifying relevant non-matches.** Merely returning relevant matches to users as query results is undesirable. A user who issues