The XXL Search Engine: Ranked Retrieval of XML Data using Indexes and Ontologies

Anja Theobald
University of the Saarland
D-66041 Saarbrücken, Germany
theobald@cs.uni-sb.de

Gerhard Weikum
University of the Saarland
D-66041 Saarbrücken, Germany
weikum@cs.uni-sb.de

1. Motivation

XML is becoming the standard for integrating and exchanging data over the Internet and within intranets, covering the complete spectrum from largely unstructured, ad hoc documents to highly structured, schematic data. For searching information in open environments such as the Web or intranets of large corporations, ranked retrieval is more appropriate: a query result is a rank list of XML elements in descending order of relevance. We have developed a core language, coined XXL for "flexible XML search language" [1], for ranked retrieval of XML data using regular element path expressions and search conditions over element contents. For similarity search we have introduced a new operator "~", which can be used for both element content comparisons and approximate matching of element names. On the XML Shakespeare play collection, for example, we can search for scenes where a woman talks about leadership in the presence of Macbeth by the XXL query:

*SELECT * FROM INDEX*
WHERE ~drama.#.scene AS A AND A.speech AS B
AND B.speaker ~ "woman" AND B.line ~"leader"
AND A.speech AS C AND C.speaker = "Macbeth".

The result would include semantically relevant scenes with Lady Macbeth or the witches even if the words "woman" or "leader" do not literally appear. The evaluation of similarity conditions of XXL queries is based on a hierarchical ontology for element names and on term-frequency-based IR-style relevance estimations for element contents, and provides relevance scores for elementary search conditions between 0 and 1. The scores from "local" similarity tests are combined into "global" relevance rankings using simple probabilistic arguments.

2. Overview of the XXL Search Engine

The XXL search engine is a client-server application. It consists of three types of components: service components, algorithmic components and data components. The client application program is a service component: the Visual XXL GUI (a Java applet). The server programs consist of 1) service components: the crawler and the query processor (both Java servlets), 2) algorithmic components: parsing and indexing documents, parsing and checking XXL queries, and 3) data components: structures and their methods for storing various kinds of information like the element path index (EPI), the element content index (ECI), and the element-name ontology index (NOI). XXL client and server exchange queries and query results via http. The XXL crawler provides XML data to the indexer programs for building index structures.

ACM SIGMOD 2002, June 4-6, 2002, Madison, Wisconsin, USA.

3. Index Structures and Query Processing

For the evaluation of XXL queries we use three index structures: element path index (EPI), element content index (ECI), and element-name ontology index (NOI).

An XXL query is evaluated as follows [2]. The Where clause of an XXL query is a logical conjunction of n regular expressions for search conditions over XML element paths. The query processor (QP) decomposes the given query into n subqueries and constructs a graph-based representation for each subquery similar to a finite state automaton. The global order of evaluating the n subqueries and the local evaluation strategy (e.g., top-down vs. bottom-up) for each subquery are automatically chosen by the QP.

For each subquery, simple path expressions with element names and the wildcard symbol # are looked up in the EPI. For example, all occurrences of a pattern *play.scene.#.speaker* can be retrieved from the EPI. Content conditions such as *speaker = "Macbeth"* are evaluated by the ECI, a text index on element and attribute contents. For "semantic" similarity conditions such as *speaker ~ "woman"* the ECI yields approximate matches and a relevance score based on IR-style tf*idf measures and "semantic distances" between concepts in the ontology. For example, *"lady"* would be a good match to the above condition, and *"witch"* would still be considered a match with a lower score. Finally, for semantic similarity conditions on element names, for example, *~drama*, the NOI is used to expand the query in order to capture semantically related element names such as *play*; again, a similarity score is computed for result ranking.

The ranked lists of relevant subgraphs from the index-based subquery evaluation are composed into a list of global graphs, each of which has assigned to it a global relevance score derived from the local scores by elementary probability computation. The QP finally extracts the result as specified by the Select clause and constructs an XML document that is returned to the XXL client.

The demo shows the query evaluation of the XXL prototype.

4. References

[1] A. Theobald, G. Weikum: Adding Relevance to XML, 3rd International Workshop on the Web and Databases, Dallas, Texas, 2000, LNCS 1997, Springer, 2001.

[2] A. Theobald, G. Weikum: The Index-based XXL Search Engine for Querying XML data with Relevance Ranking, 8th EDBT Conference, Prague, Czech Republic, 2002.