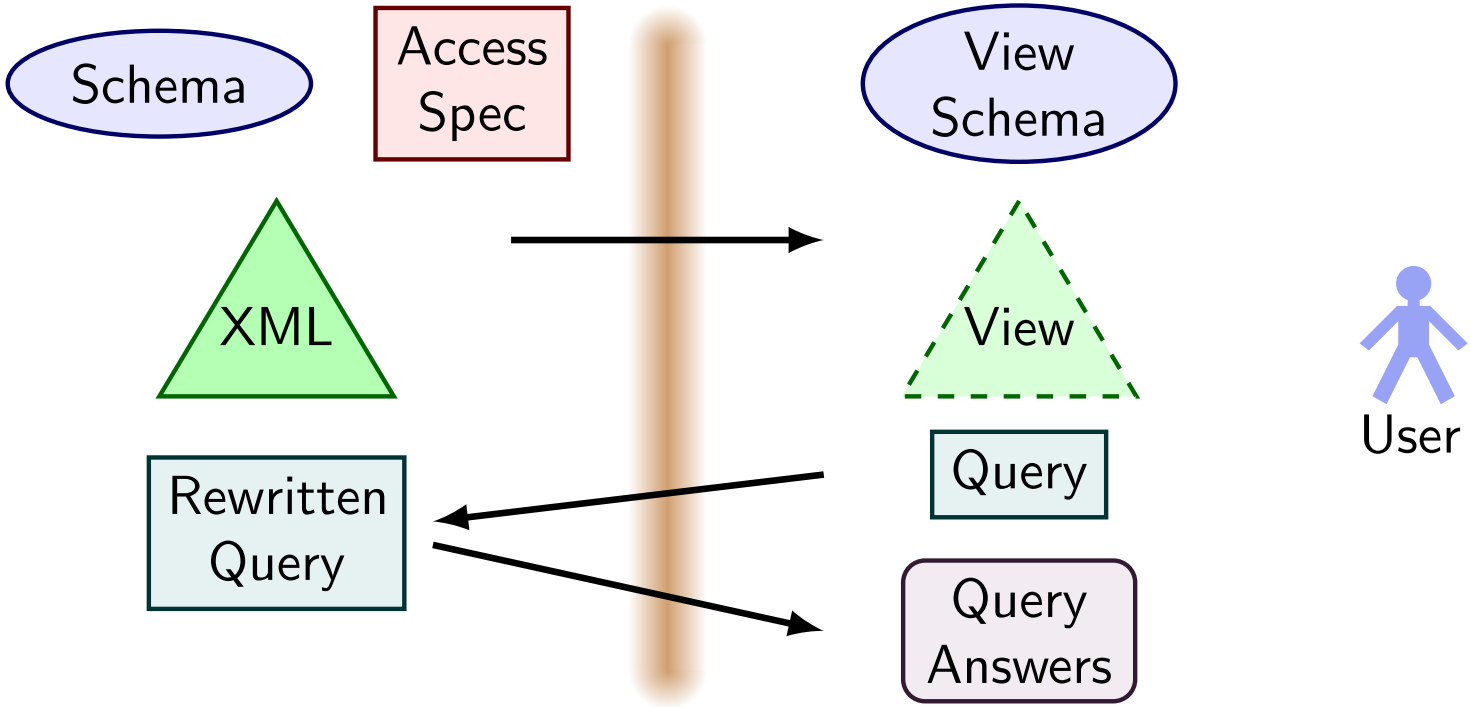
XML Security Views Revisited

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

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

DBPL'09
August 24, 2009

View Based XML Security Framework



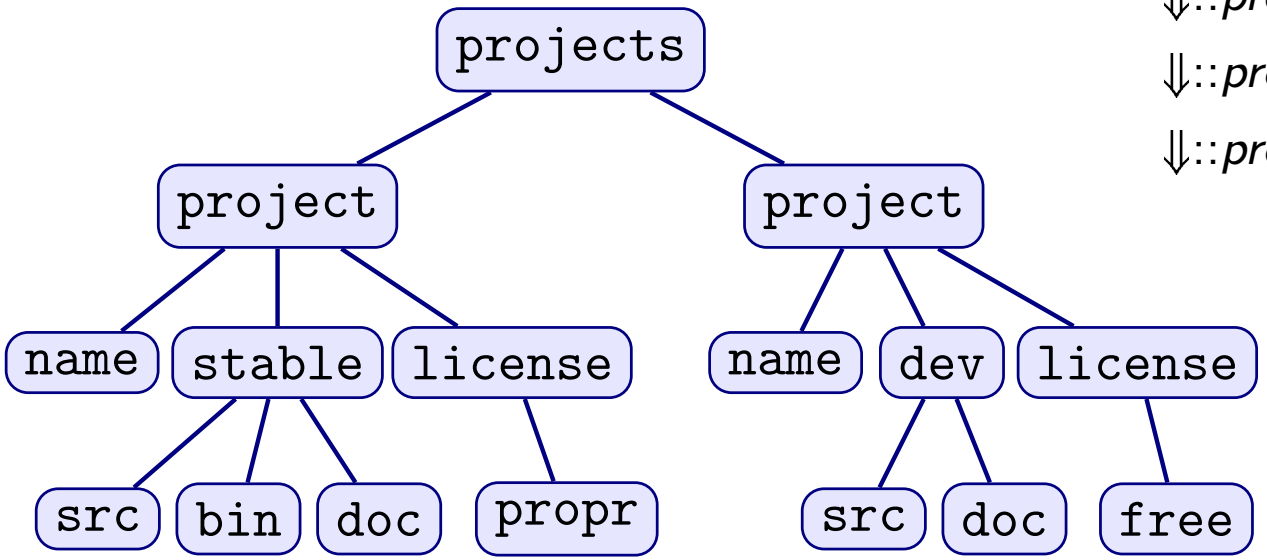
- limited query languages: **downward** Reg XPath [Fan et al. VLDB'06], **vertical** Reg XPath [Rassadko SDM'07], XPath$(\{\cap, \setminus\})$ [Vercammen et al.]
- restricted schemas and access specification: **non-recursive** DTDs [Rassadko DBSec'06], **downward closed** accessibility [Libkin and Sirangelo 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



$\Downarrow::project/\Downarrow::name$

$\Downarrow::project[\Downarrow::stable]/\Downarrow::name$

$\Downarrow::project/\Downarrow::*/\Downarrow::src$

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

$$\alpha ::\equiv \mathsf{self} \mid \Downarrow \mid \Uparrow \mid \Rightarrow \mid \Leftarrow$$

$$f ::\equiv \mathsf{lab}() = a \mid Q \mid \mathsf{true} \mid \mathsf{not}\, f \mid f \;\mathsf{and}\; f$$

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

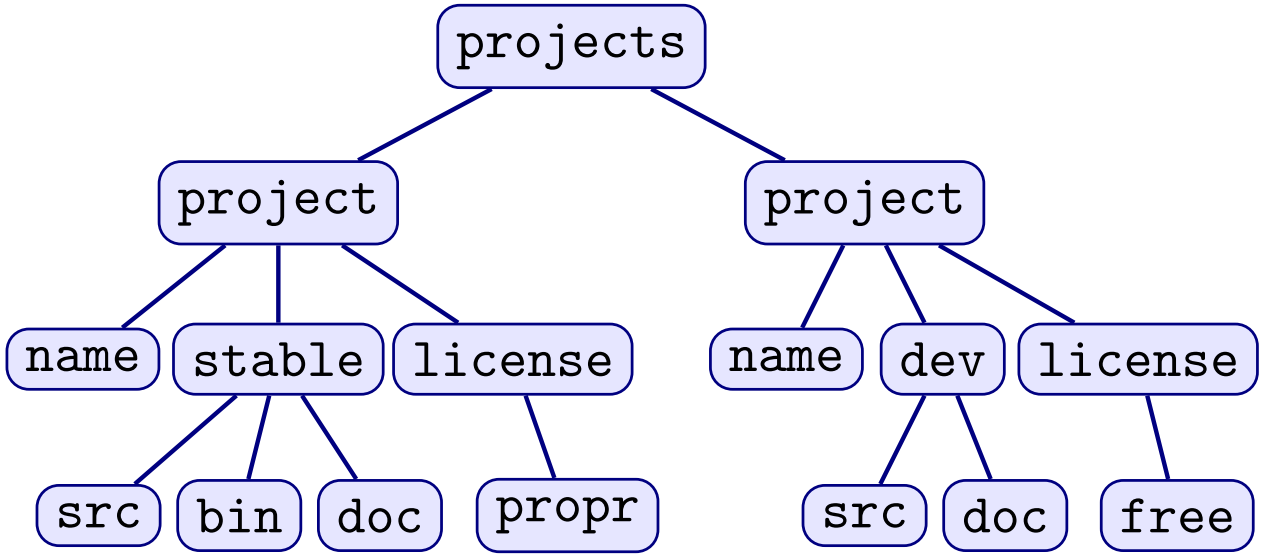
$$\alpha^+ \coloneqq \alpha^*/\alpha$$

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

$$\alpha::* \coloneqq \alpha$$

$$Q[f] \coloneqq Q/[f]$$

DTDs and Annotations



DTD

$projects \rightarrow projects^*$

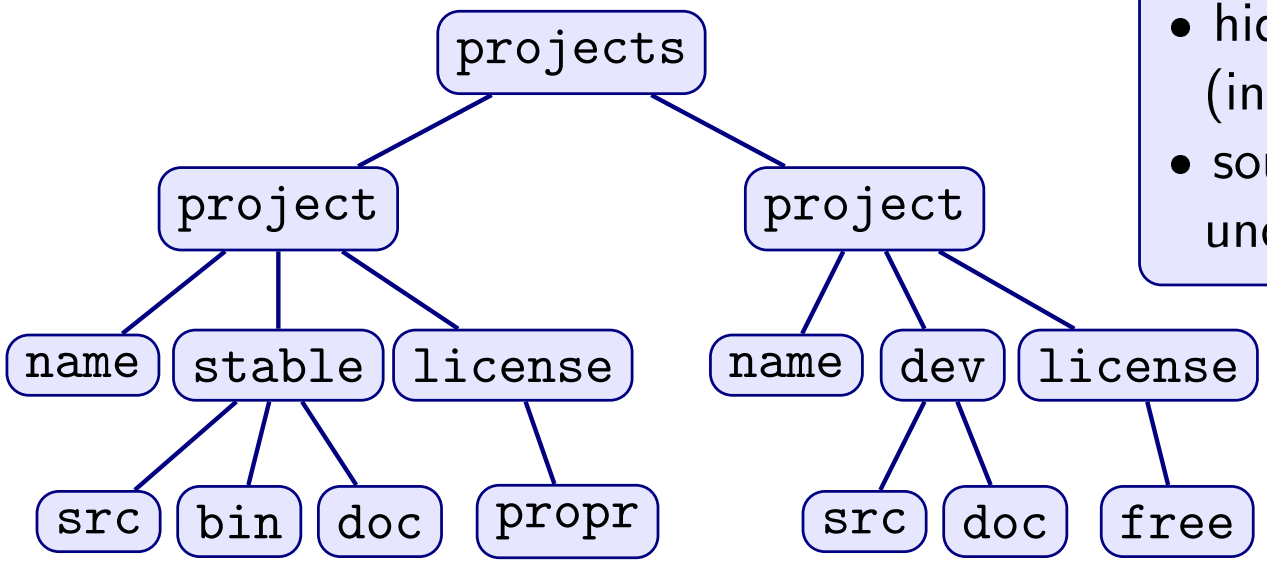
$project \rightarrow name, (stable \mid dev), license$

$stable \rightarrow src, bin, doc$

$dev \rightarrow src, doc$

$license \rightarrow free \mid propr$

DTDs and Annotations



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

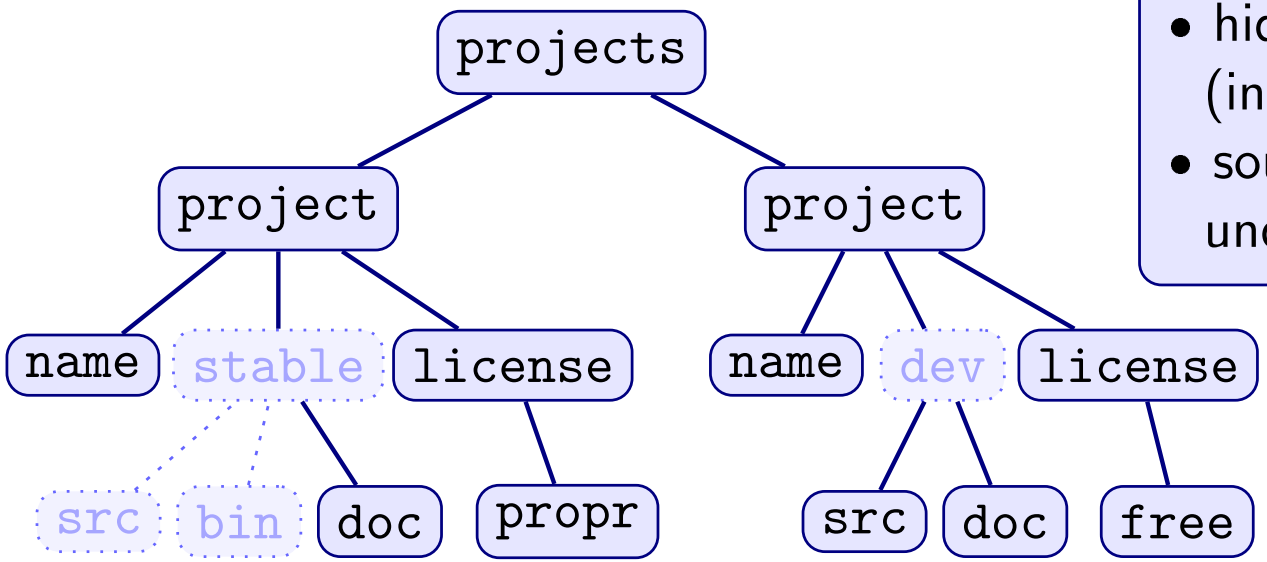
DTD

$projects \rightarrow projects^*$
$project \rightarrow name, (stable \mid dev), license$
$stable \rightarrow src, bin, doc$
$dev \rightarrow src, doc$
$license \rightarrow free \mid propr$

Annotation

$A(stable) = \text{false}$
$A(dev) = \text{false}$
$A(doc) = \text{true}$
$A(src) = [\Uparrow::*/\Rightarrow::license/\Downarrow::free]$

DTDs and Annotations



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

Accessibility:

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

View: A(t) = tree obtained from accessible nodes only.

Annotation

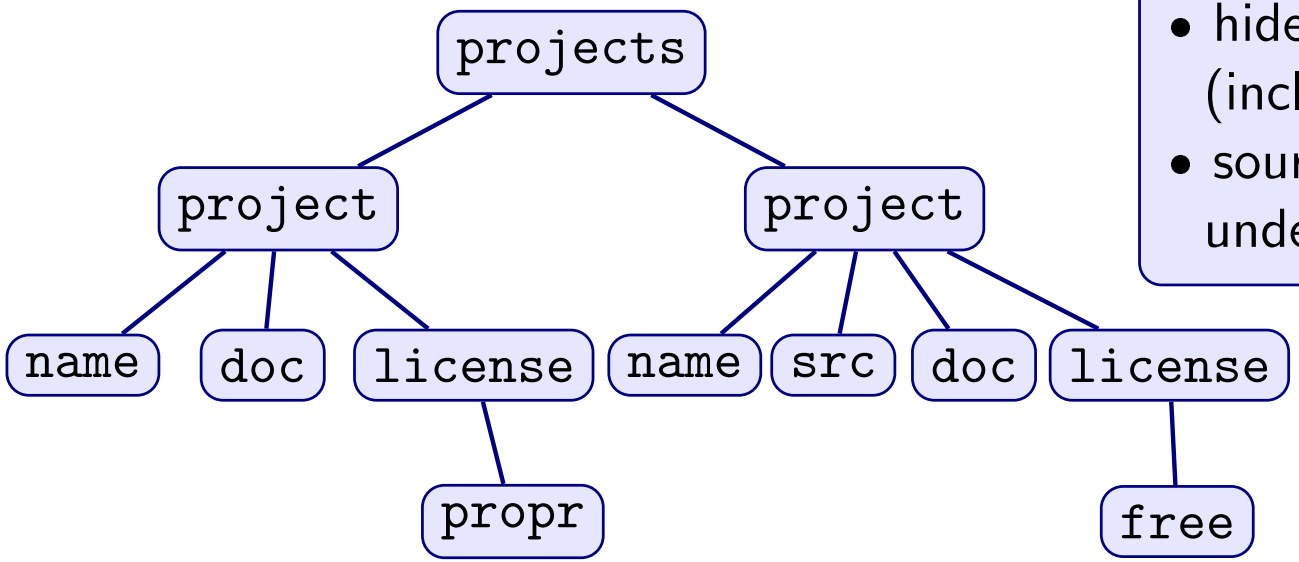
$A(stable) = \text{false}$

$A(dev) = \text{false}$

$A(doc) = \text{true}$

$A(src) = [\Uparrow::*/\Rightarrow::license/\Downarrow::free]$

DTDs and Annotations



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

Accessibility:

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

View: A(t) = tree obtained from accessible nodes only.

Annotation

$A(stable) = \text{false}$

$A(dev) = \text{false}$

$A(doc) = \text{true}$

$A(src) = [\Uparrow::*/\Rightarrow::license/\Downarrow::free]$

Query Rewriting

Query Rewriting

Lemma 1

For any annotation A there exists a filter expression f_{acc} such that

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

Proof

First check weather A defines a test for the current node

$$f_{dom} \coloneqq \bigvee_{a \in dom(A)} \text{lab}() = a$$

if so, then evaluate it

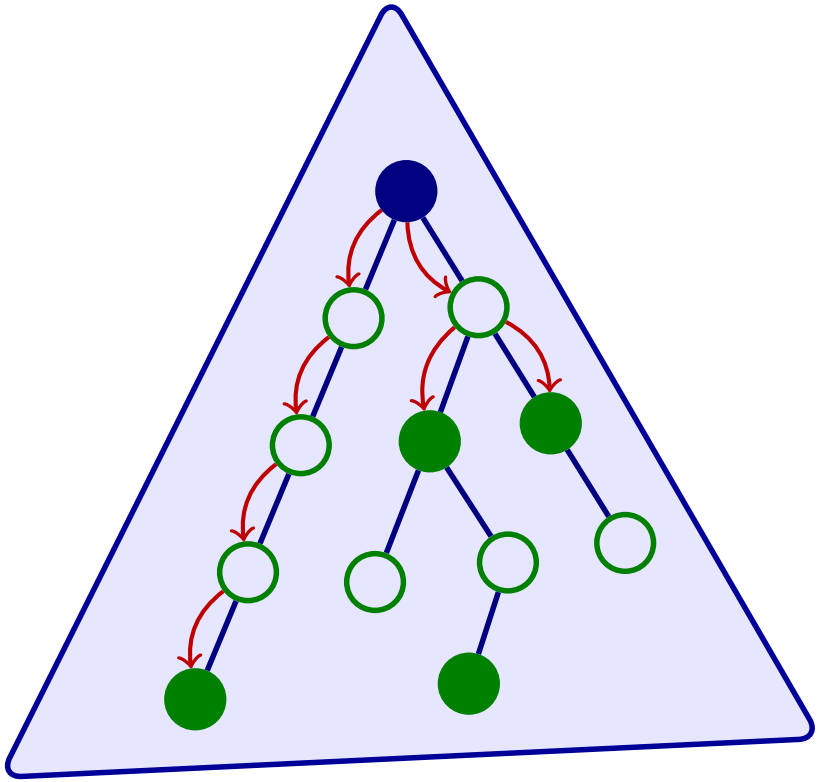
$$f_{eval} \coloneqq \bigvee_{a \in dom(A)} (\text{lab}() = a \text{ and } A(a))$$

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

$$f_{\text{acc}} \coloneqq ([\text{not } f_{dom}]/\Uparrow)^*/[f_{eval} \text{ or not}(\Uparrow)].$$

□

Query Rewriting: Vertical axes



current node

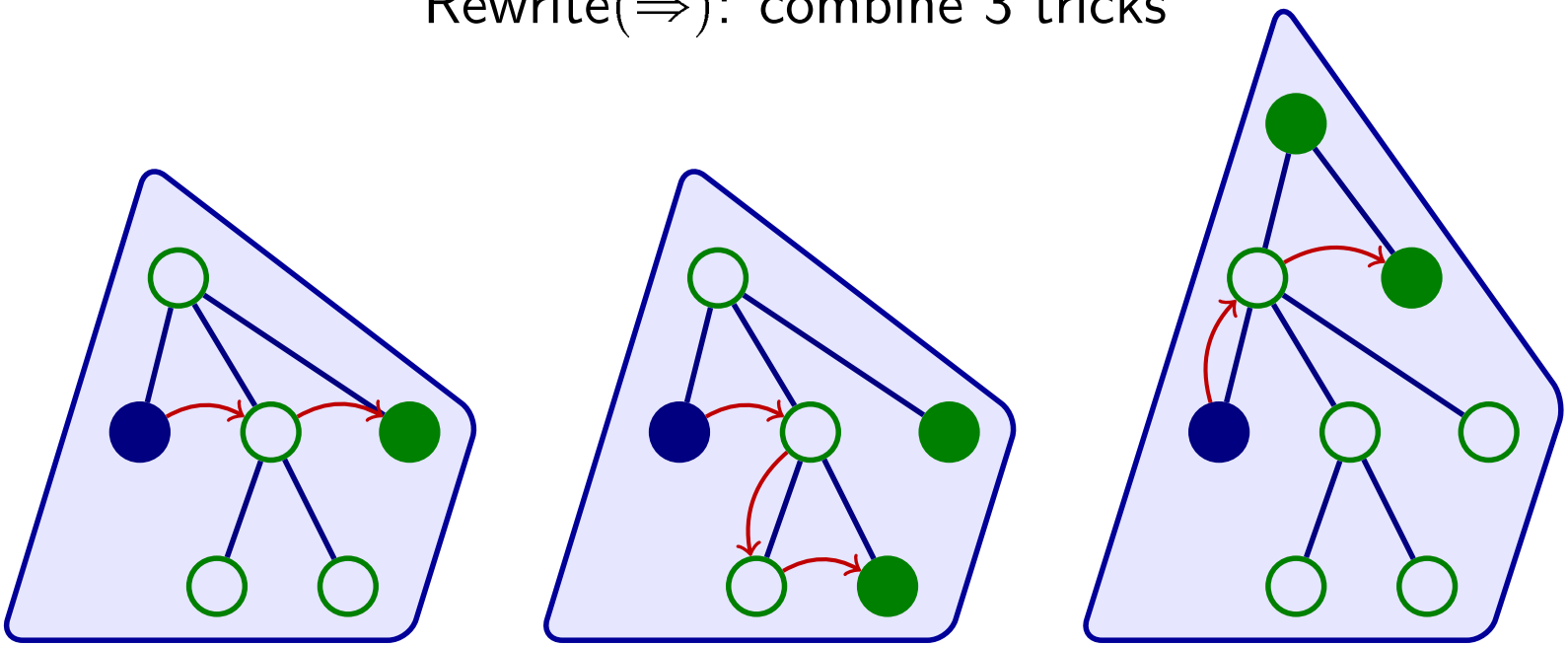
accessible node (f_{acc})

inaccessible node (not f_{acc})

$$\text{Rewrite}(\Downarrow) := [f_{\text{acc}}]/\Downarrow/\left([\text{not } f_{\text{acc}}]/\Downarrow\right)^*/[f_{\text{acc}}]$$

Query Rewriting: Horizontal axes

Rewrite($\Rightarrow$): combine 3 tricks

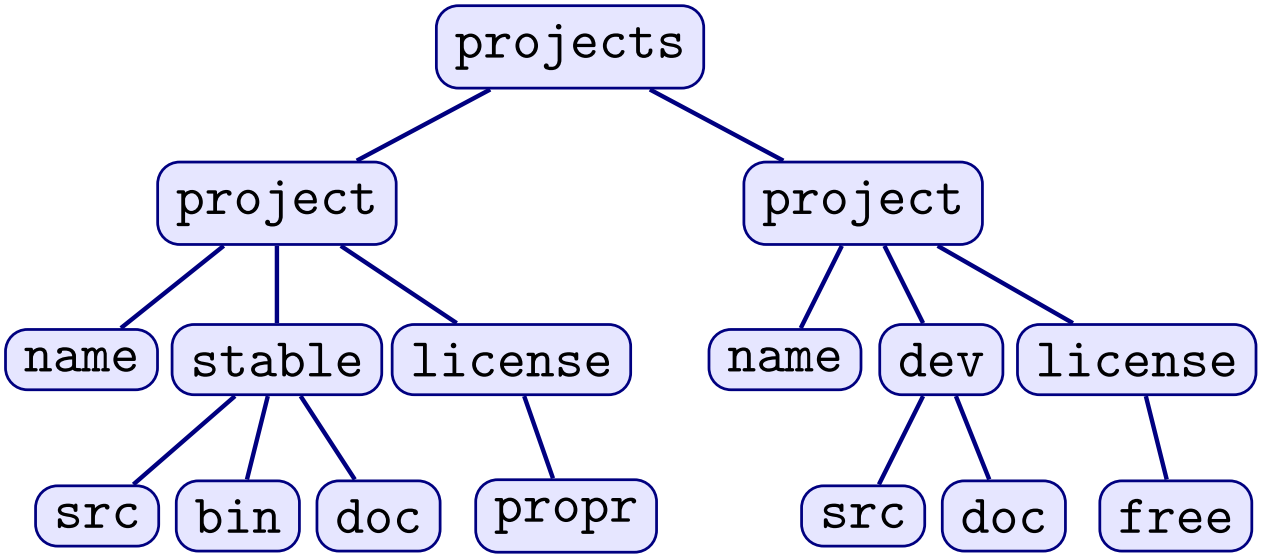


Theorem

Regular XPath is closed under rewriting over XML views.
The size of the rewritten query is $O(|A| * |Q|)$, where Q is the original query.

Constructing View Schema

Deriving view schema



Annotation

$A(stable) = \text{false}$

$A(dev) = \text{false}$

$A(doc) = \text{true}$

DTD

$projects \rightarrow projects^*$

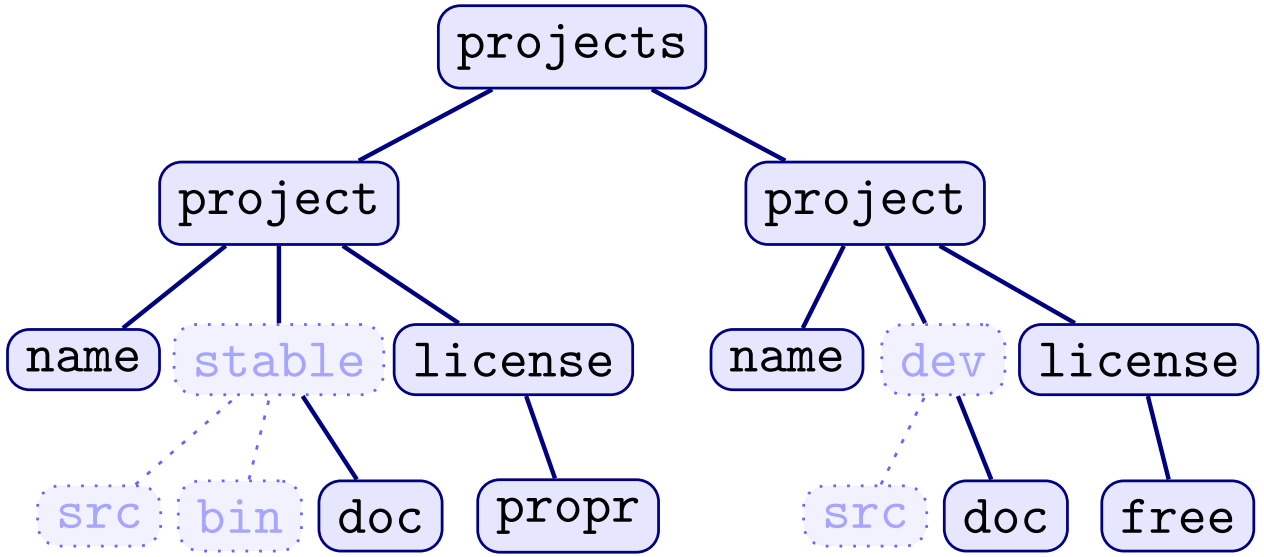
$project \rightarrow name, (stable \mid dev), license$

$stable \rightarrow src, bin, doc$

$dev \rightarrow src, doc$

$license \rightarrow free \mid propr$

Deriving view schema



Annotation

$A(stable) = \text{false}$

$A(dev) = \text{false}$

$A(doc) = \text{true}$

DTD

$projects \rightarrow projects^*$

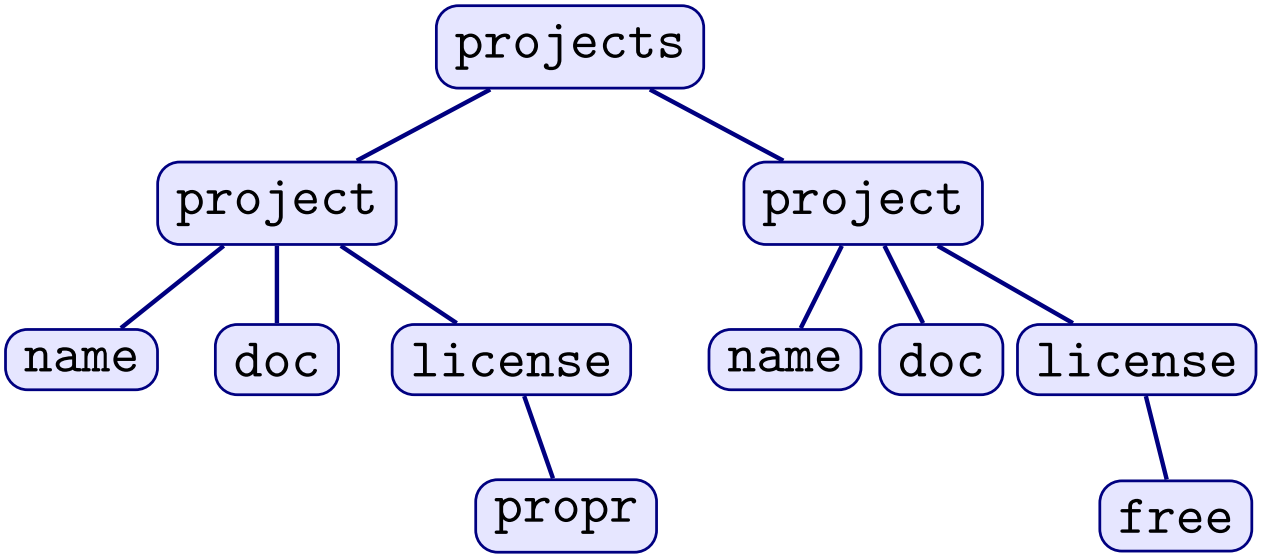
$project \rightarrow name, (stable \mid dev), license$

$stable \rightarrow src, bin, doc$

$dev \rightarrow src, doc$

$license \rightarrow free \mid propr$

Deriving view schema



Annotation

$A(stable) = \text{false}$

$A(dev) = \text{false}$

$A(doc) = \text{true}$

DTD

$projects \rightarrow projects^*$

$project \rightarrow name, (stable \mid dev), license$

$stable \rightarrow src, bin, doc$

$dev \rightarrow src, doc$

$license \rightarrow free \mid propr$

View DTD

$projects \rightarrow projects^*$

$project \rightarrow name, doc, license$

$license \rightarrow free \mid propr$

One problem: Size

DTD (annotated)

$r \to a_n$

$a_n \to a_{n-1}, a_{n-1}$

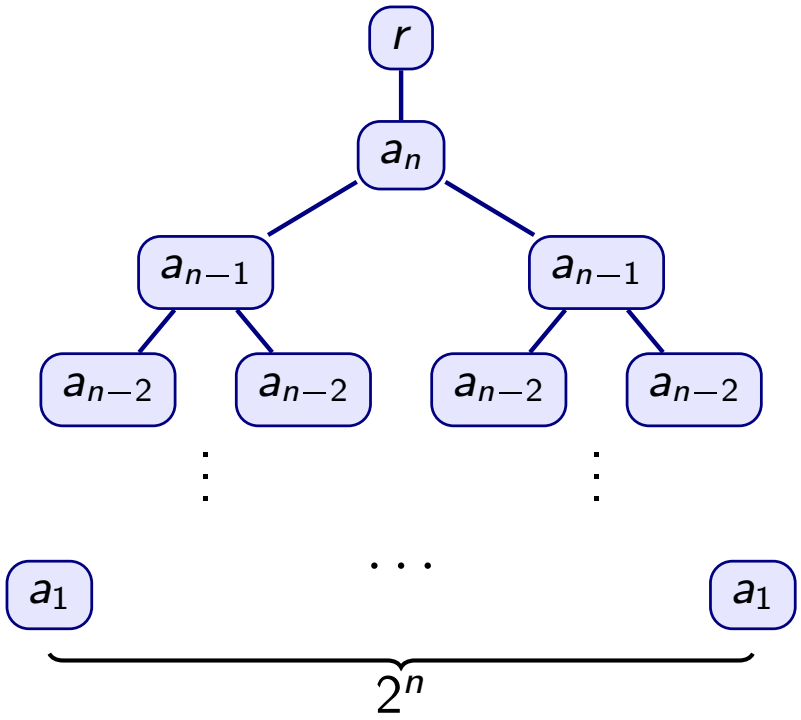
$a_{n-1} \to a_{n-2}, a_{n-2}$

$\ldots$

$a_1 \to \text{empty}$

$A(a_n) = \text{false}$

$A(a_1) = \text{true}$



View DTD

$r \to \underbrace{a_1, \ldots, a_1}_{2^n}$

$a_1 \to \text{empty}$

Observation

The view DTD may be of *exponential* size!

And another one: Regularity

DTD (annotated)

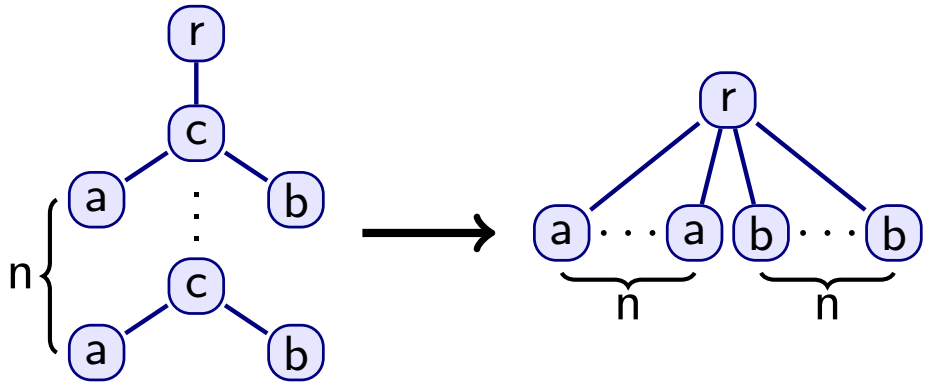
$r \to c$

$c \to (a, c?, b)$

$A(c) = \text{false}$

$A(a) = \text{true}$

$A(b) = \text{true}$



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 $\mathcal{C}$ iff

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

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)

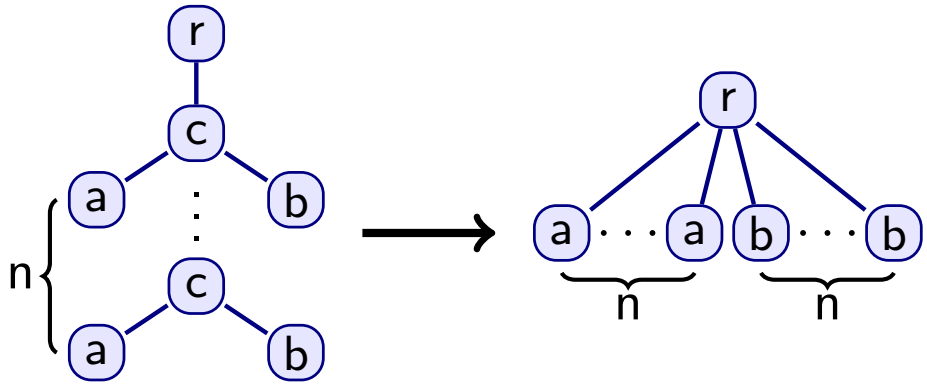
$r \to c$

$c \to (a, c?, b)$

$A(c) = \text{false}$

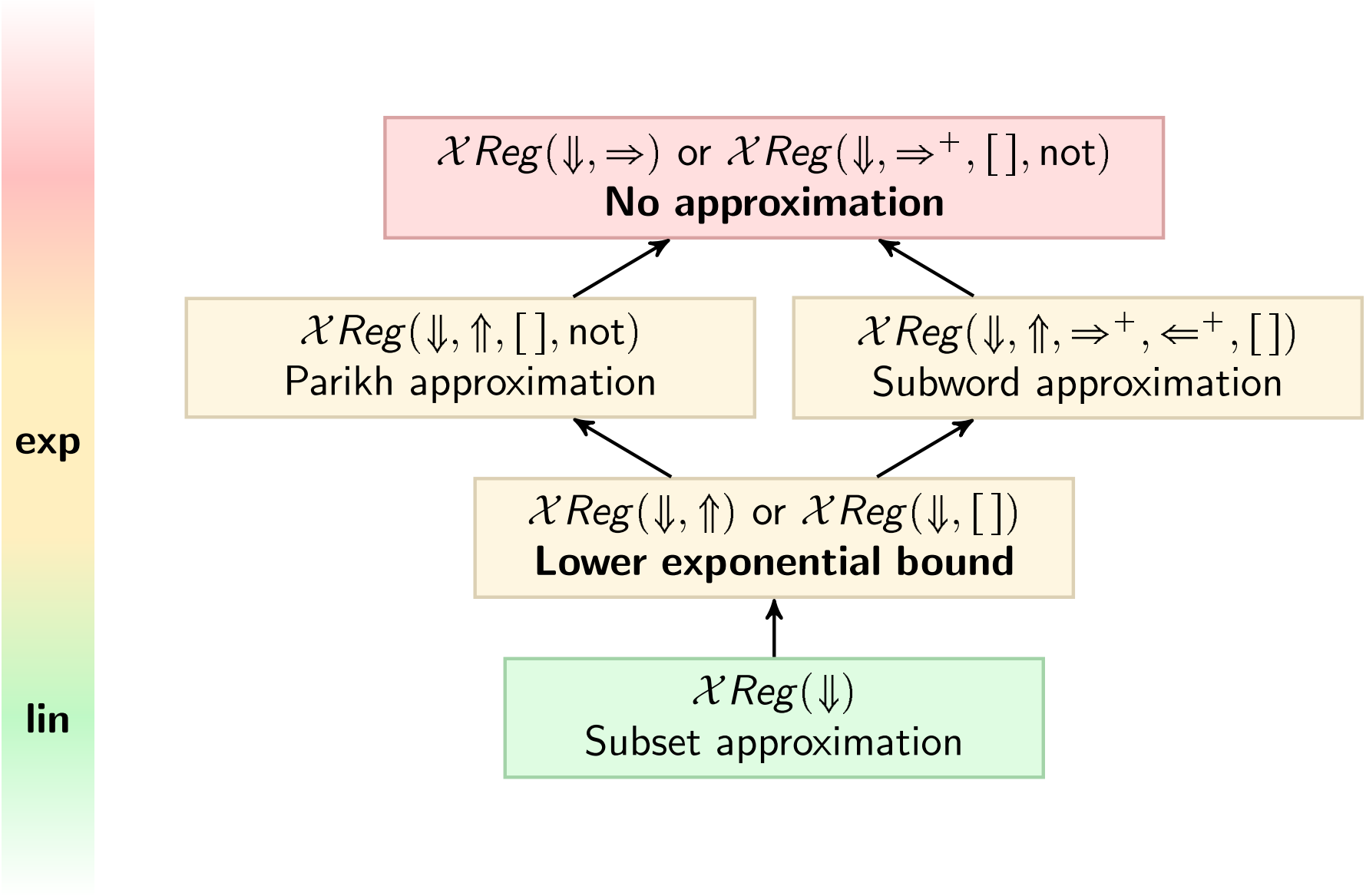
$A(a) = \text{true}$

$A(b) = \text{true}$



Parikh	**Subword**	**Subset**
$r \to (a, b)^*$	$r \to a^*, b^*$	$r \to (a \mid b)^*$
$\mathcal{X}Reg(\Downarrow, \Uparrow, [\,], \text{not})$	$\mathcal{X}Reg(\Downarrow, \Uparrow, \Rightarrow^+, \Leftarrow^+, [\,])$	$\mathcal{X}Reg(\Downarrow)$

Further results



Elements of Static Analysis

Node-based comparison

Equivalence

$$A_1 \equiv^D A_2 \iff \forall t \in L(D).\, Nodes(A_1(t)) = Nodes(A_2(t))$$

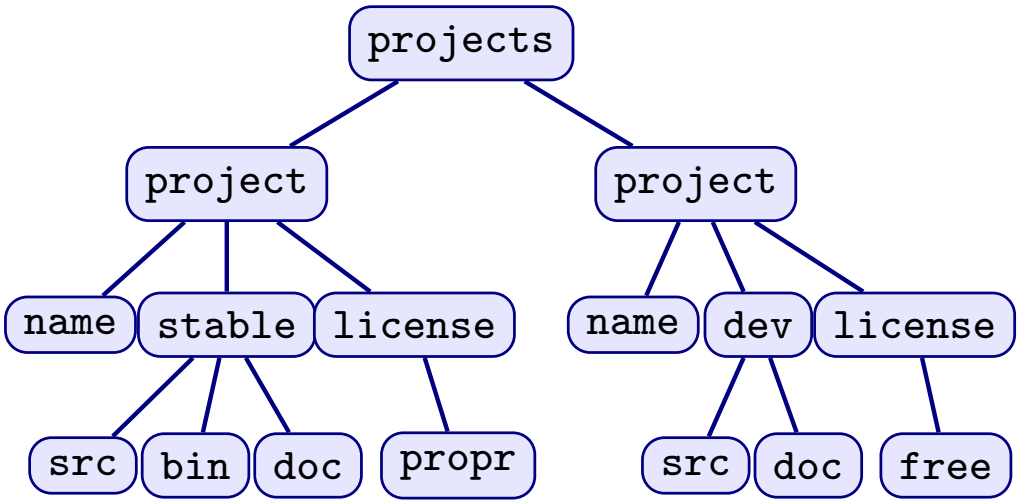
Node-based restriction

$$A_1 \preccurlyeq^D_{NB} A_2 \iff \forall t \in L(D).\, Nodes(A_1(t)) \subseteq Nodes(A_2(t))$$

Theorem

Testing equivalence and node-based restriction is EXPTIME-complete.

Potential information leaks



Original annotation

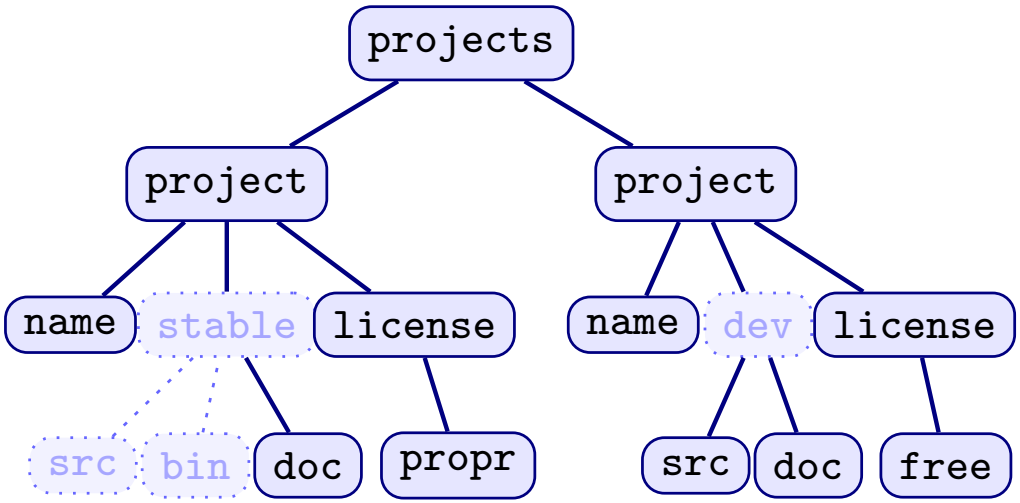
$A_1(stable) = \text{false}$

$A_1(dev) = \text{false}$

$A_1(doc) = \text{true}$

$A_1(src) = [\Uparrow::*/\Rightarrow::license/\Downarrow::free]$

Potential information leaks



Original annotation

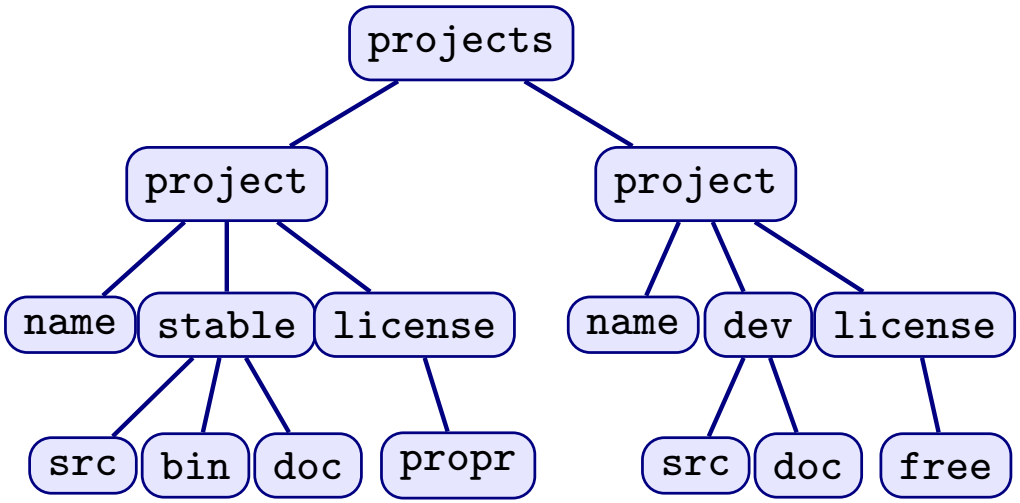
$A_1(stable) = \text{false}$

$A_1(dev) = \text{false}$

$A_1(doc) = \text{true}$

$A_1(src) = [\Uparrow::*/\Rightarrow::license/\Downarrow::free]$

Potential information leaks



Original annotation

$A_1(stable) = \text{false}$

$A_1(dev) = \text{false}$

$A_1(doc) = \text{true}$

$A_1(src) = [\Uparrow::*/\Rightarrow::license/\Downarrow::free]$

New annotation (hide sources of projects under development)

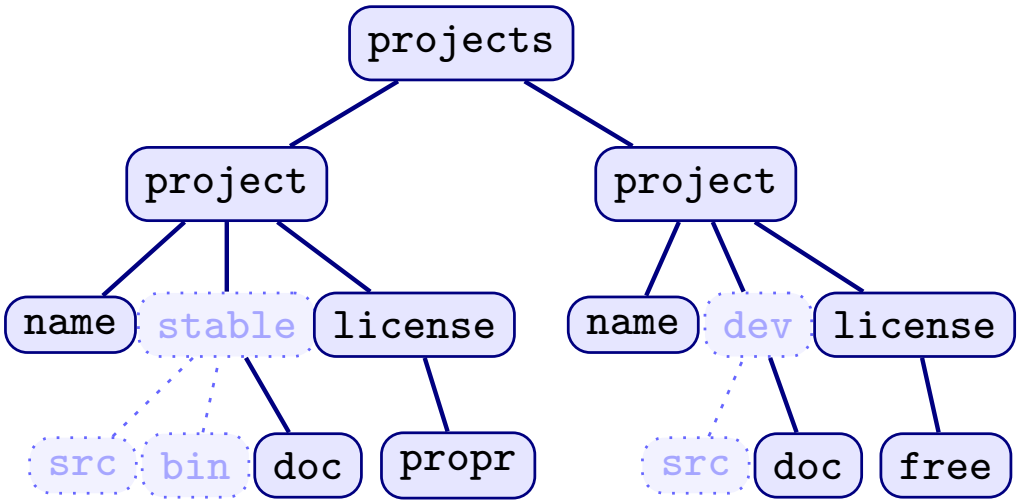
$A_2(stable) = \text{false}$

$A_2(dev) = \text{false}$

$A_2(doc) = \text{true}$

$A(src) = [\Uparrow::stable/\Rightarrow::license/\Downarrow::free]$

Potential information leaks



Original annotation

$A_1(stable) = \text{false}$

$A_1(dev) = \text{false}$

$A_1(doc) = \text{true}$

$A_1(src) = [\Uparrow::*/\Rightarrow::license/\Downarrow::free]$

New annotation (hide sources of projects under development)

$A_2(stable) = \text{false}$

$A_2(dev) = \text{false}$

$A_2(doc) = \text{true}$

$A(src) = [\Uparrow::stable/\Rightarrow::license/\Downarrow::free]$

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)

$$A_1 \preccurlyeq^{D}_{QB} A_2 \iff Public(D, A_1) \subseteq Public(D, A_2)$$

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)