Learning Twig and Path Queries

Sławek Staworko[1] Piotr Wieczorek[2]

[1]Mostrare Project
INRIA Lille – Nord Europe
University of Lille

[2]Institute of Computer Science
University of Wrocław

March 27, 2012
ICDT 2012

Motivation

The challenge

- XML is text based for easy and direct access to its contents
- XPath and XQuery might be too formal/difficult for unknowledgeable users

XML query inference

- User provides an XML document with annotations (selected nodes, etc.)
- A learning algorithm returns a query addressing the user's needs

Learning: a popular and interesting topic

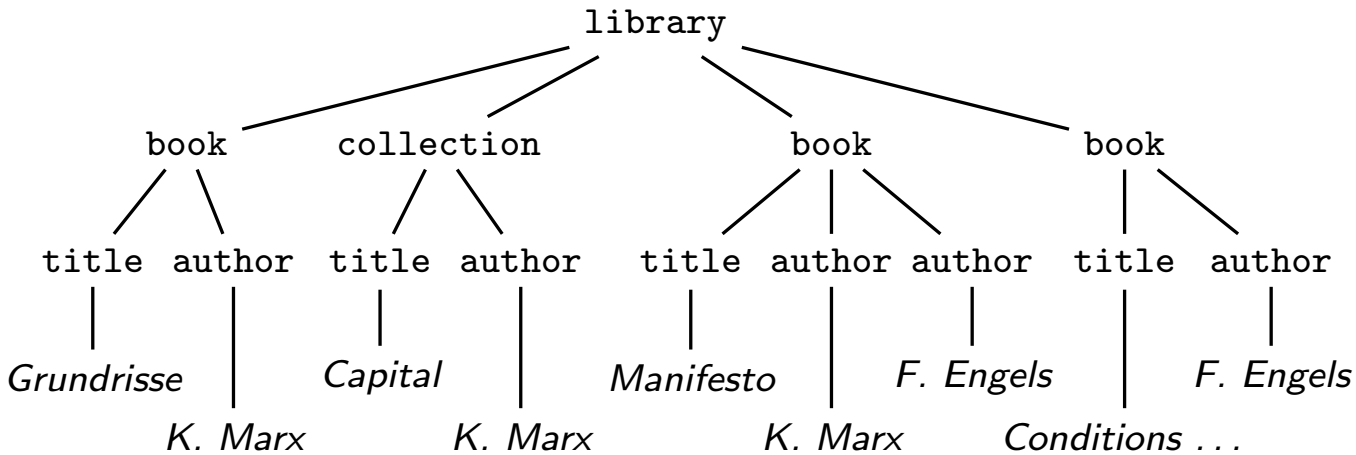
- Languages of words [Gold '67], Word patterns [Anguin '79], Regular languages [Anguin '87, Oncina and Garcia '94, de la Higuera '97, ...]
- Schemas for XML [Bex et al. '10, ...]
- Query Automata [Carme et al. '07] and tree transducers [Lemay et al. '10]
- XQuery [Morishima et al. '04] and web extraction patterns [Raeymaekers et al. '08]

Overview

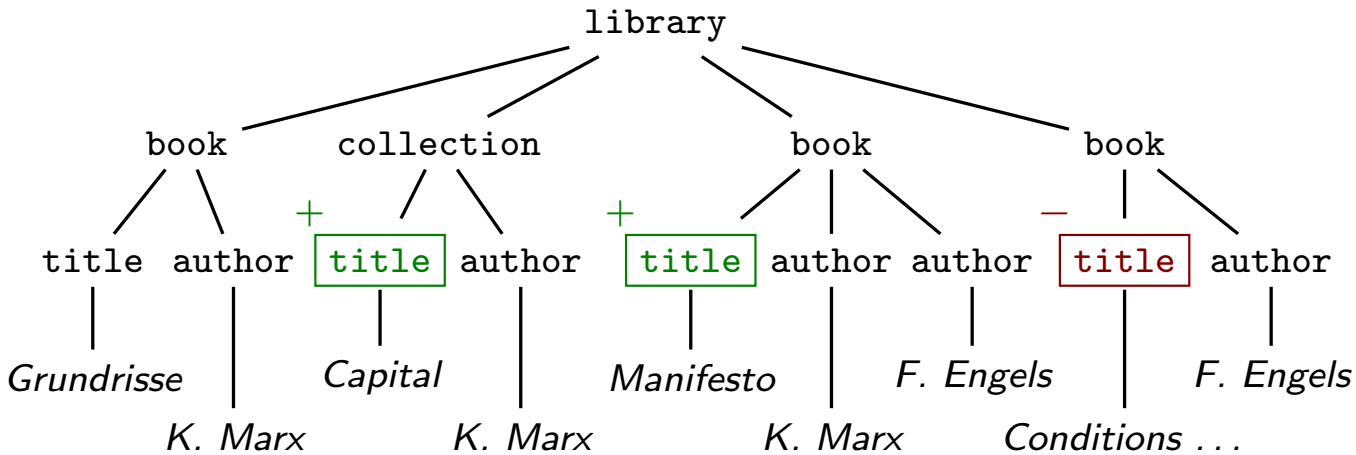
1. Preliminaries
2. What learning queries means?
3. Negative results
4. Learning algorithms

Basic notions

XML, Examples, and Unary Twig queries (tree patterns)



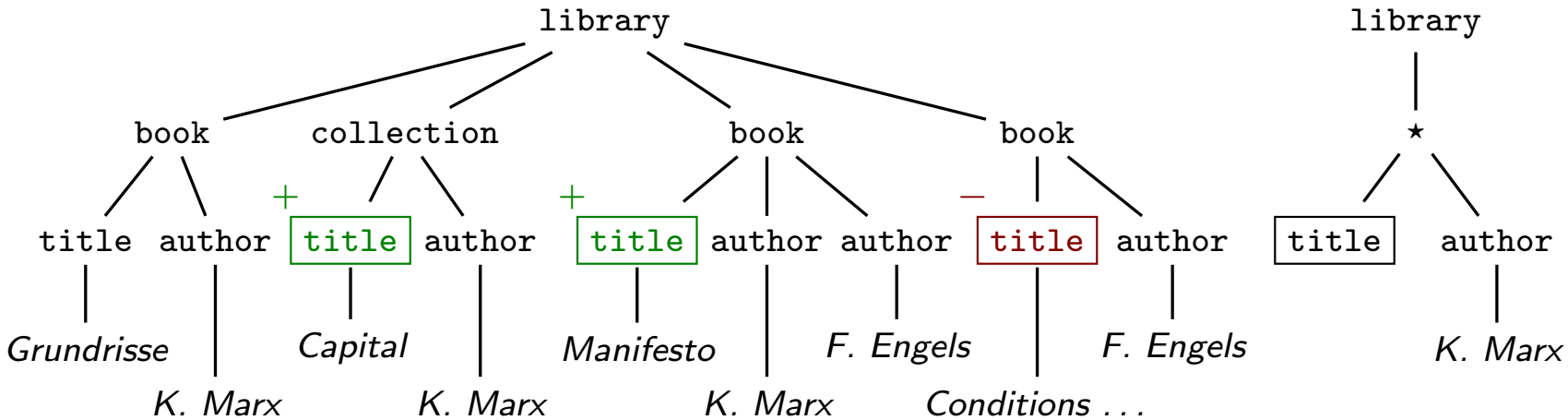
XML, Examples, and Unary Twig queries (tree patterns)



Positive examples (required nodes)

Negative examples (forbidden nodes)

XML, Examples, and Unary Twig queries (tree patterns)

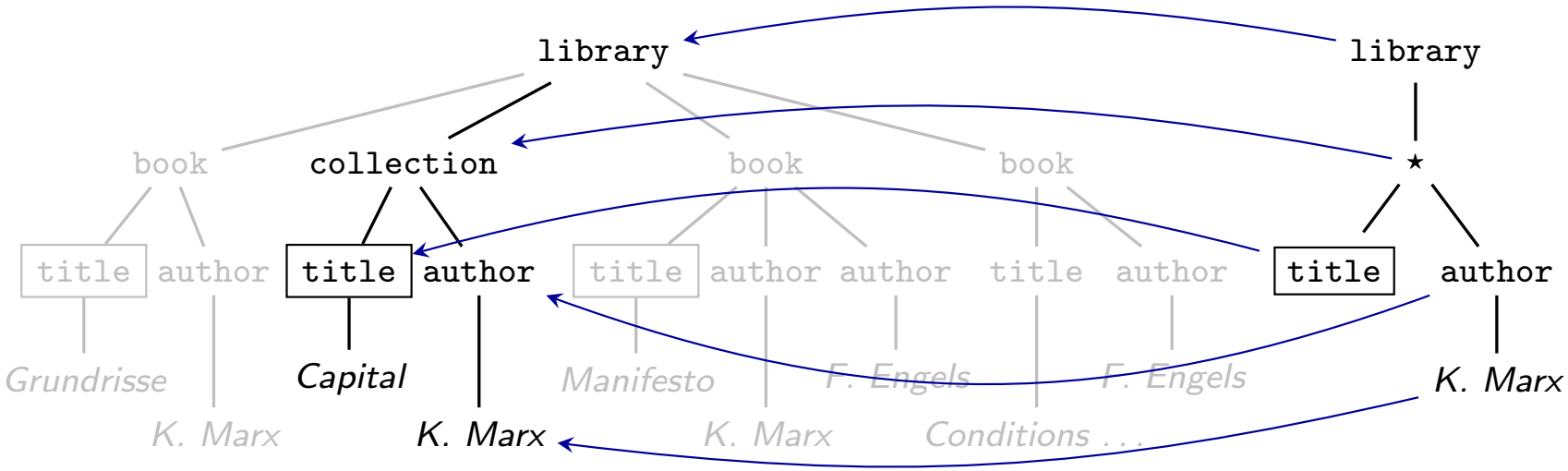


Positive examples (required nodes)

Negative examples (forbidden nodes)

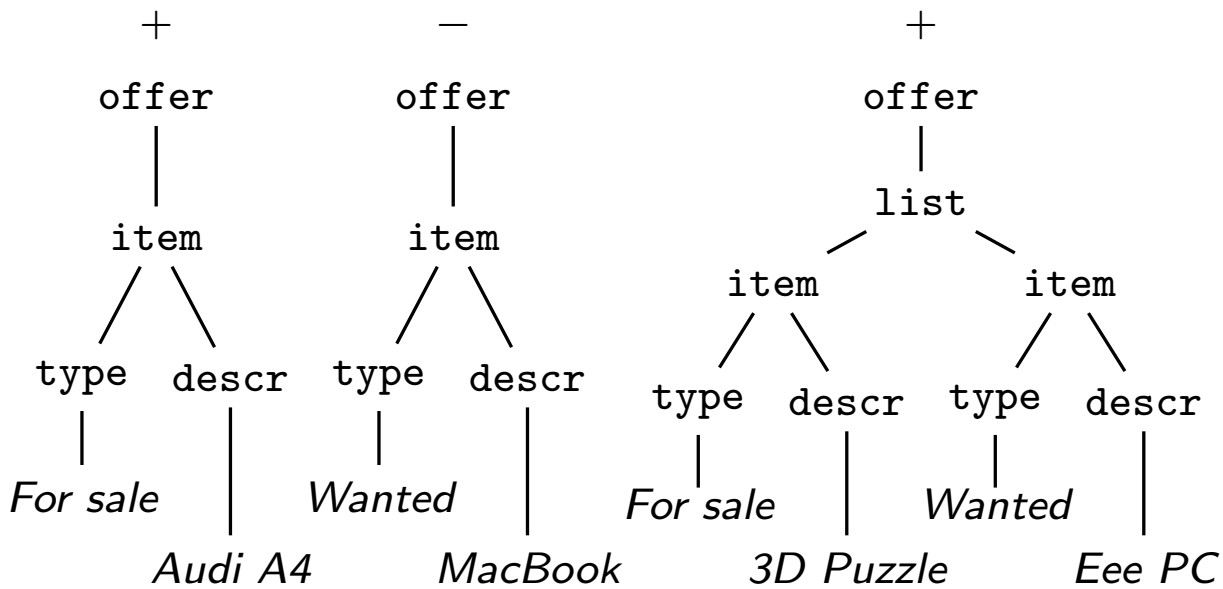
$$/\texttt{library}/\star[\texttt{author} = \text{'K. Marx'}]/\texttt{title}$$

XML, Examples, and Unary Twig queries (tree patterns)

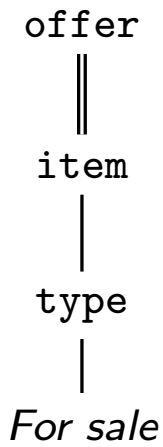


Boolean queries

Positive and Negative Examples:



Output Query:

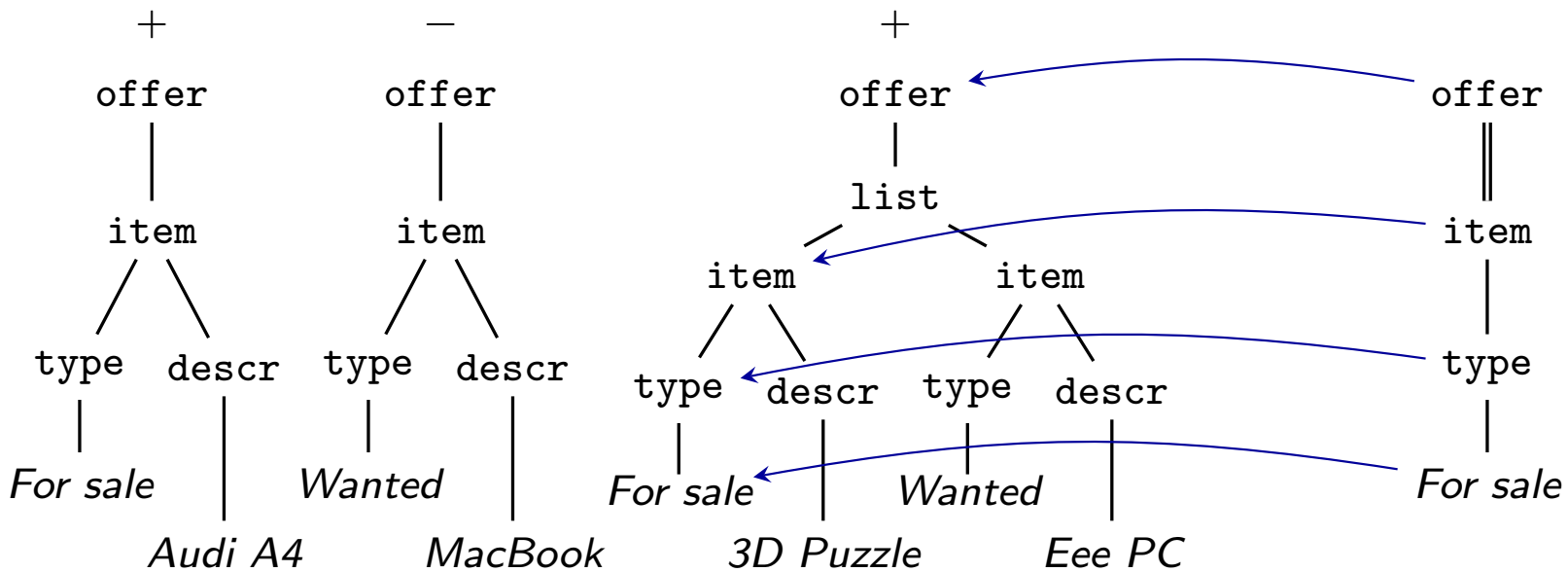


(Path query)

Boolean queries

Positive and Negative Examples:

Output Query:



(Path query)

What is learnability of queries?

Learning model

When is a class of queries $\mathcal{Q}$ learnable from examples?

There is an algorithm *learner* that takes a set of examples and returns a query in $\mathcal{Q}$.

1. *learner* works in polynomial time.
2. *learner* is sound i.e., it returns a query consistent with the set of examples (a query that selects all positive examples and none of negative examples);
   If no such query exists, then *learner* returns a null value.

Learning model

When is a class of queries $\mathcal{Q}$ learnable from examples?

There is an algorithm *learner* that takes a set of examples and returns a query in $\mathcal{Q}$.

1. *learner* works in polynomial time.
2. *learner* is sound i.e., it returns a query consistent with the set of examples (a query that selects all positive examples and none of negative examples);
   If no such query exists, then *learner* returns a null value.

Bad news: Intractability of the consistency problem

Deciding whether there exists a Twig query consistent with given a set of positive and negative examples, is NP-complete.

Dealing with the bad news

1. Approximate learning

The result query may select some negative examples and omit some of the positive ones.

2. Richer query classes

Consistency for unions of Twig queries is in PTIME.

3. Use positive examples only

In the presence of positive examples only, consistency for Twig queries is trivial (the universal query selects all nodes).

Dealing with the bad news

1. Approximate learning

The result query may select some negative examples and omit some of the positive ones.

2. Richer query classes

Consistency for unions of Twig queries is in PTIME.

3. Use positive examples only

In the presence of positive examples only, consistency for Twig queries is trivial (the universal query selects all nodes).

Learning from positive examples only

When is a class of XML queries learnable from positive examples?

There is an algorithm *learner* that takes a set of positive examples and returns a query.

1. *learner* works in polynomial time.
2. *learner* is sound i.e., it returns a query consistent with the set of examples.

Is that sufficient?

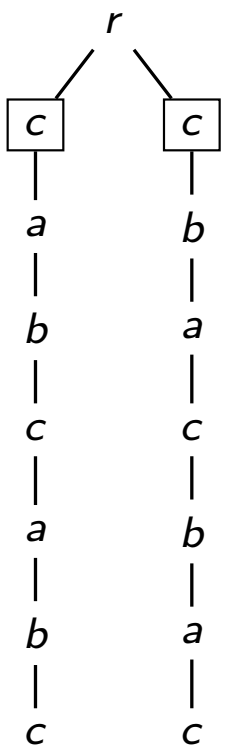
No. A trivial algorithm always returning the universal query $//\star$ satisfies 1 and 2.

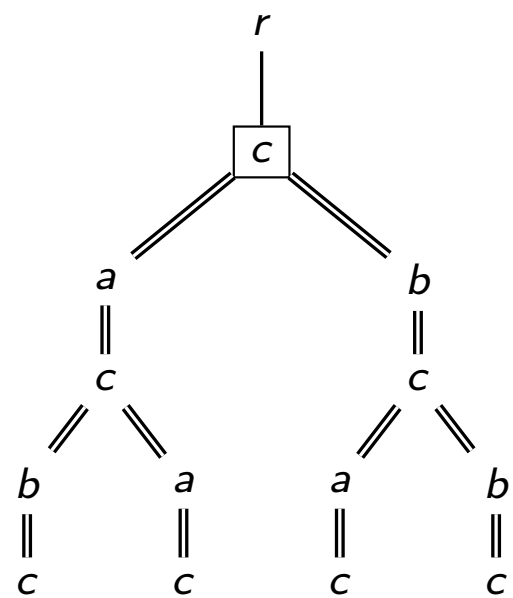
Minimality

What if we would require *learner* to return a minimal query (w.r.t. containment) that is consistent with the input set of examples?

Incompatible with polynomiality of *learner*

A set of examples may have a minimal Twig query of exponential size.





Completeness

Identification in the limit (Gold '67)

A good learning algorithm should be able to infer any concept with a sufficiently rich set of examples.

When is a class of XML queries learnable from positive examples?

There is an algorithm *learner* that takes a set of positive examples S and returns a query.

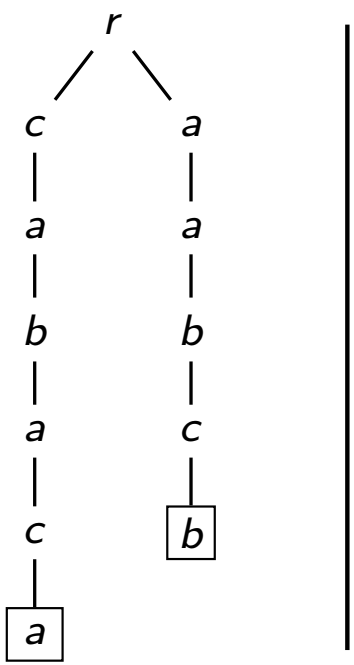
1. *learner* works in polynomial time.
2. *learner* is sound i.e., it returns a query consistent with the set of examples.
3. *learner* is complete: for every query $q \in \mathcal{Q}$ there is a (polynomially sized) set of examples CS_q for which *learner* returns q. Furthermore, CS_q is robust under inclusion of non-essential examples.

Often, CS_q is called the characteristic sample for q w.r.t. *learner*.

Learning Algorithms

Learning unary path queries

Examples:

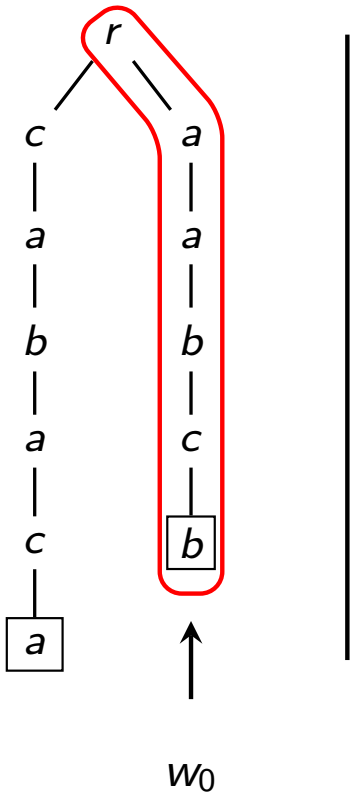


Query inference:

Invariant: constructed query is consistent with the examples

Learning unary path queries

Examples:



w_0

Find the smallest selecting path w_0

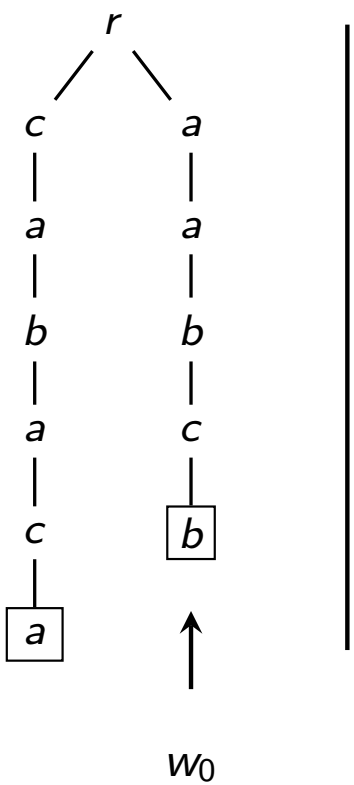
①

Query inference:

Invariant: constructed query is consistent with the examples

Learning unary path queries

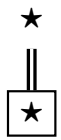
Examples:



w_0

Find the smallest selecting path w_0

①



Start with the universal query

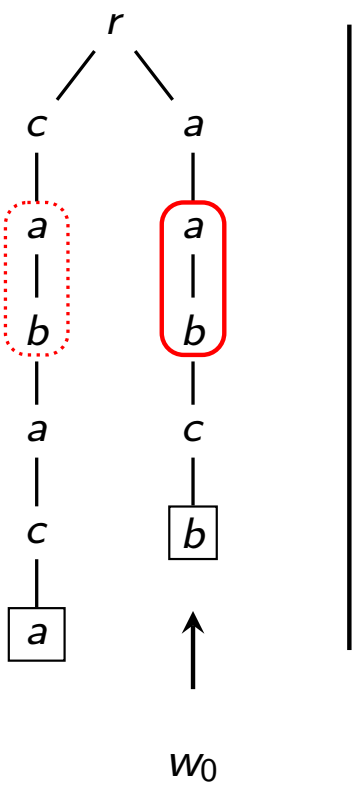
②

Query inference:

Invariant: constructed query is consistent with the examples

Learning unary path queries

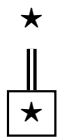
Examples:



w_0

Find the smallest selecting path w_0

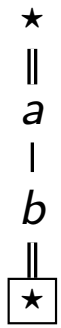
①



Start with the universal query

②

Query inference:



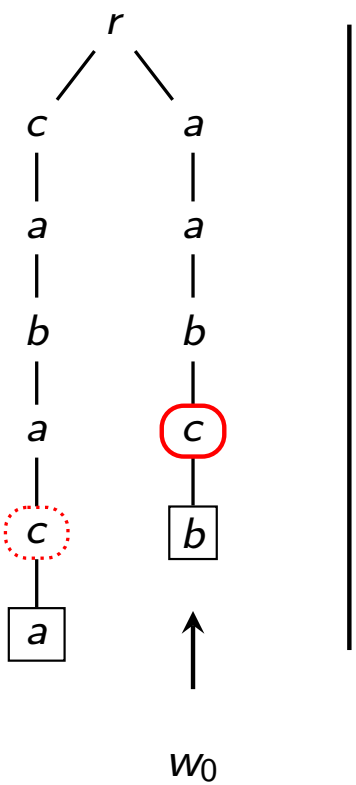
Saturate with common factors (found in w_0)

③

Invariant: constructed query is consistent with the examples

Learning unary path queries

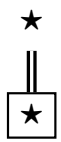
Examples:



w_0

Find the smallest selecting path w_0

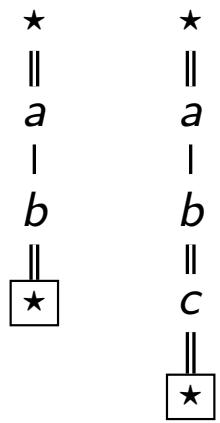
①



Start with the universal query

②

Query inference:



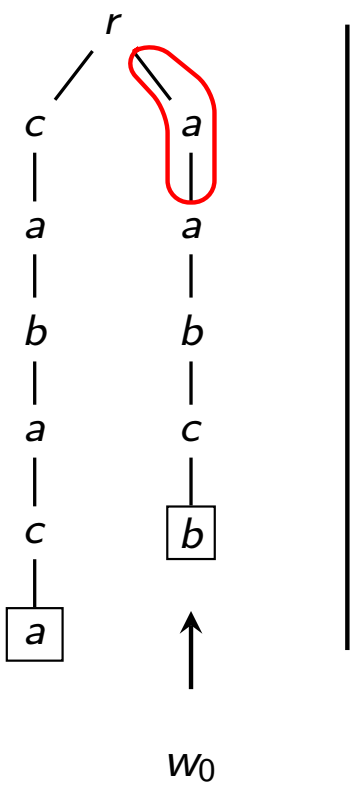
Saturate with common factors (found in w_0)

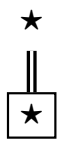
③

Invariant: constructed query is consistent with the examples

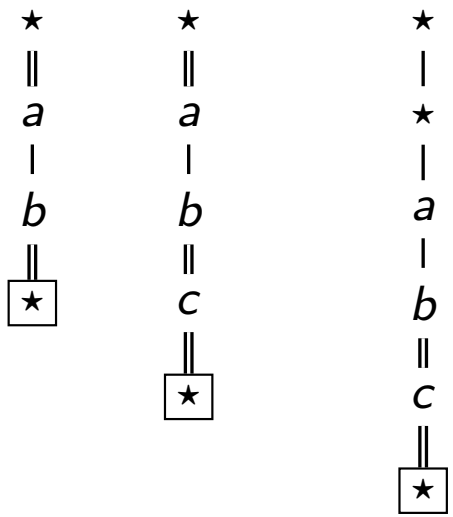
Learning unary path queries

Examples:





Query inference:



Invariant: constructed query is consistent with the examples

Find the smallest selecting path w_0

①

Start with the universal query

②

Saturate with common factors (found in w_0)

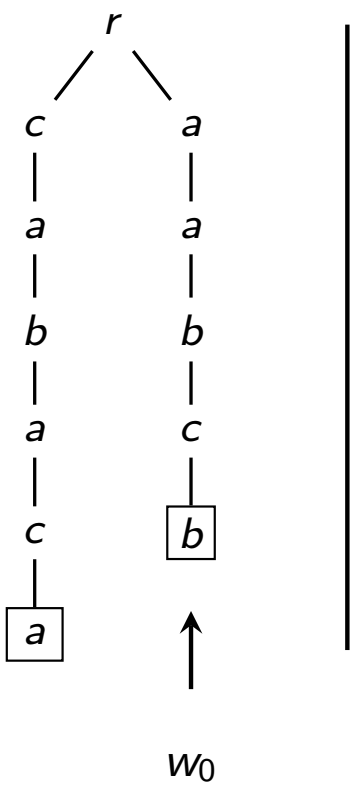
③

Specialize //-edges

④

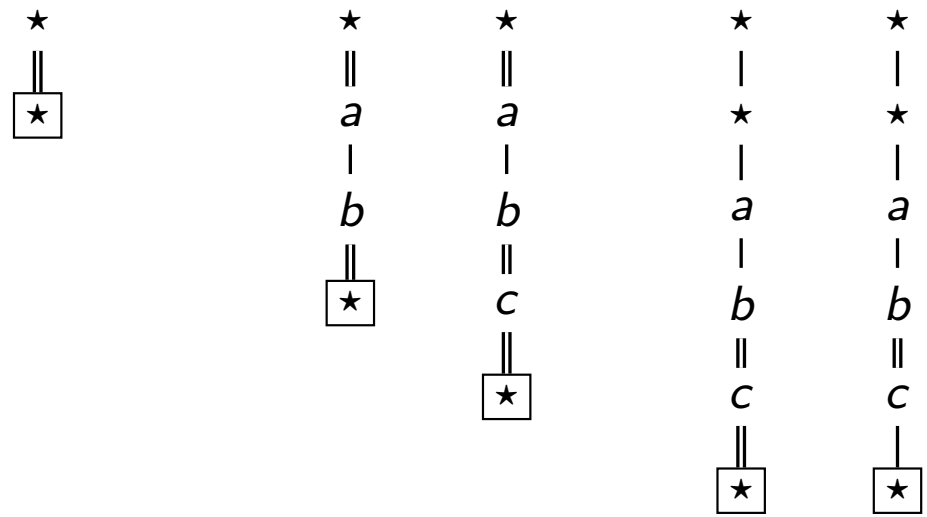
Learning unary path queries

Examples:



w_0

Query inference:



Invariant: constructed query is consistent with the examples

Find the smallest selecting path w_0

①

Start with the universal query

②

Saturate with common factors (found in w_0)

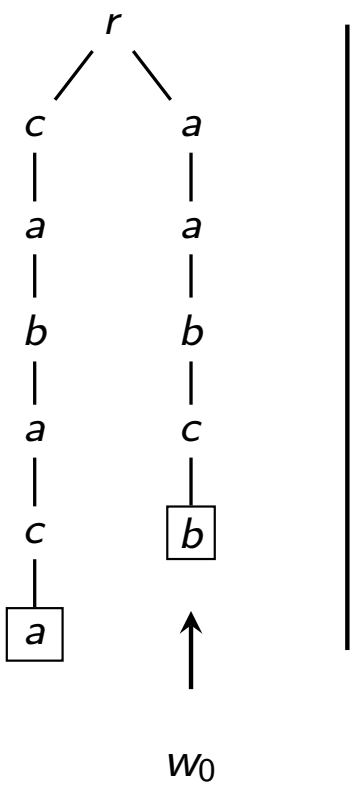
③

Specialize //-edges

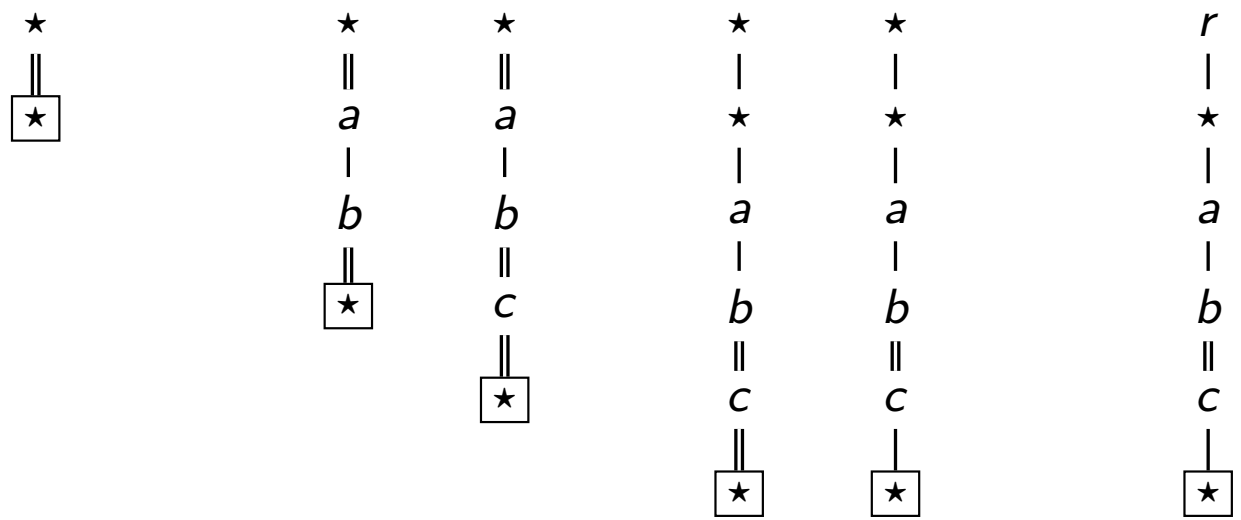
④

Learning unary path queries

Examples:



Query inference:



a minimal consistent query

Invariant: constructed query is consistent with the examples

Find the smallest selecting path w_0 ①

Start with the universal query ②

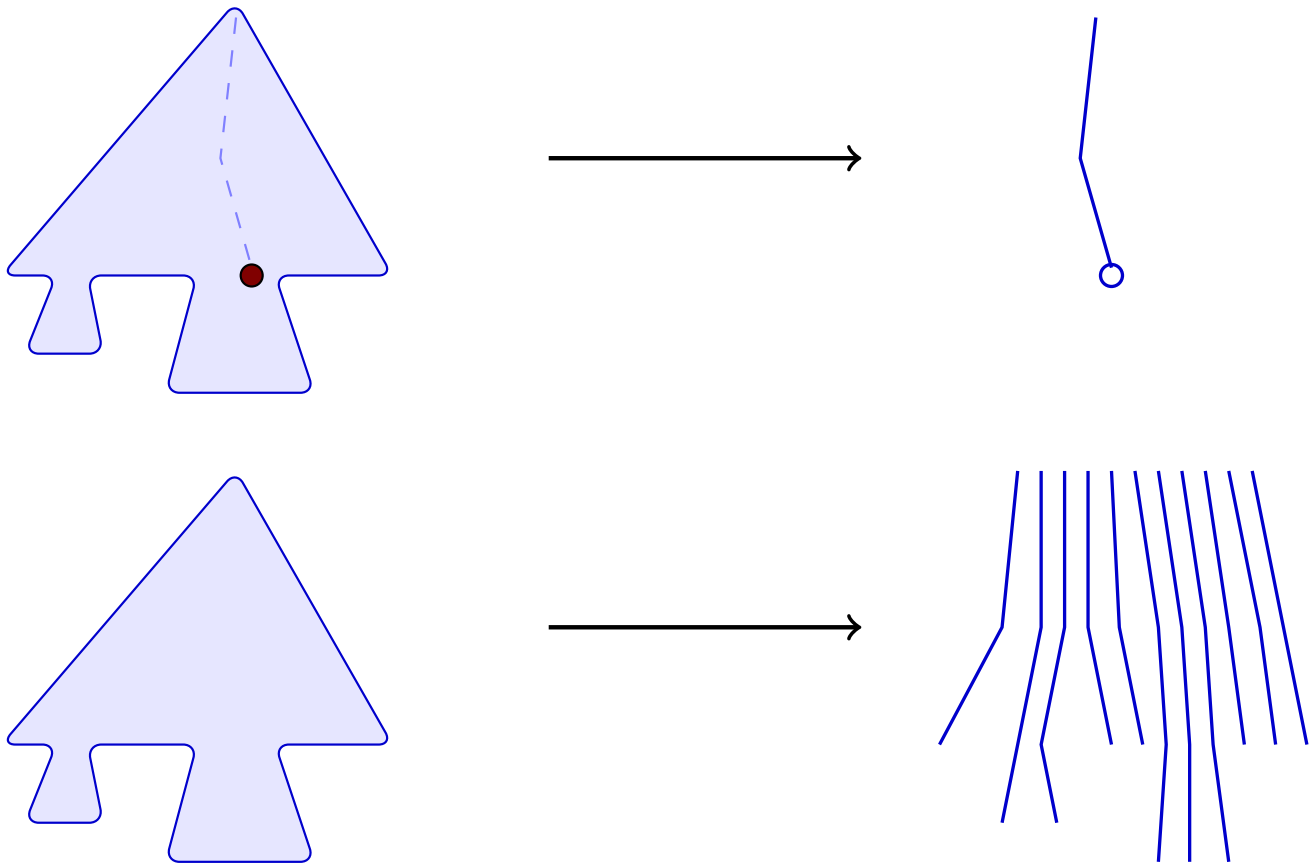
Saturate with common factors (found in w_0) ③

Specialize //-edges ④

Specialize root and leaf ⑤

Learning Boolean path queries

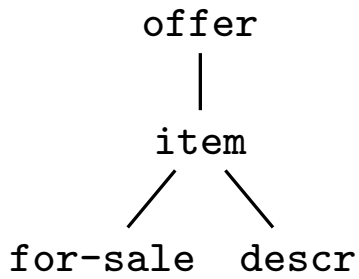
What is difficult about learning Boolean (path) queries?

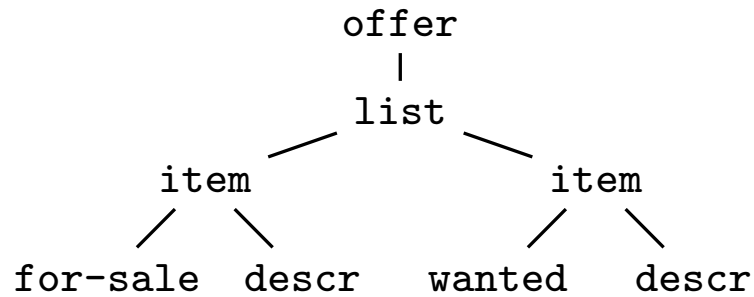


Problem: Every tree offers several paths to choose from

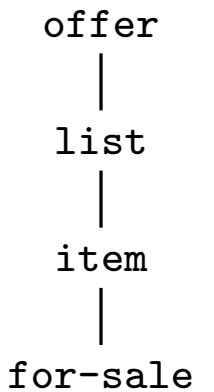
Solution: Use all of them and infer sets (conjunctions) of queries

Learning conjunctions of Boolean path queries





Invariant: constructed query is satisfied in all trees

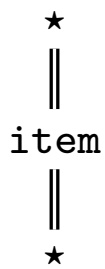


Initialize w_0

①

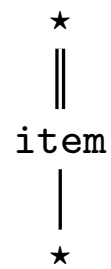
Start with the universal query

②



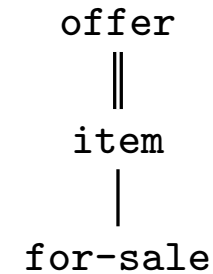
Saturate with common factors

③



Specialize //-edges

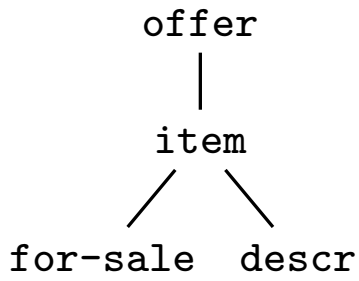
④

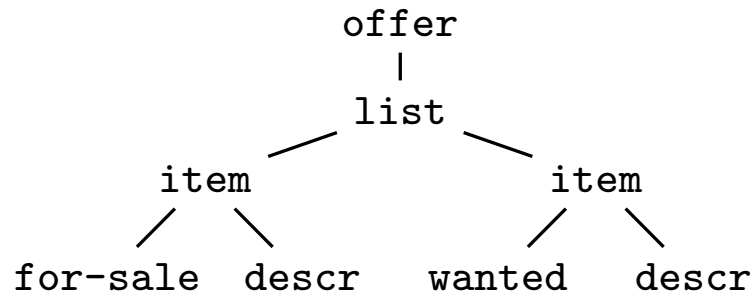


Specialize root and leaf

⑤

Learning conjunctions of Boolean path queries





w_0		Inferred query
offer/item/for-sale	$\longrightarrow$	offer//item/for-sale
offer/item/descr	$\longrightarrow$	offer//item/descr
offer/list/item/for-sale	$\longrightarrow$	offer//item/for-sale
offer/list/item/descr	$\longrightarrow$	offer//item/descr
offer/list/item/wanted	$\longrightarrow$	offer//item//$\star$

Obtained conjunction of (minimal) path queries

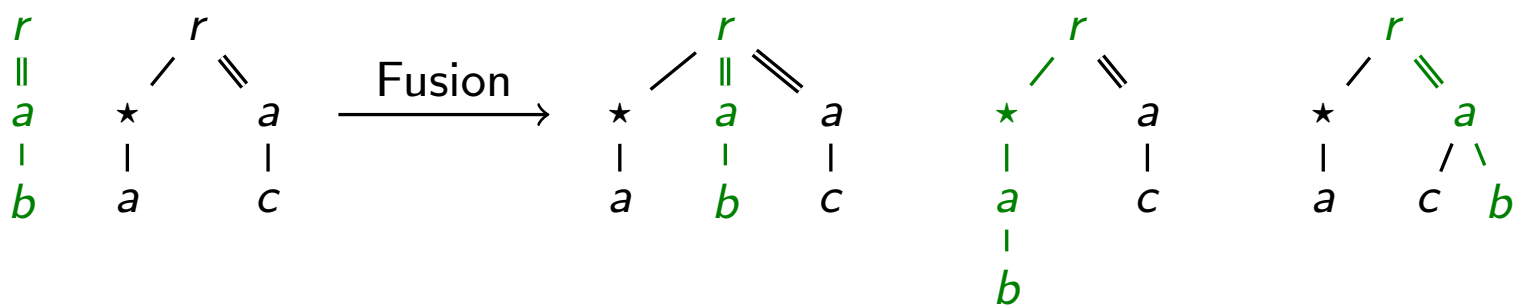
offer//item/for-sale offer//item/descr

Learning Boolean Twig Queries

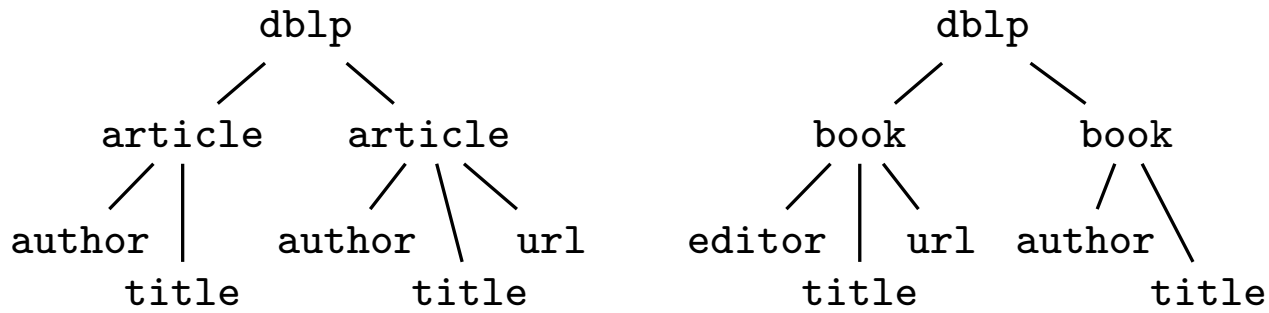
Weaving a Twig query with path queries

Fusing a path query p into Twig query f

1. split of p into some $p_1 \cdot p_2$
2. embed p_1 into f and let p_1 end at n
3. attach p_2 at n



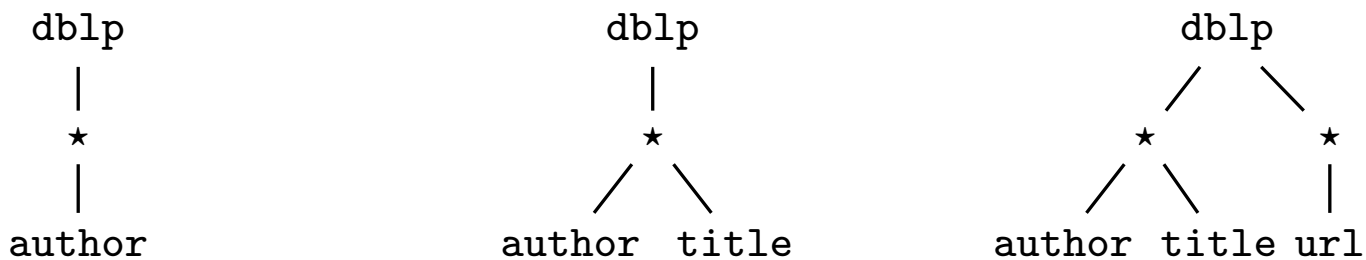
Learning Boolean Twig queries



Learning a conjunction of Boolean path queries yields:

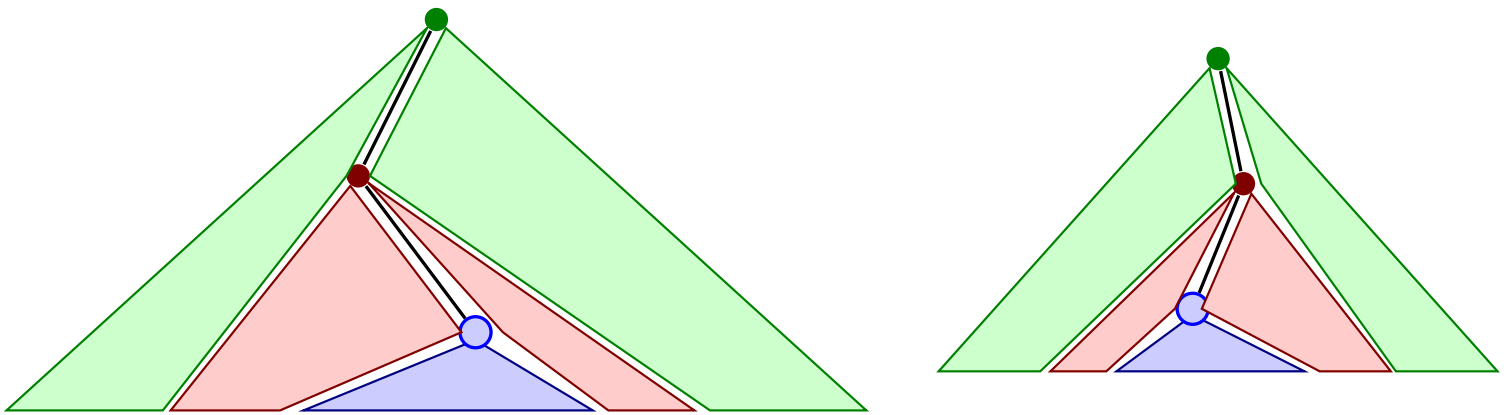
`dblp/⋆/author` `dblp/⋆/title` `dblp/⋆/url`

Iteratively fuse these queries, at each step taking a minimal consistent query.

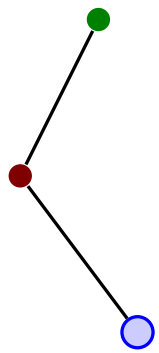


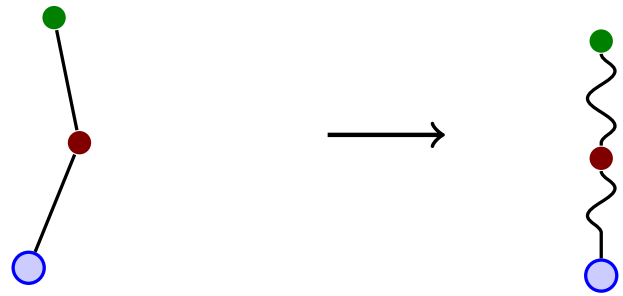
Learning Unary Twig Queries

Learning unary Twig queries

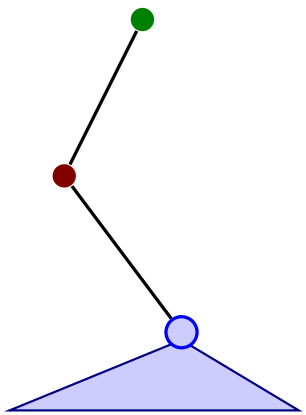


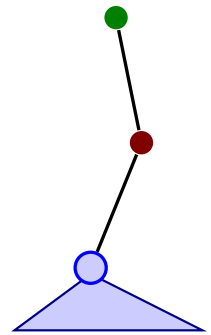
Learning unary Twig queries

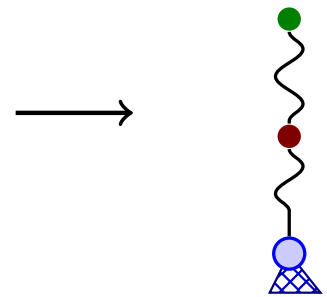




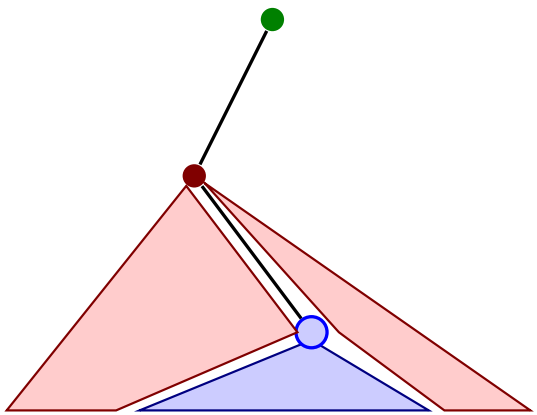
Learning unary Twig queries

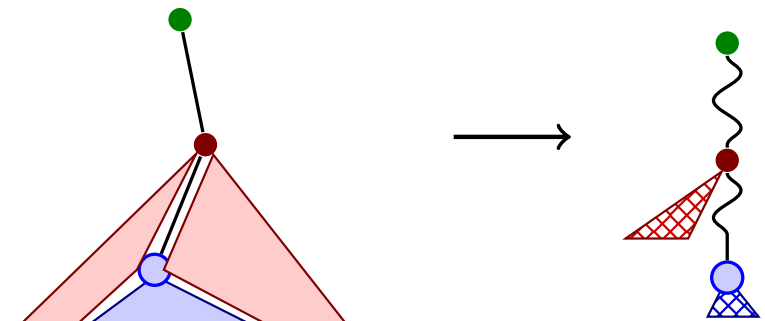




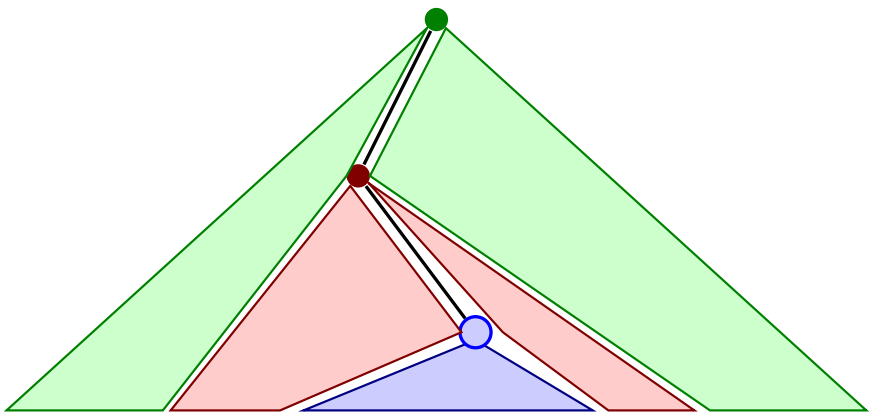


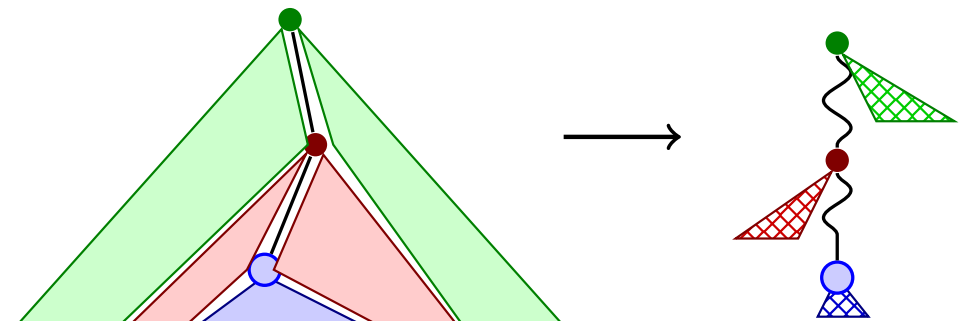
Learning unary Twig queries



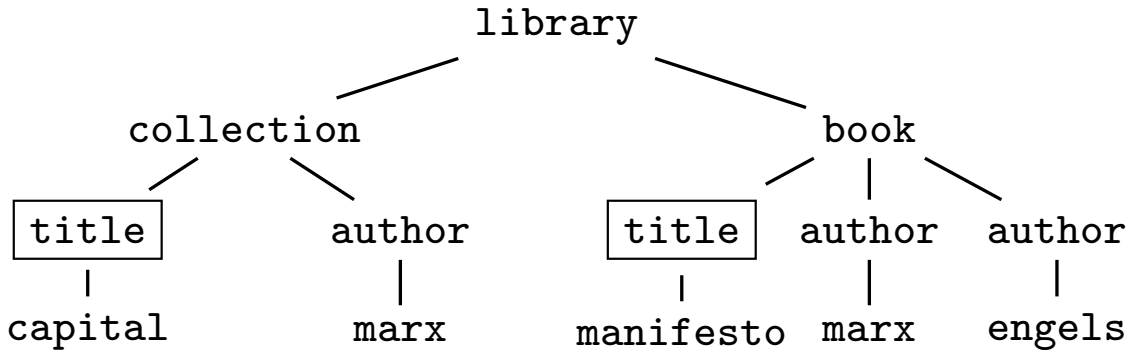


Learning unary Twig queries

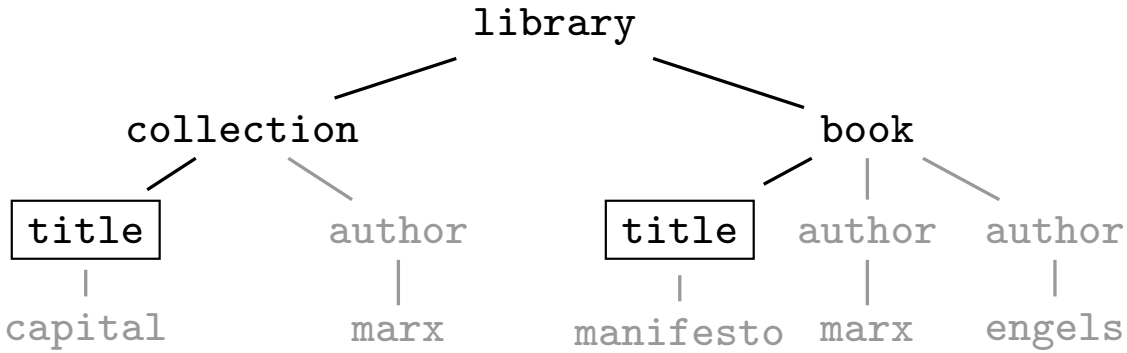


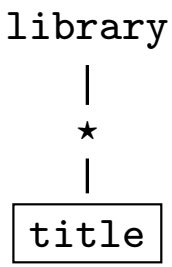


Learning unary Twig queries (cont'd.)

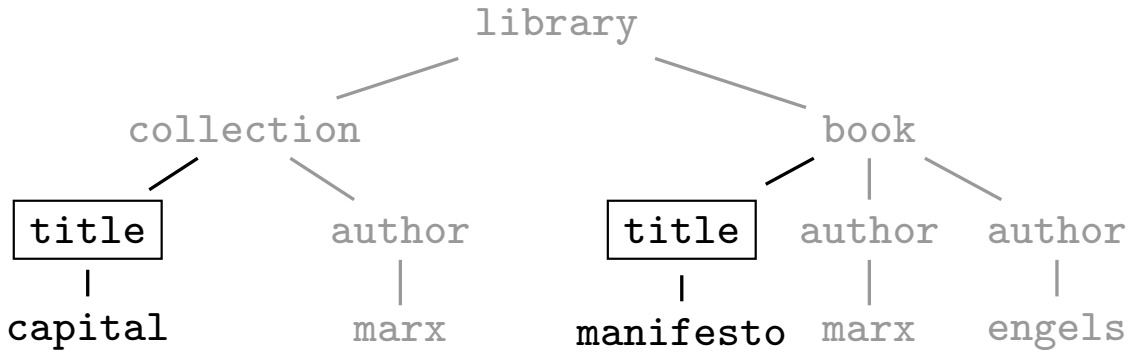


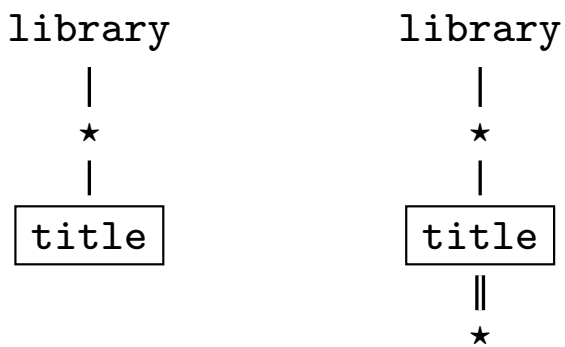
Learning unary Twig queries (cont'd.)



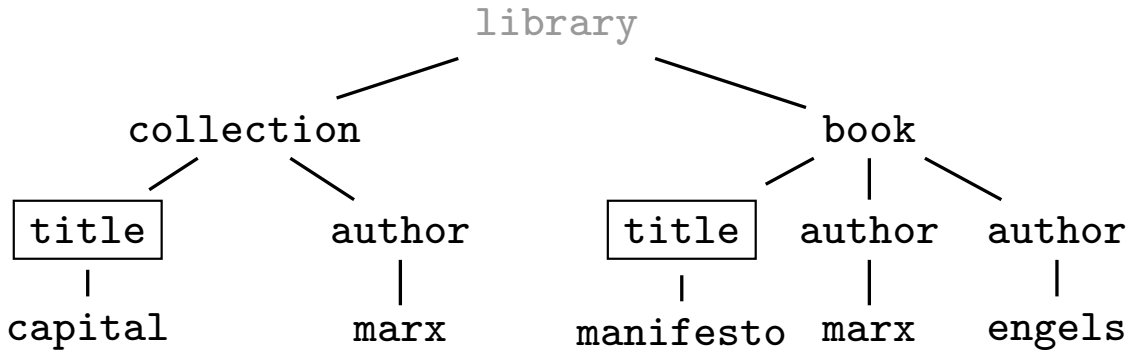


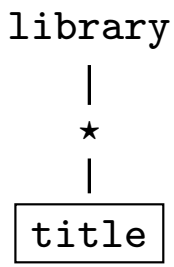
Learning unary Twig queries (cont'd.)

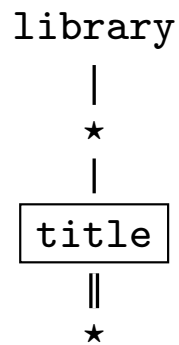


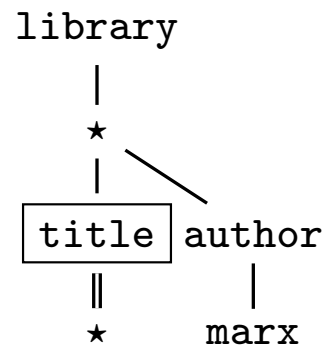


Learning unary Twig queries (cont'd.)









Conclusions and what's in the paper

Learnability (in the limit) of Twig queries

- Unfeasible in the presence of positive and negative examples
- Feasible in the presence of positive examples only

Anchored Twig queries

- //-edge cannot be incident to $\star$ unless it is the root a leaf
- $\star$ may be a non-selecting leaf node only if it is incident to //-edge

Two essential properties of Anchored Twig queries that enable our algorithms

P1 Containment of queries can be tested with embeddings (and thus is in PTIME)

P2 Existence of polynomially sized *match sets* (for completeness) [Miklau, Suciu, 2004]