XPath Query Satisfiability is in PTIME for Real-World DTDs

Manizheh Montazerian[1], Peter T. Wood[1], and Seyed R. Mousavi[2]

[1] Birkbeck, University of London, London WC1E 7HX, UK
{gmont05,ptw}@dcs.bbk.ac.uk
[2] Isfahan University of Technology, Isfahan, Iran
srm@cc.iut.ac.ir

Abstract. The problem of XPath query satisfiability under DTDs (Document Type Definitions) is to decide, given an XPath query p and a DTD D, whether or not there is some document valid with respect to D on which p returns a nonempty result. Recent studies in the literature have shown the problem to be NP-hard or worse for most fragments of XPath. However, in this paper we show that the satisfiability problem is in PTIME for most DTDs used in real-world applications. Firstly, we report on the details of our investigation of real-world DTDs and define two properties that they typically satisfy: being *duplicate-free* and being *covering*. Then we concentrate on the satisfiability problem of XPath queries under such DTDs. We obtain a number of XPath fragments for which the complexity of the satisfiablity problem reduces to PTIME when such real-world DTDs are used.

Keywords: XPath, Satisfiability, Document Type Definitions.

1 Introduction

With XML becoming the standard for data exchange, substantial work has been done on XML query processing and optimization [1,5,7,10,11,12,14]. Much of the work on query optimization has focused on XPath, since XPath is widely used in XML-related applications to select sets of nodes from an XML document tree. Particularly when an XPath query is to be evaluated over documents known to be valid with respect to a DTD (Document Type Definition), it is possible that the query might be *unsatisfiable*, that is, the query always returns an empty result, no matter what document (valid with respect to the DTD) is queried. Relatively little work has been done on detecting whether a given XPath query is satisfiable [2,6,8,9]. However, it is potentially important to detect unsatisfiable XPath queries and optimize queries to remove expressions that will always return an empty result set. Indeed, Lakshmanan *et al.* show that checking satisfiability as a first step in query processing often yields substantial savings in overall query processing time [9].

XPath supports a wide variety of operators whose presence or absence affects the complexity of the satisfiability problem. This has led to the study of various XPath fragments that include only certain operators. For example, in this

D. Barbosa et al. (Eds.): XSym 2007, LNCS 4704, pp. 17–30, 2007.

paper we study the fragment with child axis (/), descendant axis (//), qualifiers ([ ]), wildcard (*) and union (∪). As in [10], we denote this fragment by $XP^{\{/,[],*,//,\cup\}}$, indicating the operators permitted. Larger fragments allow operators such as negation, additional axes such as parent, ancestor and sibling, as well as comparisons involving data values or node identities.

Example 1. The XMark benchmark project[1] is based on an online auction application. A fragment of the XMark DTD is given below:

```
site          (regions, categories, catgraph, people, open_auctions,
               closed_auctions)
categories    (category+)
category      (name, description)
description   (text | parlist)
open_auctions (open_auction*)
open_auction  (initial, reserve?, bidder*, current, privacy?, itemref,
               seller, annotation, quantity, type, interval)
```

with `site` being the document (top-level) element.The XPath query

```
/site/open_auctions/open_auction[bidder][reserve]/seller
```

selects `seller` nodes that are children of `open_auction` nodes that have both a `bidder` and `reserve` child. It is easy to see that this query is satisfiable on documents valid with respect to the above DTD fragment.

In the full DTD, a `description` can occur as a descendant of more than one element, so one might write

```
/site//description[text][parlist]
```

to retrieve all `description` nodes that have both `text` and `parlist` children. However, this query is unsatisfiable with respect to the DTD, because a `description` can have only one of `text` or `parlist` as a child, not both.

Hidders investigates the satisfiability of XPath 2.0 expressions [8]. Various XPath fragments are classified as either in PTIME or NP-hard. Deciding satisfiability of XPath expressions in the context of a DTD/schema is one of the open problems mentioned in that paper. This problem is dealt with in [9], which considers a tree pattern formalism with expressiveness incomparable to XPath. The language expresses positive tree pattern queries, with data value equality and inequality along with a node-equality test. It shows that the satisfiability problem is NP-complete for several restrictions of this pattern language in the absence of DTDs. It also identifies cases in which satisfiability of tree pattern queries together with additional constraints, with and without schema, can be solved in polynomial time and develops algorithms for this purpose.

The most relevant work to ours is [2]. The authors consider a variety of XPath fragments widely used in practice, and investigate the impact of different XPath operators on satisfiability analysis. They study the problem for negation-free

[1] `http://monetdb.cwi.nl/xml/`