Consistency of injective tree patterns

Claire David       Nadime Francis       Filip Murlak
U. Paris-Est Marne  ENS Cachan          U. Warsaw

FIT 2015, Warsaw
Problem

Can given pattern $\pi$ be matched in some tree from regular language $L$?
Semantics 1: homomorphisms

Pattern $\pi$ is matched in tree $T$ if there is a homomorphism $h: \pi \rightarrow T$.

(Preserves labels, child and descendant relations.)
Semantics 1: homomorphisms

Pattern $\pi$ is matched in tree $T$ if there is a homomorphism $h: \pi \rightarrow T$.
(Preserves labels, child and descendant relations.)

- XPath(//b, []), query language for XML docs [Miklau, Suciu '02]

  $a[//b]/[//b, c]$ (XML docs are trees)
Semantics 1: homomorphisms

Pattern $\pi$ is matched in tree $T$ if there is a homomorphism $h: \pi \to T$.

(Preserves labels, child and descendant relations.)

- XPath(///, []), query language for XML docs [Miklau, Suciu ’02]
  
  $a[//b]//[//b,c]$ (XML docs are trees)

- Conjunctive queries over trees [Gottlob, Koch, Schultz ’04]

  $\exists x_1 \cdots \exists x_5 \ a(x_1) \land b(x_3) \land b(x_4) \land c(x_5) \land$
  
  $\land \ child(x_1, x_2) \land desc(x_1, x_3) \land$
  
  $\land desc(x_2, x_4) \land child(x_2, x_5)$
Semantics 1: homomorphisms

Pattern $\pi$ is matched in tree $T$ if there is a homomorphism $h: \pi \to T$.

(Preserves labels, child and descendant relations.)

- XPath($/,, [], []$), query language for XML docs [Miklau, Suciu ’02]
  
  $a[//b]//[//b,c]$  
  
  (XML docs are trees)

- Conjunctive queries over trees [Gottlob, Koch, Schultz ’04]

  $\exists x_1 \cdots \exists x_5 a(x_1) \land b(x_3) \land b(x_4) \land c(x_5) \land$
  $\land child(x_1, x_2) \land desc(x_1, x_3) \land$
  $\land desc(x_2, x_4) \land child(x_2, x_5)$
Semantics 2: injective

Pattern $\pi$ is matched in tree $T$ if there is an injective homomorphism $h: \pi \to T$.

(All nodes mapped to different tree nodes.)
Semantics 2: injective

Pattern $\pi$ is matched in tree $T$ if there is an injective homomorphism $h: \pi \rightarrow T$.
(All nodes mapped to different tree nodes.)

- incomplete information about XML docs in DOM model
  - [Barceló, Libkin, Poggi, Sirangelo '09]
    - nodes have unique IDs
    - labels and relations may be lost
    - when ID is lost, node is lost

pattern $\pi$: incomplete XML doc;
regular language $L$: correct docs (schema);
$\pi$ is satisfiable iff incomplete doc extends to a correct doc
Regular languages

Sets of finite labelled unranked trees

- definable in monadic second order logic;
- recognizable by automata;
- generated by RelaxNG (covers DTD, XSM).

Define several types of trees, each specified (recursively) by
- the label of the root,
- possible sequences of immediate subtree types (regexp);

and choose some of the types to form your language.

Example:
- type $\tau$: root label $a$, immediate subtree types $\sigma^* \tau \sigma^* + \epsilon$;
- type $\sigma$: root label $b$, immediate subtree types $\sigma^*$;
- choose $\tau$. 

Regular languages

Sets of finite labelled unranked trees

- definable in monadic second order logic;
- recognizable by automata;
- generated by RelaxNG (covers DTD, XSM).

Define several types of trees, each specified (recursively) by

- the label of the root,
- possible sequences of immediate subtree types (regexp);

and choose some of the types to form your language.
Regular languages

Sets of finite labelled unranked trees
- definable in monadic second order logic;
- recognizable by automata;
- generated by RelaxNG (covers DTD, XSM).

Define several types of trees, each specified (recursively) by
- the label of the root,
- possible sequences of immediate subtree types (regexp);
and choose some of the types to form your language.

Example: *a*-only path from root to leaf, *b*’s elsewhere
- type $\tau$: root label *a*, immediate subtree types $\sigma^*\tau\sigma^* + \epsilon$;
- type $\sigma$: root label *b*, immediate subtree types $\sigma^*$;
- choose: $\tau$
Satisfiability of tree patterns: complexity landscape

<table>
<thead>
<tr>
<th>Semantics</th>
<th>Consistency (fixed $L$)</th>
<th>Satisfiability (given $L$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Homomorphism</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Injective</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Satisfiability of tree patterns: complexity landscape

<table>
<thead>
<tr>
<th>Semantics</th>
<th>Consistency (fixed $L$)</th>
<th>Satisfiability (given $L$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Homomorphism</td>
<td>NP-complete</td>
<td>Can find polynomial witness.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Can reduce NP-complete problem:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>does given regular expression $\alpha$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>generate a word using all letters in $\alpha$.</td>
</tr>
<tr>
<td>Injective</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Satisfiability of tree patterns: complexity landscape

<table>
<thead>
<tr>
<th>Semantics</th>
<th>Consistency (fixed $L$)</th>
<th>Satisfiability (given $L$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Homomorphism</td>
<td>NP-complete, even child-only [Benedikt, Fan, Geerts ’05]</td>
<td>NP-complete</td>
</tr>
<tr>
<td></td>
<td>Can find polynomial witness.</td>
<td>Can reduce NP-complete problem:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>does given regular expression $\alpha$ generate a word using all letters in $\alpha$.</td>
</tr>
</tbody>
</table>
Satisfiability of tree patterns: complexity landscape

<table>
<thead>
<tr>
<th>Semantics</th>
<th>Consistency (fixed $L$)</th>
<th>Satisfiability (given $L$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Homomorphism</td>
<td>NP-complete, even child-only [Benedikt, Fan, Geerts '05]</td>
<td>NP-complete</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Can find polynomial witness.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Can reduce NP-complete problem:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>does given regular expression $\alpha$ generate a word using all letters in $\alpha$.</td>
</tr>
<tr>
<td>Injective</td>
<td>?</td>
<td></td>
</tr>
</tbody>
</table>
Zooming in on consistency under injective semantics

[Barceló, Libkin, Poggi, Sirangelo ’09]

- PTIME without descendant.
  - Can given set of words be matched disjointly in a word from $L$?
Zooming in on consistency under injective semantics

[Barceló, Libkin, Poggi, Sirangelo ’09]
- PTIME without descendant.
  - Can given set of words be matched disjointly in a word from \( L \)?

[Kopczyński ’11]
- PTIME with \( \leq 1 \) descendant on each branch.
  - Can given set of trees be matched disjointly in a tree from \( L \)?
Zooming in on consistency under injective semantics

[Barceló, Libkin, Poggi, Sirangelo ’09]
▷ PTIME without descendant.
  ▷ Can given set of words be matched disjointly in a word from $L$?

[Kopczyński ’11]
▷ PTIME with $\leq 1$ descendant on each branch.
  ▷ Can given set of trees be matched disjointly in a tree from $L$?

[David, Francis, Murlak ’14] (this work)
▷ $\leq 2$ descendants on each branch: NP-complete;
Zooming in on consistency under injective semantics

[Barceló, Libkin, Poggi, Sirangelo ’09]
- PTIME without descendant.
  - Can given set of words be matched disjointly in a word from $L$?

[Kopczyński ’11]
- PTIME with $\leq 1$ descendant on each branch.
  - Can given set of trees be matched disjointly in a tree from $L$?

[David, Francis, Murlak ’14] (this work)
- $\leq 2$ descendants on each branch: NP-complete;
- PTIME (FPT) with descendant only (without child)
  - works for injective and homomorphism semantics,
  - extends to patterns with sibling order (following-sibling).
On the hardness proof
Challenges

1. Our FPT algorithm has complexity $2^{2\|L\|} \cdot \text{poly}(\|\pi\|)$; the NP upper bound gives complexity $2^{\text{poly}(\|L\|,\|\pi\|)}$. Inherent trade-off, or can this be reconciled?

2. Assuming we stick to descendant and following sibling, is consistency still tractable for patterns that are DAGs, not trees?