Learning Tree to Word Transducers

LATA 2014

Aurélien Lemay

joint work with:
Grégoire Laurence
Joachim Niehren
Slawek Staworko
Marc Tommasi

March 11, 2014

Learning Tree Transductions

Transforming structured datas

Example of XSLT transformation : from XML to XHTML

- Many applications, many formalisms...
- requires some expertise
- One solution : infer the transformation

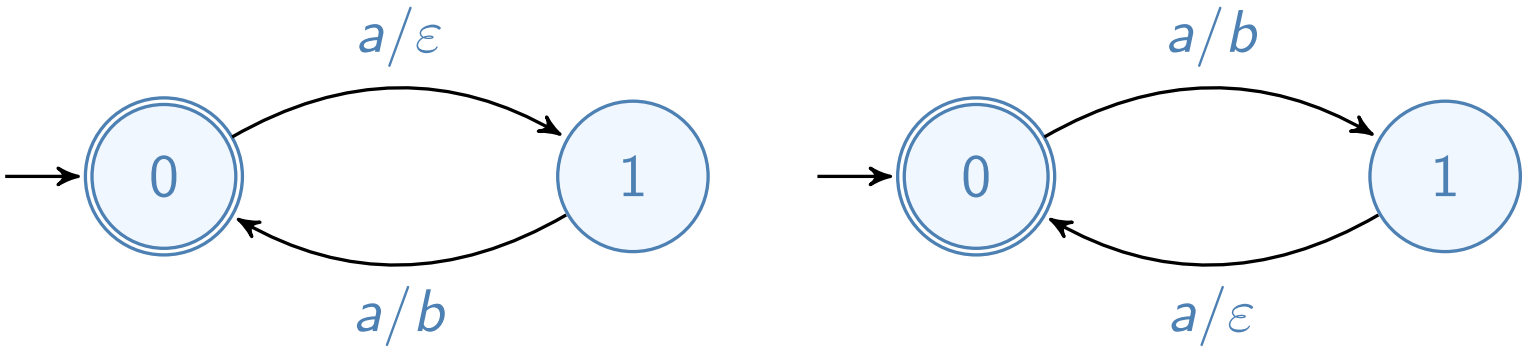
Learning Subsequential Transducer

Learning Subsequential Transducer[OncinaGarciaVidal93]

Subsequential transducers are learnable from examples with polynomial time and data (Gold Model [Gold78])

Two main ideas:

- Onward normal form [Choffrut79] : produce the output as soon as possible



Two subsequential transducers for $\tau(a^{2n}) = b^n$

- State merging algorithm : OSTIA [OncinaGarciaVidal93]

Extensions of OSTIA - two learnable classes

- **Rational Functions** [BoiretLemayNiehren12]
  - Represented by Subsequential transducers w. deterministic look-ahead
  - Normal form (inspired by bimachines [ReteunauerSchutzenberger92])
  - Learning algorithm $\simeq$ learn the look-ahead, then apply OSTIA
- **Top-Down Tree-to-Tree Transducers** [LemayManethNiehren11]
  - Earliest normal form [EngelfrietManethSeidl09] : earliest production (produce as 'up' as possible),
  - Myhill-Nerode kind of theorem in [LemayManethNiehren11]
  - Learning based on a state merging algorithm

Toward learning MSO tree transformations ?

MSO tree Transformation [Courcelle92] : an interesting target for learning tree transformation !

The big picture

MSO tree transformations
$\simeq$ Macro Tree Transducers w. regular look-ahead (MTT^R)
[EngelfrietManeth03]
$\simeq$ Top-Down + **Concatenation** + Look-ahead

- Top-Down Tree transducers : learnable ✓
- Look-ahead : learnable ✓
  - not extended to trees yet
- Concatenation in the output : ?

Outline

Outline

Tree to Word Transducers - An example

XML-like Serialization

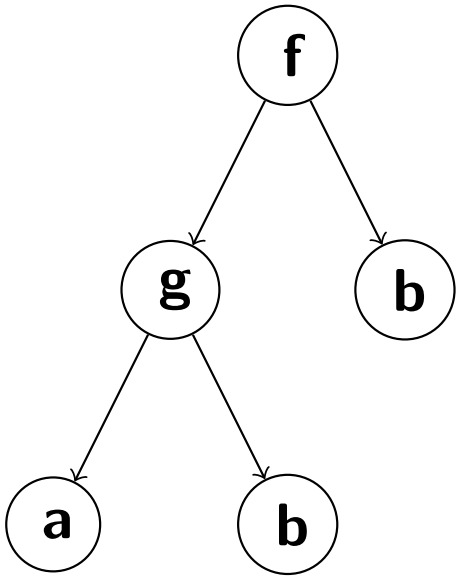
Axiom : $q(x_0)$

$q(f(x_1, x_2)) \rightarrow < f > \cdot q(x_1) \cdot q(x_2) < /f >$

$q(g(x_1, x_2)) \rightarrow < g > \cdot q(x_1) \cdot q(x_2) < /g >$

$q(a) \rightarrow < a/ >$

$q(b) \rightarrow < b/ >$



Tree to Word Transducers - An example

XML-like Serialization

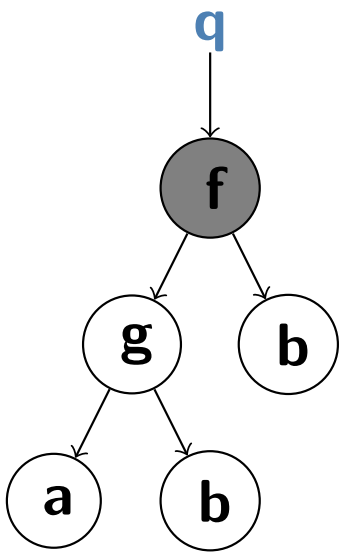
Axiom : $q(x_0)$

$q(f(x_1, x_2)) \rightarrow < f > \cdot q(x_1) \cdot q(x_2) < /f >$

$q(g(x_1, x_2)) \rightarrow < g > \cdot q(x_1) \cdot q(x_2) < /g >$

$q(a) \rightarrow < a/ >$

$q(b) \rightarrow < b/ >$



Tree to Word Transducers - An example

XML-like Serialization

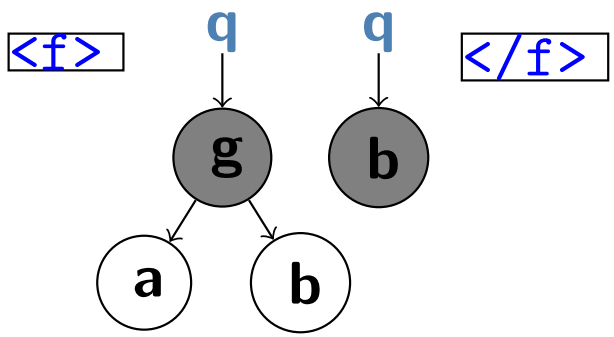
Axiom : $q(x_0)$

$q(f(x_1, x_2)) \rightarrow < f > \cdot q(x_1) \cdot q(x_2) < /f >$

$q(g(x_1, x_2)) \rightarrow < g > \cdot q(x_1) \cdot q(x_2) < /g >$

$q(a) \rightarrow < a/ >$

$q(b) \rightarrow < b/ >$



Tree to Word Transducers - An example

XML-like Serialization

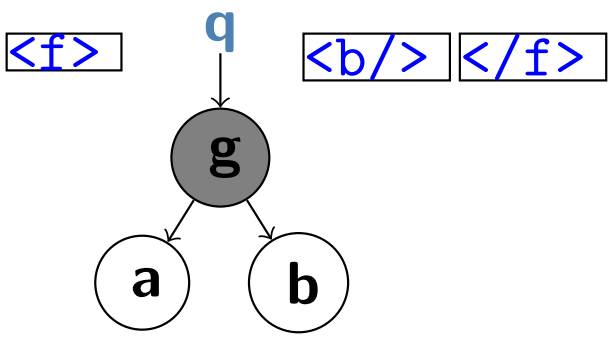
Axiom : $q(x_0)$

$q(f(x_1, x_2)) \rightarrow < f > \cdot q(x_1) \cdot q(x_2) < /f >$

$q(g(x_1, x_2)) \rightarrow < g > \cdot q(x_1) \cdot q(x_2) < /g >$

$q(a) \rightarrow < a/ >$

$q(b) \rightarrow < b/ >$



Tree to Word Transducers - An example

XML-like Serialization

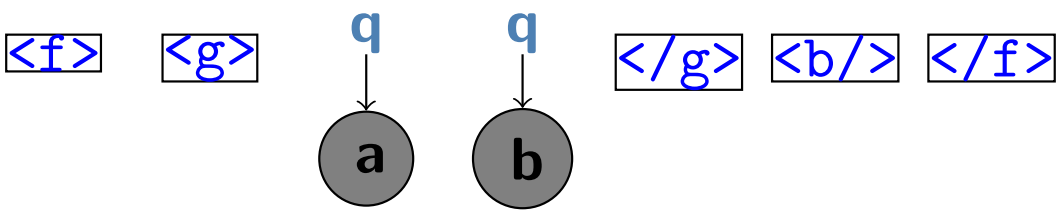
Axiom : $q(x_0)$

$q(f(x_1, x_2)) \rightarrow < f > \cdot q(x_1) \cdot q(x_2) < /f >$

$q(g(x_1, x_2)) \rightarrow < g > \cdot q(x_1) \cdot q(x_2) < /g >$

$q(a) \rightarrow < a/ >$

$q(b) \rightarrow < b/ >$



Tree to Word Transducers - An example

XML-like Serialization

Axiom : $q(x_0)$

$q(f(x_1, x_2)) \rightarrow < f > \cdot q(x_1) \cdot q(x_2) < /f >$

$q(g(x_1, x_2)) \rightarrow < g > \cdot q(x_1) \cdot q(x_2) < /g >$

$q(a) \rightarrow < a/ >$

$q(b) \rightarrow < b/ >$

<f> <g> <a/> <b/> </g> <b/> </f>

Tree to word Transducers - Presentation

Axiom: $u_0 \cdot q(x_0) \cdot u_1$
Rules: $q(f(x_1, x_2)) \rightarrow u_0 \cdot q_1(x_1) \cdot u_1 \cdot q_2(x_2) \cdot u_2$

Three Restrictions

- Deterministic
- Linear (no copy)
- Ordered (no swap)

Deterministic Sequential Tree to Words (STW)

Outline

Normal Form

Earliest STW : produce as soon as possible

Example transformation : count the number of symbols.
$\tau_{count}(f(a, f(a, b))) = \#\#\#\#\#$

An STW for τ_{count}

Axiom: q
$q(f) \to \#q(x_1)q(x_2)$
$q(a) \to \#$
$q(b) \to \#$

Not earliest ! At least one '$\#$' could be output from the beginning.

Normal Form

Another STW for τ_{count} :
Axiom: $\#q$
$q(f) \rightarrow \#q(x_1)\#q(x_2)$
$q(a) \rightarrow \varepsilon$
$q(b) \rightarrow \varepsilon$

Earliest (Rule 1)

Produce as 'up' as possible

Normal Form

We want a unique normal form :
Do we want : $q(f) \rightarrow \#q(x_1)\#q(x_2)$
or $q(f) \rightarrow \#\#q(x_1)q(x_2)$
(or another choice ?)

Earliest - Rule 2

produce as 'left' as possible

Normal Form

Earliest STW (eSTW) : produce as 'up' and as 'left' as possible

Theorem [LaurenceLemayNiehrenStaworkoTommasi11]

For any STW, there exists an equivalent unique minimal eSTW

Possibly of exponential size

The minimal eSTW of τ_{count}

Axiom: $\#q$

$q(f) \to \#\#q(x_1)q(x_2)$

$q(a) \to \varepsilon$

$q(b) \to \varepsilon$

Outline

A Myhill-Nerode Theorem for STW

- constructive algorithm for $can(\tau)$ (minimal eSTW for τ)
- builds for each input path p a τ_p
- $p \simeq p'$ iff $\tau_p = \tau_{p'}$

Myhill-Nerode Theorem for STW

τ is represented by a STW
$\Leftrightarrow \simeq$ is of Finite Index
$\Leftrightarrow can(\tau)$ is the minimal eSTW of τ

Building Axiom

Axiom : $lcp(range(\tau)) \cdot q_\varepsilon \cdot lcs'(range(\tau))$
lcp : longest common prefix
lcs' : longest common suffix (minus what is in lcp)
For τ_{count} :
$lcp(range(\tau_{count})) = \#$
$lcs'(range(\tau_{count})) = \varepsilon$
Axiom : $\#q_\varepsilon$

Building τ_ε

Axiom : $\#q_\varepsilon$
We define τ_ε:
For any t, $\tau_\varepsilon(t) = \#^{-1}\tau_{count}(t)$

Defining τ_ε

$a \to \varepsilon$
$b \to \varepsilon$
$f(a,a) \to \#^2$
...
$f(f(a,b),a) \to \#^4$
...

$\tau_\varepsilon(t)$: $\#^{|t|-1}$

Building Rules for Leaf Symbols

Rules from state q_ε
For leaf symbols :
$\tau_\varepsilon(a) \to \varepsilon$
$\tau_\varepsilon(b) \to \varepsilon$

Rules

$q_\varepsilon(a) \to \varepsilon$
$q_\varepsilon(b) \to \varepsilon$

Building Other Rules (1)

build the rule $q_\varepsilon(f(x_1, x_2)) \to u_0 \cdot q_{(f,1)} \cdot u_1 \cdot q_{(f,2)} \cdot u_2$

First, $u_0 = lcp(\{\tau_\varepsilon)(f(?, ?)))\})$

Compute u_0 from $\tau_\varepsilon(f(?, ?))$

$f(a, a) \to \#^2$

...

$f(f(a, b), a) \to \#^4$

...

$u_0 = \#^2$

$q_\varepsilon(f(x_1, x_2)) \to \#^2 \cdot q_{(f,1)} \cdot u_1 \cdot q_{(f,2)} \cdot u_2$

Building Other Rules (2)

$q_\varepsilon(f(x_1, x_2)) \to \#^2 \cdot q_{(f,1)} \cdot u_1 \cdot q_{(f,2)} \cdot u_2$

To have $\tau_{(f,1)}(t)$, take the lcp of $\tau_\varepsilon(f(t, ?))$

Compute $\tau_{(f,1)}(a)$ from $\tau_\varepsilon(f(a, ?))$

$\tau_\varepsilon(f(a, a)) = \#^2$
$\tau_\varepsilon(f(a, b)) = \#^2$
$\tau_\varepsilon(f(a, f(a, a)) = \#^4$
...
$\tau_\varepsilon(f(a, f(a, f(a, a)))) = \#^6$

$lcp(\tau_\varepsilon(f(a, ?))) = \#^2 = u_0\tau_{(f,1)}(a)u_1$ (guaranteed by earliesness)
As $u_0 = \#^2$, $\tau_{(f,1)}(a) = u_1 = \varepsilon$

Building Other Rules (3)

Compute $\tau_{(f,1)}$

$\tau_{(f,1)}(a) = \varepsilon$
$\tau_{(f,1)}(b) = \varepsilon$
$\tau_{(f,1)}(f(a,a)) = \#^2$
...
$\tau_{(f,1)}(f(a,f(a,f(a,a)))) = \#^4$
...

$\tau_{(f,1)} : t \rightarrow \#^{|t|-1}$

State Equivalence

$\tau_\varepsilon : t \to \#^{|t|-1}$
$\tau_{(f,1)} : t \to \#^{|t|-1}$
$\varepsilon \simeq (f,1)$ So : $q_\varepsilon = q_{(f,1)}$
Similarly $u_2 = u_1 = \varepsilon$ and $(f,2) \simeq \varepsilon$

eSTW $Can(\tau)$

Axiom : $\#q_\varepsilon$
$q_\varepsilon(a) \to \varepsilon$
$q_\varepsilon(b) \to \varepsilon$
$q_\varepsilon(f(x_1,x_2)) \to \#\#q_\varepsilon q_\varepsilon$
minimal eSTW for τ_{count}

Outline

Learning Algo

Learning algorithm $Learn_{STW}(S)$

- Essentially the same as the construction algorithm
  - Input : a finite sample $S \subseteq \tau$
  - from each path p, compute S_p (approximation of τ_p)
  - $p \simeq p'$ if S_p does not contradict with $S_{p'}$
- $Learn_{STW}(S)$ answers in polynomial time
- For any STW τ, there exists a sample CS_τ of polynomial cardinality such that $Learn_{STW}(S) = Can(\tau)$ if $CS_\tau \subseteq S$

Consistancy Problem

Consistancy Issues

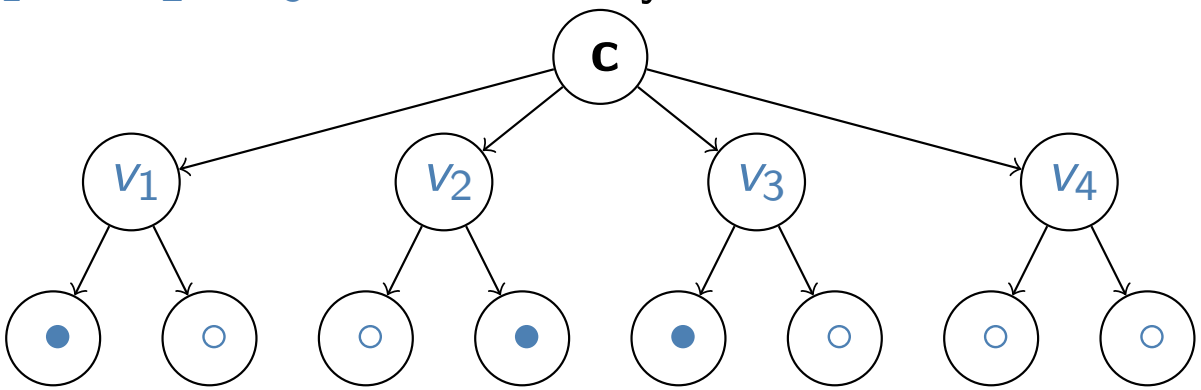
Checking if there exists an STW consistent with a given sample is NP-complete.

Idea : Encoding 1-in-3 SAT Problem
1-in-3 SAT : Variant of 3-SAT with exactly 1 litteral satisfied per clause
NP-Complete [Schaefer78]

Consistancy Problem

Example with a 4 literal formulas $(v_1 \vee \neg v_2 \vee v_3) \wedge (v_2 \vee \neg v_3 \vee v_4)$

$v_1 \vee \neg v_2 \vee v_3$ is encoded by:



$\rightarrow$ T

Consistancy Problem

The encoding of $(v_1 \vee \neg v_2 \vee v_3) \wedge (v_2 \vee \neg v_3 \vee v_4)$

$$
\begin{array}{lcll}
(v_1 \vee \neg v_2 \vee v_3) & : & c(v_1(\bullet,\circ), v_2(\circ,\bullet), v_3(\bullet,\circ), v_4(\circ,\circ)) & \rightarrow \quad \mathsf{T} \\
(v_2 \vee \neg v_3 \vee v_4) & : & c(v_1(\circ,\circ), v_2(\bullet,\circ), v_3(\circ,\bullet), v_4(\bullet,\circ)) & \rightarrow \quad \mathsf{T} \\
(v_1 \vee \neg v_1) & : & c(v_1(\bullet,\bullet), v_2(\circ,\circ), v_3(\circ,\circ), v_4(\circ,\circ)) & \rightarrow \quad \mathsf{T} \\
(v_2 \vee \neg v_2) & : & c(v_1(\circ,\circ), v_2(\bullet,\bullet), v_3(\circ,\circ), v_4(\circ,\circ)) & \rightarrow \quad \mathsf{T} \\
(v_3 \vee \neg v_3) & : & c(v_1(\circ,\circ), v_2(\circ,\circ), v_3(\bullet,\bullet), v_4(\circ,\circ)) & \rightarrow \quad \mathsf{T} \\
(v_4 \vee \neg v_4) & : & c(v_1(\circ,\circ), v_2(\circ,\circ), v_3(\circ,\circ), v_4(\bullet,\bullet)) & \rightarrow \quad \mathsf{T} \\
 & & c(v_1(\circ,\circ), v_2(\circ,\circ), v_3(\circ,\circ), v_4(\circ,\circ)) & \rightarrow \quad \varepsilon
\end{array}
$$

Idea: An eSTW that recognizes this sample encodes a solution of *SAT* one-in-three.

Consistancy Problem

Unique STW solution

Axiom : q

$q(c(x_1, \ldots, x_n)) \to q_1(x_1) \ldots q_n(x_n)$

$q_i(v_i(x_1, x_2)) \to q_i^L(x_1) q_i^R(x_2)$

$q_i^{L/R}(\circ) \to \varepsilon$

if v_i is true : $q_i^L(\bullet) \to T$ and $q_i^R(\bullet) \to \varepsilon$

if v_i is false : $q_i^L(\bullet) \to \varepsilon$ and $q_i^R(\bullet) \to T$

Idea : one state q_i per variable v_i, and

$v_i(\bullet, \circ)$ produces T iff v_i is true

$v_i(\circ, \bullet)$ produces T iff v_i is false

Learning Theorem

Learning Theorem

The Class of STW represented by eSTW is learnable from examples with polynomial time and data with abstain.

With abstain : the algorithm may not answer (non characteristic sample)

Conclusion - Future Works

Results

- Sequential Tree to Word Transducers :
  - A Myhill-Nerode Theorem
  - learnable in a Gold-like Model

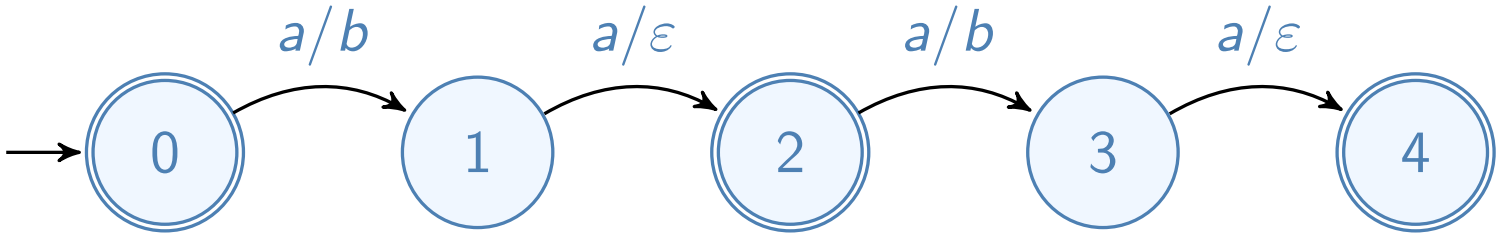
Future Works

- extension to non-ordered : good hope !
- extension to non-linear : bad hope ...
- toward a learning algorithm for MSO tree transduction ?

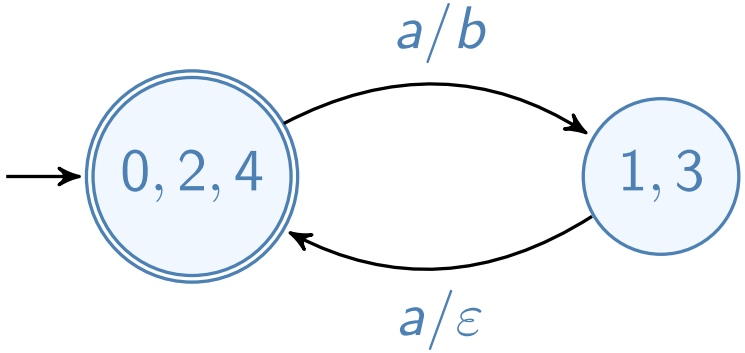
OSTIA [OncinaGarciaVidal93]

Example Input : $S = \{(\varepsilon, \varepsilon), (aa, b), (aaaa, bb)\}$

1. Align Input / output in an onward way, and build initial transducer



2. Perform State merging in an ordered way



S *characteristic* for $\tau \Rightarrow$ OSTIA(S) : canonical subsequential of τ