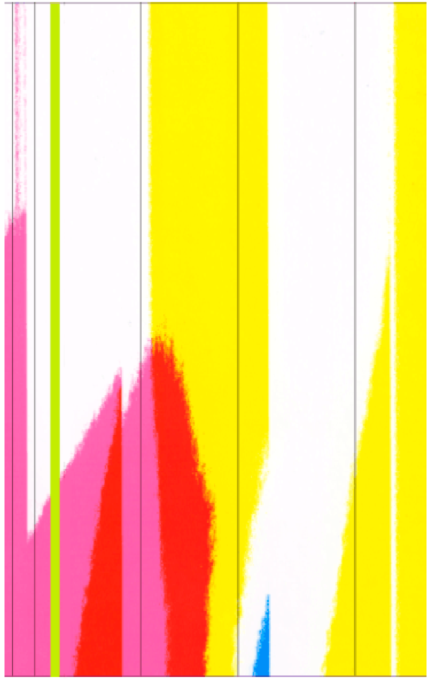


On XPath Dialects with Variables

Emmanuel Filiot, **Joachim Niehren**, Jean-Marc Talbot, Sophie Tison

INRIA Futurs, Lille, Mostrare project

October 24, 2006

Plan

- **Logical queries in trees**
- **Standard XML query languages and Core XPath 2.0**
- **First-order expressiveness**
- **NP-completeness**
- **An efficient FO-expressive fragment**

XML Documents

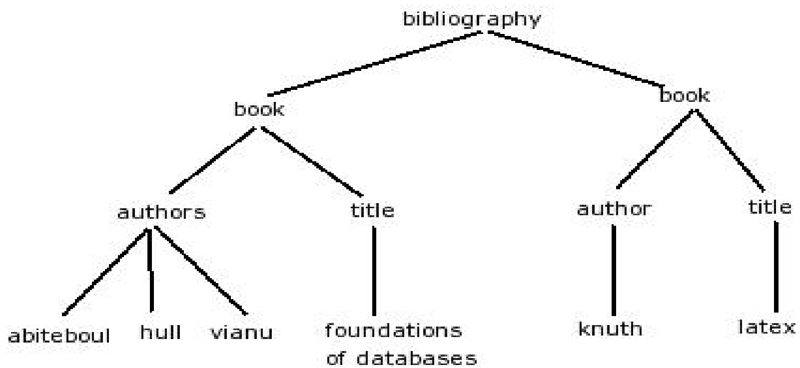
- **standard format for data exchange**
- parse into **trees**
  - unranked, sibling-ordered
  - nodes contain data values (text)
  - unordered attributes
- ignore details of data model and values in this talk

Unranked Sibling-Ordered Trees

labels $a \in \Sigma$

trees $t \in T_\Sigma ::= a(t_1, \ldots, t_n)$ where $n \geq 0$

example



N-ary Queries in Trees

function q mapping trees to sets of n-tuples of nodes

$$\forall t \in T_\Sigma : \; q(t) \subseteq \texttt{nodes}(t)^n$$

- n=0: **Boolean queries** are **tree languages**
- n=1: **monadic queries** select nodes
- n=2: **binary queries** select pairs of nodes

Trees as Logical Structures

- domain is $\texttt{nodes}(t)$
- binary relations: $\texttt{child}^*, \texttt{nextsib}^*$
- unary relations: lab_a for all $a \in \Sigma$
- enough relations for FO, but more relations are needed for constraints.

Logical Languages

First-order logic (FO)

$$\phi ::= \texttt{child}^*(x,y) \mid \texttt{nextsib}^*(x,y) \mid \neg\phi \mid \exists x \phi$$

Monadic second-order logic (MSO)

$$\phi ::= \ldots \mid x \in X \mid \exists X.\phi$$

Logical Queries

- let ϕ be a logical formulas
- let $\overline{x} = (x_1, \ldots, x_n)$ be a sequence of n variables
- they define an n-ary queries such that for all trees $t \in T_\Sigma$:

$$q_{\phi,\overline{x}}(t) = \{(\alpha(x_1), \ldots, \alpha(x_n)) \mid t, \alpha \models \phi\}$$

Parametrized Logical Languages

Fix **set of queries Γ as parameter**, for instance:

- $\{\texttt{child}, \texttt{nextsib}, \texttt{root}, \texttt{leaf}, \texttt{child}^*, \texttt{nextsib}^*\}$
- all FO-definable binary queries
- all MSO-definable binary queries

Positive propositional formulas

$$\phi ::= r(x_1, \dots, x_n) \mid x{=}y \mid \phi \wedge \phi' \mid \phi \vee \phi' \mid \exists x \phi \qquad \text{where } r \in \Gamma$$

Constraints (conjunctive queries)

$$\phi ::= r(x_1, \dots, x_n) \mid x{=}y \mid \phi \wedge \phi' \mid \exists x \phi \qquad \text{where } r \in \Gamma$$

Standard XML Query Languages

- **XPath 1.0**
  - **select nodes in trees (monadic queries)**
  - **no variables**
- **XPath 2.0**
  - **select n-tuples of nodes in trees (n-ary queries)**
  - **with variables**
- **XQuery 1.0 / XSLT 2.0**
  - **tree-to-tree transformations**
  - **use XPath 2.0**

Core XPath 2.0

$$\begin{array}{lcl} s & ::= & \texttt{child} \mid \texttt{nextsib} \mid \texttt{child}^{-1} \mid \texttt{nextsib}^{-1} \\ p & ::= & s \mid s^* \mid x \mid a \mid p/p' \mid p \cup p' \mid [p] \mid \neg p \end{array}$$

navigational language inspired by **modal logics**

- define **navigation paths from start nodes to end nodes**
- **binary queries**: pairs (start-node, end-node)
- **monadic queries**: end-nodes when starting at the root

variables to capture nodes as in **hybrid logic** (Blackburn 95)

- define **n-ary queries** by navigations paths with n-variables
- not available in XPath 1.0

Example

select all author-title-pairs in books of a bibliography:

Core XPath 2.0 $\texttt{child}^*/book/$
$[\texttt{child}/author/\texttt{child}/x]/$
$[\texttt{child}/title/\texttt{child}/y]$

FO $\exists z_1, z_2, z_3, z_4$
$(\texttt{child}^*(z_1, z_2) \wedge lab_{book}(z_2) \wedge$
$\texttt{child}(z_2, z_3) \wedge lab_{author}(z_3) \wedge \texttt{child}(z_3, x)$
$\texttt{child}(z_2, z_4) \wedge lab_{title}(z_4) \wedge \texttt{child}(z_4, y)\ )$

FO-Expressiveness

Theorem

All FO-queries can be expressed in the Core XPath 2.0, and vice versa.

Proof.

- binary FO-queries can be expressed in Core XPath 2.0 (Marx 2005)
- propositional connectives can be expressed in XPath 2.0.
- quantifier elimination (Schwentick 2000):
  - Ehrenfeucht-Frässe games
  - Shelah's composition method

□

Algorithmic Problems

- **Query answering**: compute the set of all valid variable assignments for a given a tree.

$$\lambda(\phi, \overline{x}, t).\ q_{\phi,\overline{x}}(t)$$

- **Query non-emptiness**: decide whether the set of query answer is nonempty

$$\lambda(\phi, \overline{x}, t).\ q_{\phi,\overline{x}}(t) \neq \emptyset$$

Query Non-Emptiness is NP-hard

Booleans

- tree $t = b(0, 1)$
- define $\texttt{child}_0 = \texttt{child}/[0]$ and $\texttt{child}_1 = \texttt{child}/[1]$

Encode 3-Sat

- example: $(x \vee y) \wedge (\neg x \vee z)$
- Core XPath 2.0 encoding:

$$[lab_b]/[\texttt{child}_1/x \cup \texttt{child}_1/y)]/[\texttt{child}_0/x \cup \texttt{child}_1/z]$$

query on t is non-empty iff all clauses are satisfiable.

Query Expressiveness and Efficiency

Overview

Core XPath 2.0	$=$	FO	$<$	MSO	$=$	tree automata
PSpace		PSpace		PSpace		P

Question

Are there FO-expressive fragments of Core XPath 2.0 which allow for efficient query answering?

Only partial answers so far

- Gottlob & Koch & Pichler (2004) present efficient algorithms for monadic queries in Core XPath 1.0 (without variables).
- Marx (2005) presents an efficient algorithm for binary queries in Core XPath 2.0.

Efficient Fragment of Core XPath 2.0

Restrictions

NVS(/) no sharing of variables between the path on two sides of composition operators

NV($\neg$) no variable below negation

Query answering algorithm

- valid more generally
- for a parametrized path language without negation

Parametrized Path Composition Language

Let Γ be a set of binary queries.

Composition language $C(\Gamma)$

$$\psi \in C(\Gamma) ::= r \mid \psi/\psi' \mid x \mid [\psi] \mid \psi \cup \psi' \qquad \text{where } r \in \Gamma$$

If $\texttt{child}^* \in \Gamma$ then $C(\Gamma)$ has the same expressiveness as **positive propositional formulas over** Γ up to linear time conversions:

$$r(x,y) \Rightarrow \texttt{child}^*/x/r/y \qquad \psi \wedge \psi' \Rightarrow [\psi]/[\psi']$$

Lemma (Relation to Core XPath 2.0)

$C(bin(Core\ XPath\ 2.0)) = Core\ XPath\ 2.0 \cap NV(\neg)$
Restriction $NVS(/)$ is identical on both sides.

Answering $C(\Gamma)$ Queries satisfying $NVS(/)$

Ideas of the Algorithm

- test emptiness of all subqueries at all nodes of the tree.
- needs a new sharing trick, to treat $\cup$ on the left of $/$
- process queries recursively:
  - always filter unsatisfiable cases
  - eliminate duplicates
  - memoize auxiliary values

Theorem

Computing $q_{\psi,\bar{x}}(t)$ for paths ψ of Core XPath 2.0 satisfying $NVS(/)$ and $NV(\neg)$ on trees t is in time $O(|\psi|^2|t|^2(1+|q_{\psi,\bar{x}}(t)|)$.

Comparison with Acyclic Conjunctive Queries

- Paths in $C(\Gamma)$ without $\cup$ and satisfying $NVS(/)$ can be translated to acyclic conjunctive queries
- With $\cup$ a different translation is needed, which fails to map to disjunctions of acyclic conjunctive queries.

FO-Expressiveness

Theorem

The restriction of Core XPath 2.0 satisfying $NV(\neg)$ and $NVS(/)$ is FO-expressive (and permits efficient query answering).

Proof.

- all binary FO-queries can be expressed (Marx 05)
- paths of the composition language $C(bin(FO))$ restricted by $NVS(/)$ can be expressed.
- new quantifier elimination theorem concludes.

□

Theorem (Quantifier elimination)

$C(bin(FO)) = FO$. *Compositions restricted by* $NVS(/)$ *are enough, too.*

Quantifier Elimination for MSO

Theorem

$C(bin(MSO)) = MSO$.

Future Work

- efficient enumeration algorithms for fragments of Core XPath 2.0 and $C(\Gamma)$.
- towards XML transformations.

Conclusions

- Core XPath 2.0 is FO-expressive
- Efficient FO-expressive fragments exists
- Path perspective is essential to define restriction $NVS(/)$.
- Otherwise, propositional formulas would be good enough.