Efficient Cache Answerability for XPath Queries

Pedro José Marrón and Georg Lausen
University of Freiburg, Institute of Computer Science
Georges-Köhler-Allee, 79110 Freiburg, Germany
{pjmarron,lausen}@informatik.uni-freiburg.de

Abstract. The problem of cache answerability has traditionally been studied over conjunctive queries performed on top of a relational database system. However, with the proliferation of semistructured data and, in particular, of XML as the de facto standard for information interchange on the Internet, most of the assumptions and methods used for traditional systems – and cache answerability is no exception – need to be revisited from the point of view of the semistructured data and query model. In this paper, we present a formal framework for the efficient processing of XPath queries over XML documents in a cache environment that is based on the classic rewriting approach. Furthermore, we provide details on the implementation of our formal methods on top of HLCaches, an LDAP-based distributed caching system for XML, and argue that our approach is more efficient than traditional query rewriting algorithms while, at the same time, supporting the full expressive power of XPath queries.

Keywords: Semistructured data, cache answerability, query rewritability, XML, XPath, LDAP

1 Introduction

Cache answerability has been traditionally studied in the realm of conjunctive predicates and queries performed on top of relational database systems [Lev00], but the increasing interest in recent years on the characteristics and capabilities of semistructured models and, in particular, XML [BPSMM00], have lead to the restatement of the cache answerability problem in terms of the semistructured data and query model [CGLV00, KNS99, PV99]. Furthermore, the proliferation of techniques to perform data integration (let it be semistructured or not) on the Internet strive the need for efficient cache mechanisms.

The use of XPath [CD99] and XPath-based models for the querying and processing of semistructured data has changed the focus of the rewriting algorithms from conjunctive predicates to regular path queries [CGLV00], or other query languages specifically designed for a particular semistructured data model [PV99].

Other query caching systems, like [LRO96], [DFJ+96] or [QCR00], do not take into consideration semistructured data, and although interesting in their approach, cannot be used in the context our model can be brought up.
The approach we take in our work, and therefore, the focus of this paper, is on the definition of a very simple, but highly efficient general-purpose formal model that allows us to tackle the problem of cache answerability for XML from a more pragmatic perspective than the one usually taken by traditional papers on the topic. The generality of our model enables its implementation on any XPath-aware caching system, and in order to show its feasibility, we have implemented it as part of HLCaches [ML01, Mar01], a hierarchical LDAP-based caching system for XML.

In our system, the methods and algorithms described throughout this paper serve as the basis for the efficient processing of XPath queries in the distributed caching environment offered by HLCaches, since it allows the definition of partial XPath query evaluation techniques, query preprocessing mechanisms, and parallel processing routines that are crucial for the maintenance of the level of availability and processing capabilities expected from a distributed caching system.

This paper is structured as follows: Section 2 presents a formal description of the XPath query model needed to understand the reformulation of the cache answerability problem detailed in section 3. Section 4 provides an insight in some of the more important implementation issues related to our model, and section 5 concludes this paper.

2 XPath Query Model

As specified in the XPath standard [CD99], the primary purpose of the XPath query language is to address parts of an XML document, usually represented in the form of a tree that contains element, attribute and text nodes.

An XPath Query $Q_X$ is formed by the concatenation of path expressions that perform walk-like operations on the document tree retrieving a set of nodes that conform to the requirements of the query. Each expression is joined with the next by means of the classical Unix path character ‘/’.

**Definition 1 (XPath Query).** An XPath Query $Q_X$ is defined as:

$$Q_X = /q_0/q_1/\ldots/q_n,$$

where $q_i$ is an XPath subquery defined below, and ‘/’ the XPath subquery separator.

**Definition 2 (XPath Subquery).** An XPath Subquery $q_i$ is a 3-tuple

$$q_i = (C_i, w_i, C_{i+1}),$$

where:

- $C_i$ is a set of XML nodes that determine the input context.
- $w_i$ is the Path Expression to be applied to each node of the input context (defined below).