Query Answering for Core XPath 1.0 and Beyond

Gottlob and Koch's Algorithm Revisited

Joachim Niehren

Mostrare project, INRIA and LIFL, Lille

November 10, 2020

Examples

Select all books in bibliographie

XPath short: $//book$

XPath long: $\text{child}^* :: book$

FO logic: $\text{child}^*(root, x) \wedge lab_{book}(x)$

Select all co-authors of Abiteboul in books of bibliographie

XPath short: $//book[author ='' abitebool'']/$
$author[not[self ='' abiteboul'']]$

XPath long: $\text{child}^* :: book[\text{child} :: author[self ='' abiteboul'']]/$
$\text{child} :: author[not[self ='' abiteboul'']]$

FO logic:

$$\exists y_1 \ (\text{child}^*(root, y_1) \wedge lab_{book}(y_1)$$
$$\wedge \exists y_2(\text{child}(y_1, y_2) \wedge lab_{author}(y_2) \wedge content_{''abiteboul''}(y_2))$$
$$\wedge \text{child}(y_1, x) \wedge lab_{author}(x) \wedge \neg content_{''abiteboul''}(x))$$

Core XPath 1.0

Syntax where $a \in \Delta$:

$$
\begin{array}{llll}
\text{filter} & f & ::= & * \mid a \mid \mathit{not}[f] \mid f[p] \\
\text{axis} & \mathrm{r} & ::= & \text{child} \mid \text{next\_sib} \mid \mathrm{r}^{-1} \mid \mathrm{r}^{*} \mid \text{self} \mid \ldots \\
\text{paths} & p & ::= & \mathrm{r} :: f \mid p/p' \mid p \cup p' \mid /p
\end{array}
$$

Semantics for unranked tree $t \in T_\Delta$:

$$
\begin{array}{c}
\mathit{eval}^t(f) \subseteq \mathit{nod}(t) \\
\mathit{eval}^t(\mathrm{r}) \subseteq \mathit{nod}(t)^2 \\
\mathit{eval}^t(p) \subseteq \mathit{nod}(t)^2
\end{array}
$$

Simple Filters

Adding label tests ℓ for negative axis

$$
\begin{array}{rcl}
f, f' & ::= & a \mid \ell \mid f \wedge f' \;\mid\; [r :: f] \mid \mathit{not}[f] \mid f \vee f' \\
\ell & ::= & \mathit{root} \mid \mathit{leaf} \mid \mathrm{left} \mid \mathrm{right} \\
r & ::= & \mathrm{self} \mid \mathrm{child} \mid \mathrm{child}^{-1} \mid \mathrm{child}^{*} \mid \mathrm{child}^{*-1} \\
 & \mid & \mathrm{next\_sib} \mid \mathrm{next\_sib}^{-1} \mid \mathrm{next\_sib}^{*} \mid \mathrm{next\_sib}^{*-1}
\end{array}
$$

Positive Simple Filters

Expressing label test for negative axis

The additional label tests ℓ can be expressed as follows:

$$\begin{array}{rcl} \mathit{root} & = & \mathit{not}[\text{child}^{-1} :: *] \\ \mathit{leaf} & = & \mathit{not}[\text{child} :: *] \\ \text{left} & = & \mathit{not}[\text{next\_sib}^{-1} :: *] \\ \text{right} & = & \mathit{not}[\text{next\_sib} :: *] \end{array}$$

Positive simple filters

Simple filters without negation.

- Label tests are no more redundant.
- Disjunction is no more redundant.

Eliminate Negation

In Axis Only Filters

Function $elim'$ removes negated axis-only filters $neg([r :: *])$:

$$
\begin{aligned}
&elim'(not[\text{child} :: *]) = leaf \\
&elim'(not[\text{child}^{-1} :: *]) = root \\
&elim'(not[\text{next\_sib} :: *]) = \text{left} \\
&elim'(not[\text{next\_sib}^{-1} :: *]) = \text{right} \\
&elim'(not[r^* :: *]) = a[b] \text{ where } a \neq b \text{ fixed arbitrarily}
\end{aligned}
$$

Eliminate Negation 2

Function *elim* maps maps simple filters to positive simple filters by pushing down negation into label tests and axis, and removing double negations:

Push Negation Down

$$
\begin{aligned}
&elim(not[*]) = a[b] \text{ where } a \neq b \text{ fixed arbitrarily}\\
&elim(not[a]) = \vee_{b \in \Delta \setminus \{a\}} b\\
&elim(not[root]) = [\mathrm{child}^{-1} :: *]\\
&elim(not[leaf]) = [\mathrm{child} :: *]\\
&elim(not[\mathrm{left}]) = [\mathrm{next\_sib}^{-1} :: *]\\
&elim(not[\mathrm{right}]) = [\mathrm{next\_sib} :: *]\\
&elim(not[f \wedge f']) = not[elim(f)] \vee not[elim(f')]\\
&elim(not[r :: f]) = elim'(not[r :: *]) \vee [r :: *[not[elim(f)]]\\
&elim(not[not[f]]) = elim(f)\\
&elim(not[f \vee f']) = elim(not[f]) \wedge elim(not[f'])
\end{aligned}
$$

Eliminate Negation 3

Continue Homomorphically

$$\begin{aligned}
elim(*) &= * \\
elim(a) &= a \\
elim(root) &= root \\
elim(leaf) &= leaf \\
elim(f \wedge f') &= elim(f) \wedge elim(f') \\
elim([r :: f]) &= [r :: elim(f)] \\
elim([f \vee f']) &= elim(f) \vee elim(f')
\end{aligned}$$

Positive Filters in Monadic Datalog

The monadic Datalog program of a filter f consists of the set of all clauses for the predicates $q_{f'}$ as defined below such that f' is a subexpressions f.

Basic cases

For all filters f, f' and labels $a \in \Delta$:

$$
\begin{array}{ll}
q_*(x) :- . & \\
q_a(x) :- lab_a(x). & \\
q_\ell(x) :- \ell(x). & (\ell \in \{root, leaf, \mathrm{left}, \mathrm{right}) \\
q_{f \wedge f'}(x) :- q_f(x), q_{f'}(x). & \\
q_{f \vee f'}(x) :- q_f(x). & \\
q_{f \vee f'}(x) :- q_{f'}(x). &
\end{array}
$$

Positive Filters in Monadic Datalog

Axis

For all $r \in \{\mathrm{child}, \mathrm{next\_sib}\}$ we define the clauses:

$$
\begin{array}{l}
q_{[r::f]}(x) :- r(x,y), q_f(y). \\
q_{[r^{-1}::f]}(x) :- r(y,x), q_f(y). \\
q_{[r^*::f]}(x) :- q_f(x). \\
q_{[r^*::f]}(x) :- r(x,y), q_{[r^*::f]}(y). \\
q_{[r^{-1*}::f]}(x) :- q_f(x). \\
q_{[r^{-1*}::f]}(x) :- r(y,x), q_{[r^{-1*}::f]}(y).
\end{array}
$$

Removing child in favor of first_child

Replace

$$q(x) :- \mathrm{child}(x, y), q'(y).$$

by

$$\begin{aligned}
&q(x) :- \mathrm{first_child}(x, y), \tilde{q}(y).\\
&\tilde{q}(y) :- q'(y).\\
&\tilde{q}(y) :- \mathrm{next_sib}(y, z), \tilde{q}(z).
\end{aligned}$$

and similarly for the symetric clause:

$$q(x) :- \mathrm{child}(y, x), q'(y).$$

Results

Proposition

Any simple filter f of Core XPath 1.0 can be translated to an equivalent monadic Datalog program in time $O(|f|)$.

Proposition

Given a monadic Datalog program P and a tree t an closed monadic Datalog program for computing the least fixed point of P on t can be computed in time $O(|P||t|)$.

Theorem

For any simple filter f of Core XPath 1.0 and any tree t we can compute $eval_t(f)$ in time $O(|f||t|)$.