

XPath Query Satisfiability is in PTIME for Real-World DTDs

Manizheh Montazerian, Peter T. Wood

{gmont05,ptw}@dcs.bbk.ac.uk

School of Computer Science and Information Systems
Birkbeck, University of London, UK

Seyed R. Mousavi

seyedrm@cc.iut.ac.ir

Isfahan University of Technology, Isfahan, IRAN

Motivation

- XPath is used in numerous places in XML
- makes sense to study optimization of XPath queries
- in the presence of DTDs in particular
 - parts of a query may be redundant
 - the query may not be satisfiable
- Lakshmanan et al. (2004) show that checking satisfiability can yield savings in overall query processing time
- checking satisfiability is hard in general
- might it be easier if (real-world) DTDs turn out to be restricted in some way

Outline

- Example of DTD and XPath unsatisfiability
- Definitions
 - XPath fragments, satisfiability
- Previous work
- Definitions
 - duplicate-free DTDs, covering DTDs
- Satisfiability complexity results
 - duplicate-free DTDs, covering DTDs
- Real-world DTDs
- Conclusion and future work

XMark DTD Fragment

```
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)
```

site is the document (top-level) element

Example of XPath Unsatisfiability

- XPath query
`/site/open_auctions/
open_auction[bidder] [reserve]/seller`
is satisfiable on documents valid with respect to the above
DTD fragment
- XPath query
`/site//description[text] [parlist]`
is unsatisfiable with respect to the DTD

Definitions—XPath Fragments

- syntax used in this talk is given by the following grammar:

$$q \rightarrow \text{'/' } p$$

$$p \rightarrow p \text{'/' } p \mid p \text{'//'} p \mid p \text{'\cup'} p \mid p \text{'[' } p \text{'\]' } \mid \text{'*'} \mid n \mid \text{'.'}$$

where q is the start symbol, n is an element name and '.' refers to the context node

- fragments denoted by indicating those operators supported
- so full fragment denoted by $\text{XP}\{/, [, *, //, \cup\}$, since child axis ($/$), descendant axis ($//$), qualifiers ($[]$), wildcard ($*$) and union (\cup) permitted

Definitions—XPath Satisfiability

- adapted from Benedikt et al. (2005)
- expression p is **satisfiable** if there is an XML tree T such that the answer of p on T is not empty, denoted $T \models p$
- given DTD D , we denote the fact that an XML tree **satisfies** (or is **valid** with respect to) D by $T \models D$
- given DTD D and a query p , an XML tree T **satisfies** p and D , denoted by $T \models (p, D)$, iff $T \models p$ and $T \models D$
- for XPath fragment \mathcal{X} , the **XPath satisfiability problem** $\text{SAT}(\mathcal{X})$ is, given a DTD D and a query p in \mathcal{X} , is there an XML tree T such that $T \models (p, D)$

Previous Work on Satisfiability

- Hidders (DBPL'03)
 - not with respect to DTDs
- Lakshmanan, G. Ramesh, H. Wang, and Z. Zhao (VLDB'04)
 - tree pattern queries
- Geerts and Fan (DBPL'05)
 - sibling axes
- Benedikt, Fan and Geerts (PODS'05)
 - include negation, data values, parent and ancestor axes,
...

Benedikt et al.'s Work on Satisfiability

- show that $\text{SAT}(\text{XP}\{/,//,*,\cup\})$ is in PTIME whereas corresponding containment problem is EXPTIME-complete (Neven And Schwentick, LMCS, 2006)
- however, the following are NP-hard:
 - $\text{SAT}(\text{XP}\{/,[],*\})$
 - $\text{SAT}(\text{XP}\{[],//\})$
 - $\text{SAT}(\text{XP}\{/,[],\cup\})$
- above results still hold for **non-recursive** DTDs
- show that $\text{SAT}(\text{XP}\{/,[],*,//,\cup\})$ under **disjunction-free** DTDs is in PTIME

Definitions—Duplicate-free and Covering

- let R be a regular expression, Σ the set of symbols appearing in R
- R is **duplicate-free** if each symbol in Σ occurs exactly once in R
- R **covers** Σ , or simply that R is **covering**, if there is a string in $L(R)$ that contains every symbol in Σ
- DTD D is called **duplicate-free (covering)** if and only if each content model in D is duplicate-free (covering)

Examples—Duplicate-free

- all rules in the XMark fragment are duplicate-free
- non-duplicate-free example, from the XML Schema DTD:

```

schema ((include | import | redefine | annotation)*,
        ((simpleType | complexType | element
         | attribute | attributeGroup | group
         | notation), (annotation)*)* )

```

where the element name `annotation` is repeated
- definition of duplicate-free is syntactic
- e.g. $a?, b$ and $(a, b)|b$ denote the same language, but only the former expression is duplicate-free

Examples—Covering

- all rules in XMark DTD fragment are covering, except
description (text | parlist)
since the language denoted by (text | parlist) does not contain a sequence that includes both element names
- every disjunction-free rule is covering, but ...
- $(a|b)^*$ is covering, but not disjunction-free

Results—Duplicate-free DTDs

Theorem For duplicate-free DTDs, $\text{SAT}(\text{XP}\{/, []\})$ is in PTIME.

Corollary For queries p in $\text{XP}\{/, []\}$ and DTDs D such that each symbol in p appears in a duplicate-free rule in D , $\text{SAT}(\text{XP}\{/, []\})$ is in PTIME

Theorem For duplicate-free DTDs, the following problems are NP-hard:

1. $\text{SAT}(\text{XP}\{/, [], *\})$
2. $\text{SAT}(\text{XP}\{[], //\})$
3. $\text{SAT}(\text{XP}\{/, [], \cup\})$

Results—Covering DTDs

Proposition Given a DTD D , deciding whether D is covering is NP-complete.

Theorem Under covering DTDs, $\text{SAT}(\text{XP}\{/, [], *, //, \cup\})$ is in PTIME.

Corollary For queries p in $\text{XP}\{/, [], *, //, \cup\}$ and DTDs D such that each symbol in p appears in a covering rule in D , $\text{SAT}(\text{XP}\{/, [], *, //, \cup\})$ is in PTIME

Real-World DTDs

DTD Name	Number of Rules	Covering		Non-covering	
		D-f	Dup.	D-f	Dup.
XML Schema	26	19	1	6	0
RSS-091	24	24	0	0	0
XHTML1-strict	77	74	1	2	0
DBLP	37	37	0	0	0
XMark DTD	77	76	0	1	0
SigmodRecord	11	11	0	0	0
News ML	116	112	0	4	0

Real-World DTDs

For 100 DTDs:

	Duplicate-free	Duplicates
Covering	47	8
Non-covering	28	17

In terms of rules:

DTD Name	Number of Rules	Covering		Non-covering	
		D-f	Dup.	D-f	Dup.
Total	5534	5053	236	201	44
Percentage	100%	91.3%	4.3%	3.6%	0.8%

over 95% of rules are covering

Conclusion and future work

- properties of real-world DTDs and their impact on the satisfiability problem
- examined several real-world DTDs and discovered a new property, called covering, which most of them preserved
- satisfiability problem of $XP\{/, [], *, //, \cup\}$ reduces to PTIME when the underlying DTD is covering
- possibly combine and extend the classification of DTDs
- investigate whether XPath **containment** is simplified by covering DTDs