XML Security Views Revisited

Sławek Staworko
(joint work with Benoît Groz, Anne-Cécile Caron, Yves Roos, and Sophie Tison)

MOSTRARE Project
INRIA Lille - Nord Europe
University of Lille 1
ENS Cachan

DBPL’09
August 24, 2009
limited query languages: **downward** Reg XPath [Fan et al. VLDB’06], **vertical** Reg XPath [Rassadko SDM’07], XPath(\{\cap, \setminus\}) [Vercammen et al.]

restricted schemas and access specification: **non-recursive** DTDs [Rassadko DBSec’06], **downward closed** accessibility [Libkin and Sirangelo LPAR’08]
Overview

1. Framework:
   - Regular XPath (both vertical and horizontal axes)
   - Arbitrary DTDs
   - DTD annotations for access specification (defining views)

2. Rewriting queries over views

3. View schema construction

4. Static analysis (equivalence, comparison, . . . )
Basic Notions
XML and XPath

Regular XPath ($\mathcal{X} \text{Reg}$)

$$
\alpha ::= \text{self} \mid \downarrow \mid \uparrow \mid \Rightarrow \mid \Leftarrow
$$

$$
f ::= \text{lab()} = a \mid Q \mid \text{true} \mid \text{not } f \mid f \text{ and } f
$$

$$
Q ::= \alpha \mid [f] \mid Q/Q \mid Q \cup Q \mid Q^*
$$

$$
\alpha^+ ::= \alpha^*/\alpha
$$

$$
\alpha :: a ::= \alpha[\text{lab()} = a]
$$

$$
\alpha :: \ast ::= \alpha
$$

$$
Q[f] ::= Q/[f]
$$
DTDs and Annotations

```
DTD
projects → projects*
project → name, (stable | dev), license
stable → src, bin, doc
dev → src, doc
license → free | propr
```
DTDs and Annotations

- **projects**
  - project
    - name
    - stable
    - license
    - src
    - bin
    - doc
    - propr
  - project
    - name
    - dev
    - license
    - src
    - doc
    - free

**DTD**

- projects → projects*
- project → name, (stable | dev), license
- stable → src, bin, doc
- dev → src, doc
- license → free | propr

**Annotation**

- $A(\text{stable}) = \text{false}$
- $A(\text{dev}) = \text{false}$
- $A(\text{doc}) = \text{true}$
- $A(\text{src}) = [\uparrow::*/\downarrow::\text{license}/\downarrow::\text{free}]$

- hide status of project (including binaries)
- source visible only if under free license
**DTDs and Annotations**

**Accessibility:**
- root always accessible
- if $A$ defined for the node label, then evaluate the filter
- otherwise, accessibility inherited from the parent

**View:** $A(t) = \text{tree obtained from accessible nodes only.}$

**Annotation**
- $A(\text{stable}) = \text{false}$
- $A(\text{dev}) = \text{false}$
- $A(\text{doc}) = \text{true}$
- $A(\text{src}) = [\uparrow::*:\Rightarrow::\text{license}/\downarrow::\text{free}]$

- hide status of project (including binaries)
- source visible only if under free license
DTDs and Annotations

Accessibility:
- root always accessible
- if \( A \) defined for the node label, then evaluate the filter
- otherwise, accessibility inherited from the parent

View: \( A(t) = \text{tree obtained from accessible nodes only.} \)

Annotation
\[
\begin{align*}
A(\text{stable}) &= \text{false} \\
A(\text{dev}) &= \text{false} \\
A(\text{doc}) &= \text{true} \\
A(\text{src}) &= [\uparrow:*/\Rightarrow::\text{license}/\downarrow::\text{free}] 
\end{align*}
\]
Query Rewriting
Lemma 1
For any annotation $A$ there exists a filter expression $f_{acc}$ such that

\[ a \text{ node } n \text{ of a tree } t \text{ is accessible w.r.t. } A \iff (t, n) \models f_{acc} \]

Proof
First check whether $A$ defines a test for the current node

\[ f_{dom} := \bigvee_{a \in \text{dom}(A)} \text{lab}() = a \]

if so, then evaluate it

\[ f_{eval} := \bigvee_{a \in \text{dom}(A)} (\text{lab}() = a \text{ and } A(a)) \]

otherwise, go up until you find such a node (or you reach the root)

\[ f_{acc} := ([\text{not } f_{dom}] / \uparrow)^* / [f_{eval} \text{ or not(\uparrow)}] . \]
Query Rewriting: Vertical axes

\[
\text{Rewrite}(\downarrow) := [f_{\text{acc}}]/\downarrow/(\lnot f_{\text{acc}})/\downarrow)^{*}/f_{\text{acc}}
\]
Query Rewriting: Horizontal axes

Rewrite(⇒): combine 3 tricks

Theorem

Regular XPath is closed under rewriting over XML views.
The size of the rewritten query is $O(|A| \ast |Q|)$, where $Q$ is the original query.
Constructing View Schema
Deriving view schema

**Annotation**
- $A(\text{stable}) = \text{false}$
- $A(\text{dev}) = \text{false}$
- $A(\text{doc}) = \text{true}$

**DTD**

```xml
projects → projects^*

project → name, (stable | dev), license

stable → src, bin, doc

dev → src, doc

license → free | propr
```
Deriving view schema

**DTD**

\[
\begin{align*}
\text{projects} & \rightarrow \text{projects}^* \\
\text{project} & \rightarrow \text{name}, (\text{stable} \mid \text{dev}), \text{license} \\
\text{stable} & \rightarrow \text{src}, \text{bin}, \text{doc} \\
\text{dev} & \rightarrow \text{src}, \text{doc} \\
\text{license} & \rightarrow \text{free} \mid \text{propr}
\end{align*}
\]

**Annotation**

\[A(\text{stable}) = \text{false} \]
\[A(\text{dev}) = \text{false} \]
\[A(\text{doc}) = \text{true} \]
Deriving view schema

**DTD**

- `projects → projects*`
- `project → name, (stable | dev), license`
- `stable → src, bin, doc`
- `dev → src, doc`
- `license → free | propr`

**View DTD**

- `projects → projects*`
- `project → name, doc, license`
- `license → free | propr`

**Annotation**

- \( A(\text{stable}) = \text{false} \)
- \( A(\text{dev}) = \text{false} \)
- \( A(\text{doc}) = \text{true} \)
One problem: Size

**DTD (annotated)**
- \( r \rightarrow a_n \)
- \( a_n \rightarrow a_{n-1}, a_{n-1} \)
- \( a_{n-1} \rightarrow a_{n-2}, a_{n-2} \)
- ...
- \( a_1 \rightarrow \) empty
- \( A(a_n) = \) false
- \( A(a_1) = \) true

**View DTD**
- \( r \rightarrow a_1, \ldots, a_1 \)
- \( a_1 \rightarrow \) empty
- \( 2^n \)

**Observation**
The view DTD may be of *exponential* size!
And another one: Regularity

**Observation**

The view schema needs not be regular (in particular may not have a DTD)

**Proposition**

It is **undecidable** to test if the view schema can be captured with a DTD.
Approximation: Optimality criterion

Definition (Indistinguishability)
Two sets of trees \( L_1 \) and \( L_2 \) are *indistinguishable* by a class of queries \( C \) iff

\[
\forall Q \in C. \left[ (\exists t_1 \in L_1. t_1 \models Q) \iff (\exists t_2 \in L_2. t_2 \models Q) \right].
\]

Approximation
A DTD \( D^* \) is a **good approximation** of the view schema of \( D \) and \( A \) if \( L(D^*) \) and \( \{A(t) \mid t \in L(D)\} \) are indistinguishable by a relatively large class of queries.
Three approximations

**DTD (annotated)**
- $r \rightarrow c$
- $c \rightarrow (a, c?, b)$
- $A(c) = \text{false}$
- $A(a) = \text{true}$
- $A(b) = \text{true}$

![Diagram of DTD](image)

<table>
<thead>
<tr>
<th>Parikh</th>
<th>Subword</th>
<th>Subset</th>
</tr>
</thead>
<tbody>
<tr>
<td>$r \rightarrow (a, b)^*$</td>
<td>$r \rightarrow a^<em>, b^</em>$</td>
<td>$r \rightarrow (a \mid b)^*$</td>
</tr>
<tr>
<td>$\chi \text{Reg}(\downarrow, \uparrow, [], \text{not})$</td>
<td>$\chi \text{Reg}(\downarrow, \uparrow, \Rightarrow^+, \Leftarrow^+, [])$</td>
<td>$\chi \text{Reg}(\downarrow)$</td>
</tr>
</tbody>
</table>
Further results

- No approximation
- Parikh approximation
- Subword approximation
- Lower exponential bound
- Subset approximation

\( \langle \psi, \|, [], not \rangle \)
\( \langle \psi, \|, [], \rangle \)
\( \langle \psi, \|, [\], \rangle \)
\( \langle \psi, \|, [\|, \rangle \)
\( \langle \psi, \|, [\|, [\], not \rangle \)
\( \langle \psi, \|, [\|, [+], \rangle \)
\( \langle \psi, \|, [\|, [+], [\], not \rangle \)
Elements of Static Analysis
Node-based comparison

Equivalence

\[ A_1 \equiv^D A_2 \iff \forall t \in L(D). \text{Nodes}(A_1(t)) = \text{Nodes}(A_2(t)) \]

Node-based restriction

\[ A_1 \triangleleft^D_{NB} A_2 \iff \forall t \in L(D). \text{Nodes}(A_1(t)) \subseteq \text{Nodes}(A_2(t)) \]

Theorem

Testing equivalence and node-based restriction is \text{EXPTIME}-complete.
Potential information leaks

Original annotation
\[ A_1(\text{stable}) = \text{false} \]
\[ A_1(\text{dev}) = \text{false} \]
\[ A_1(\text{doc}) = \text{true} \]
\[ A_1(\text{src}) = [\uparrow:\*:\Rightarrow:\text{license}/\downarrow:\text{free}] \]

With the new annotation, the query
\[ \rightarrow:\text{project} \not\rightarrow:\text{src} \text{ and } \rightarrow:\text{license} \text{ identifies a subset of projects under development which could not be selected before!} \]
Potential information leaks

Original annotation
\[ A_1(\text{stable}) = \text{false} \]
\[ A_1(\text{dev}) = \text{false} \]
\[ A_1(\text{doc}) = \text{true} \]
\[ A_1(\text{src}) = [\uparrow::\star\mapsto::\text{license}/\downarrow::\text{free}] \]
Potential information leaks

Original annotation
\[ A_1(\text{stable}) = \text{false} \]
\[ A_1(\text{dev}) = \text{false} \]
\[ A_1(\text{doc}) = \text{true} \]
\[ A_1(\text{src}) = [\uparrow::/* / \Rightarrow::\text{license}/ \downarrow::\text{free}] \]

New annotation (hide sources of projects under development)
\[ A_2(\text{stable}) = \text{false} \]
\[ A_2(\text{dev}) = \text{false} \]
\[ A_2(\text{doc}) = \text{true} \]
\[ A(\text{src}) = [\uparrow::\text{stable}/ \Rightarrow::\text{license}/ \downarrow::\text{free}] \]
Potential information leaks

Original annotation
\[ A_1(stable) = false \]
\[ A_1(dev) = false \]
\[ A_1(doc) = true \]
\[ A_1(src) = [\uparrow::*/\Rightarrow::license/\downarrow::free] \]

New annotation (hide sources of projects under development)
\[ A_2(stable) = false \]
\[ A_2(dev) = false \]
\[ A_2(doc) = true \]
\[ A(src) = [\uparrow::stable/\Rightarrow::license/
\downarrow::free] \]

With the new annotation, the query
\[ \downarrow::project[\text{not}(\downarrow::src) \text{ and } \downarrow::license/\downarrow::free] \]
identifies a subset of projects under development which could not be selected before!
Query-based comparison

Identify accessible information

\[ Public(D, A) = \{ Q \mid \exists Q'. Rewrite(Q', A) \equiv^D Q \} \]

Definition (Query-based restriction)

\[ A_1 \leq^D_{QB} A_2 \iff Public(D, A_1) \subseteq Public(D, A_2) \]

Negative results

Testing query-based restriction is **undecidable**.

Positive results

Testing query-based restriction for non-recursive DTDs is in EXPTIME (and is PSPACE-hard).
Future work

- Implementation (Conditional XPath, XQuery engine, ...)
- Richer schema formalisms for approximation (EDTD, XML Schema)
- Other approximation criteria
- Further study of static analysis problems (interval bounded DTDs)