On the Effectiveness of Flexible Querying Heuristics for XML Data

Zografoula Vagena, Latha Colby, Fatma Özcan,
Andrey Balmin, and Quanzhong Li

IBM Almaden Research Center
650 Harry Road, San Jose, CA

Abstract. The ability to perform effective XML data retrieval in the absence of schema knowledge has recently received considerable attention. The majority of relevant proposals employs heuristics that identify groups of meaningfully related nodes using information extracted from the input data. These heuristics are employed to effectively prune the search space of all possible node combinations and their popularity is evident by the large number of such heuristics and the systems that use them. However, a comprehensive study detailing the relative merits of these heuristics has not been performed thus far. One of the challenges in performing this study is the fact that these techniques have been proposed within different and not directly comparable contexts. In this paper, we attempt to fill this gap. In particular, we first abstract the common selection problem that is tackled by the relatedness heuristics and show how each heuristic addresses this problem. We then identify data categories where the assumptions made by each heuristic are valid and draw insights on their possible effectiveness. Our findings can help systems implementors understand the strengths and weaknesses of each heuristic and provide simple guidelines for the applicability of each one.

1 Introduction

The expressive power of XML and the many data representation alternatives that it provides can lead to the creation of datasets with very complex schemata. In certain cases, such as Web XML documents created in an ad-hoc manner, a schema might not even exist. These reasons, coupled with the fact that the main usage of XML, as a standard for data sharing, necessitates the ability to query heterogeneous data sources, have made the employment of existing structured XML query languages, such as XPath and XQuery [18], cumbersome for XML data retrieval. Without knowledge of the exact structure of the underlying data it is very difficult to come up with the right query, because XPath expressions follow the document structure. Even if the user had this knowledge, the need to query multiple heterogeneous sources may require the generation of a different query for each data source (either directly from the user or from a complex query translation module) and as a result makes the querying process both cumbersome and error prone.

D. Barbosa et al. (Eds.): XSym 2007, LNCS 4704, pp. 77–91, 2007.

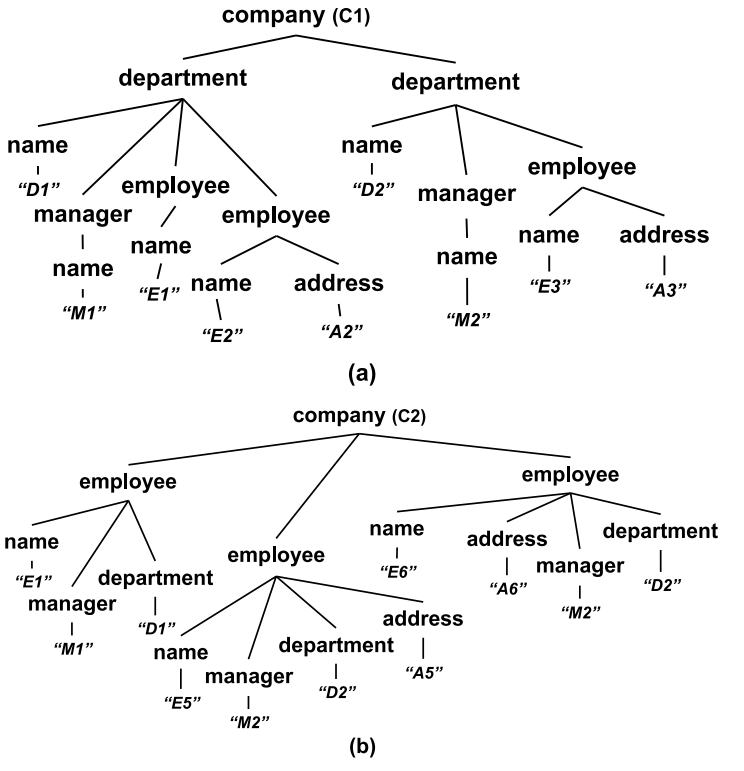

Fig. 1. Querying Heterogeneous XML Data Collections

To illustrate the problem consider the example in Figure 1, which shows two different schemata for *departments* and *employees* within two different companies, $C1$ and $C2$, which have recently merged. The data in company $C1$ (Figure 1a) are grouped by *department*, while the data in company $C2$ (Figure 1b) are grouped by *employee* that works in the company. If a user wants to retrieve information about the employees that work in department $D1$ using XPath to perform the retrieval, she has to issue two, structurally different queries over the two datasets, namely $/company/department[name = "D1"]/employee$ and $/company/employee[department = "D1"]$ respectively. With an increasing number of data sources the retrieval task will become increasingly complex.

To tackle the problem, a number of XML search engines [16,8,7,17] have been developed, which aim to leverage the keyword search paradigm to support XML data retrieval. The main advantage of keyword search is its simplicity. In particular, users do not have to know a complex query language and can query any dataset without prior knowledge of the structure of the underlying data. Nevertheless, pure keyword search is not always the appropriate querying paradigm. First, as pinpointed in [11] it is often difficult and sometimes impossible to convey semantic knowledge (e.g. that the user is looking for the *manager* of a particular *department*). Second,