Applications of Tree Automata Theory
Lecture V: Theory of Tree Transducers

Andreas Maletti

Institute of Computer Science
Universität Leipzig, Germany

on leave from: Institute for Natural Language Processing
Universität Stuttgart, Germany

maletti@ims.uni-stuttgart.de

Yekaterinburg — August 25, 2014

Roadmap

1. Theory of Tree Automata
2. Parsing — Basics and Evaluation
3. Parsing — Advanced Topics
4. Machine Translation — Basics and Evaluation
5. Theory of Tree Transducers
6. Machine Translation — Advanced Topics

Always ask questions right away!

Motivation

From Extracted Rules to Tree Transducers

Syntax-based Machine Translation

Extracted rules

$$\textit{Yugoslav} \xrightarrow{q_Y} \textit{AlywgwslAfy} \qquad \textit{President} \xrightarrow{q_P} \textit{Alr\}ys}$$

$$\textit{Voislav} \xrightarrow{q_V} \textit{fwyslAf} \qquad \textit{for} \xrightarrow{q_f} \textit{En}$$

$$\textit{Serbia} \xrightarrow{q_S} \textit{SrbyA}$$

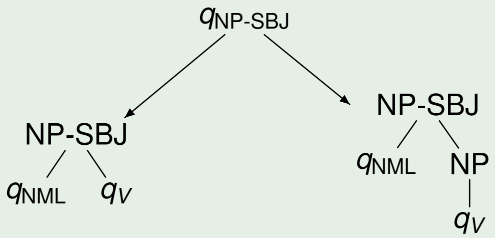
$$\mathrm{NML}(q_Y, q_P) \xrightarrow{q_{\mathrm{NML}}} \mathrm{NP}(q_P, q_Y) \qquad \mathrm{NP}(q_S) \xrightarrow{q_{\mathrm{NP}}} \mathrm{NP}(q_S)$$

$$\mathrm{PP}(q_f, q_{\mathrm{NP}}) \xrightarrow{q_{\mathrm{PP}}} \mathrm{PP}(q_f, q_{\mathrm{NP}})$$

$$\text{NP-SBJ}(q_{\mathrm{NML}}, q_V) \xrightarrow{q_{\text{NP-SBJ}}} \text{NP-SBJ}(q_{\mathrm{NML}}, \mathrm{NP}(q_V))$$

From Automata to Transducers

Graphical representation



corresponds to two productions

$$q_{\text{NP-SBJ}} \longrightarrow \text{NP-SBJ}(q_{\text{NML}}, q_V)$$
$$q_{\text{NP-SBJ}} \longrightarrow \text{NP-SBJ}(q_{\text{NML}}, \text{NP}(q_V))$$

From Automata to Transducers

General idea

Synchronous grammars are essentially two grammars
over the same nonterminals whose productions are paired

Convention

same nonterminals are synchronized (or linked)
and develop at the same time

From Automata to Transducers

Approach

- join two productions $q_1 \to r_1$ and $q_2 \to r_2$ to $(q_1, q_2) \to (r_1, r_2)$

From Automata to Transducers

Approach

- join two productions $q_1 \to r_1$ and $q_2 \to r_2$ to $(q_1, q_2) \to (r_1, r_2)$
- demand $q_1 = q = q_2$ for simplicity and write $r_1 \stackrel{q}{\text{---}} r_2$

From Automata to Transducers

Approach

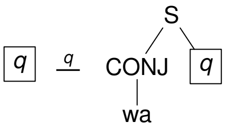
- join two productions $q_1 \to r_1$ and $q_2 \to r_2$ to $(q_1, q_2) \to (r_1, r_2)$
- demand $q_1 = q = q_2$ for simplicity and write $r_1 \stackrel{q}{\text{---}} r_2$
- paired productions develop input and output tree at the same time

From Automata to Transducers

$\boxed{q}$ $\boxed{q}$

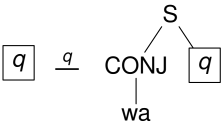
Used rule:

Next rule:

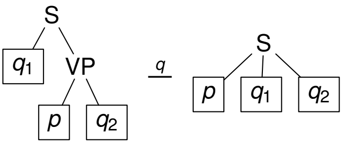


From Automata to Transducers

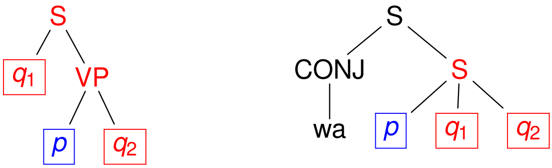
Used rule:



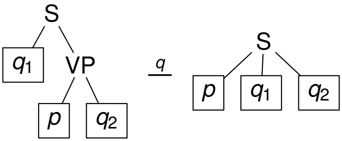
Next rule:



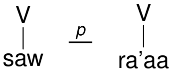
From Automata to Transducers



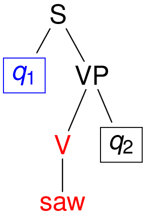
Used rule:

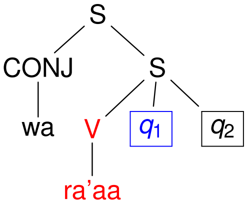


Next rule:

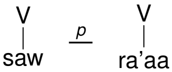


From Automata to Transducers

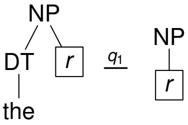




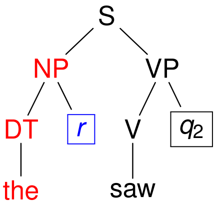
Used rule:

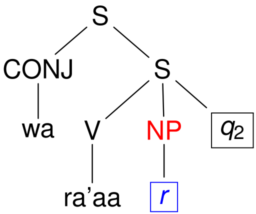


Next rule:

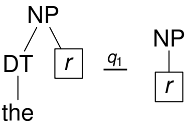


From Automata to Transducers

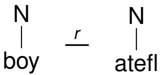




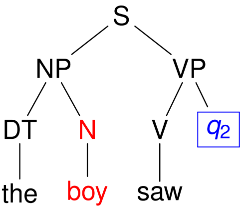
Used rule:

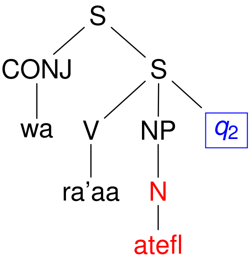


Next rule:

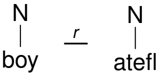


From Automata to Transducers

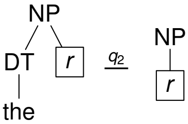




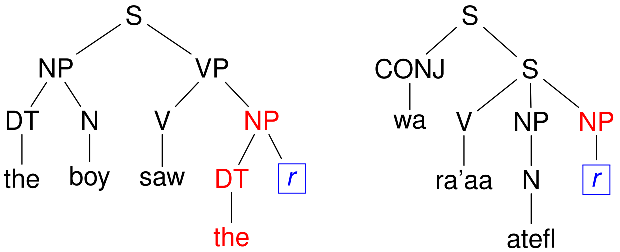
Used rule:



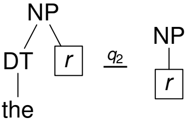
Next rule:



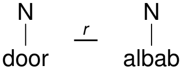
From Automata to Transducers



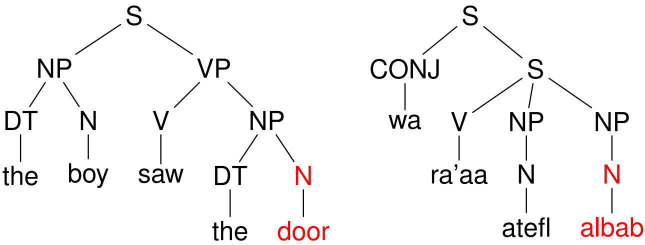
Used rule:



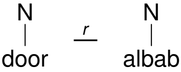
Next rule:



From Automata to Transducers



Used rule:



Next rule:

From Automata to Transducers

Remarks

- synchronization breaks almost all existing constructions
  (e.g., the normalization construction)

$\rightarrow$ the basic grammar model very important

From Automata to Transducers

Remarks

- synchronization breaks almost all existing constructions (e.g., the normalization construction)

$\rightarrow$ the basic grammar model very important

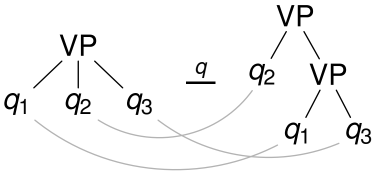
Major models

1. linear top-down tree transducer
   - input-side grammar: TA   normalized RTG
   - output-side grammar: RTG
2. linear extended top-down tree transducer
   - input-side grammar: RTG
   - output-side grammar: RTG

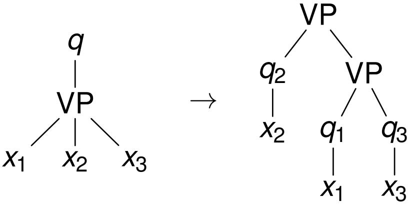
Top-down Tree Transducers

Rule Transformation

Synchronous grammar rule:



Top-down tree transducer rule:



Top-down Tree Transducer

Notation

- variables $X = \{x_0, x_1, \ldots,\} = \{x_i \mid i \in \mathbb{N}\}$

Top-down Tree Transducer

Notation

- variables $X = \{x_0, x_1, \dots,\} = \{x_i \mid i \in \mathbb{N}\}$
- unary top-concatenation set Q; $T \subseteq T_\Sigma(V)$

$$Q(T) = \{q(t) \mid q \in Q,\ t \in T\}$$

Top-down Tree Transducer

Notation

- variables $X = \{x_0, x_1, \dots, \} = \{x_i \mid i \in \mathbb{N}\}$
- unary top-concatenation set Q; $T \subseteq T_\Sigma(V)$

$$Q(T) = \{q(t) \mid q \in Q,\ t \in T\}$$

- $\mathrm{var}(t) = \{x \in X \mid x \text{ occurs in } t\}$ $t \in T_\Sigma(X)$

Top-down Tree Transducer

Notation

- variables $X = \{x_0, x_1, \dots, \} = \{x_i \mid i \in \mathbb{N}\}$
- unary top-concatenation set Q; $T \subseteq T_\Sigma(V)$

$$Q(T) = \{q(t) \mid q \in Q,\, t \in T\}$$

- $\text{var}(t) = \{x \in X \mid x \text{ occurs in } t\}$ $t \in T_\Sigma(X)$
- $t \in T_\Sigma(X)$ linear if each $x \in \text{var}(t)$ occurs at most once

Top-down Tree Transducer

Definition (Thatcher, 1970)

A top-down tree transducer is a tuple $(Q, \Sigma, \Delta, I, R)$

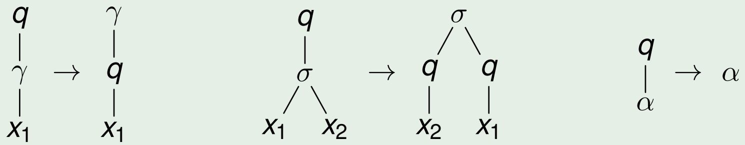
- alphabet Q — states
- alphabets Σ and Δ — input/output symbols
- $I \subseteq Q$ — initial states
- finite set $R \subseteq Q(\Sigma(X)) \times T_\Delta(Q(X))$ — rules
  - $\text{var}(r) \subseteq \text{var}(\ell)$ — for all $(\ell, r) \in R$
  - ℓ is linear — for all $(\ell, r) \in R$

Top-down Tree Transducer

Example

Mirror-image top-down tree transducer $(Q, \Sigma, \Sigma, Q, R)$ with

- $Q = \{q\}$
- $\Sigma = \{\sigma, \gamma, \alpha\}$
- the following rules in R



Top-down Tree Transducer

Definition (Derivation)

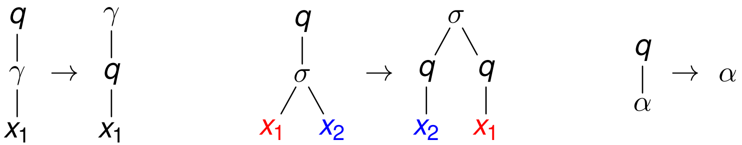
Sentential forms $\xi, \zeta \in T_\Delta(Q(T_\Sigma))$

$$\boxed{\xi \Rightarrow_M \zeta}$$

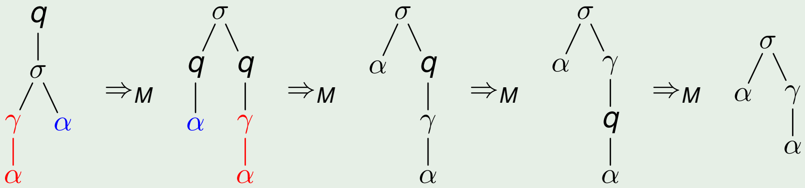
if there exist $\ell \to r \in R$, position $w \in \text{pos}(\xi)$, substitution $\theta \colon X \to T_\Sigma$

- $\xi = \xi[\ell\theta]_w$
- $\zeta = \xi[r\theta]_w$

Top-down Tree Transducer



Example



Top-down Tree Transducer

Definition

$$M = \{\langle t, u\rangle \in T_\Sigma \times T_\Delta \mid \exists q \in I\colon q(t) \Rightarrow_M^* u\}$$

Top-down Tree Transducer

Definition

$$M = \{\langle t, u\rangle \in T_\Sigma \times T_\Delta \mid \exists q \in I \colon q(t) \Rightarrow_M^* u\}$$

Example

Top-down tree transducer N with

$$\{\langle \sigma(t, u), \sigma(u, t)\rangle \mid t, u \in T_{\{\gamma,\alpha\}}\} \subseteq N$$

$$q(\gamma(x_1)) \to \gamma(q(x_1)) \qquad q(\sigma(x_1, x_2)) \to \sigma(q(x_2), q(x_1)) \qquad q(\alpha) \to \alpha$$

Top-down Tree Transducer

Definition (Syntactic restrictions)

Transducer $(Q, \Sigma, \Delta, I, R)$ is

- linear if r is linear — for every $\ell \to r \in R$
- nondeleting if $\text{var}(r) = \text{var}(\ell)$ — for every $\ell \to r \in R$
- strict if $r \notin Q(X)$ — for every $\ell \to r \in R$

Top-down Tree Transducer

Definition (Syntactic restrictions)

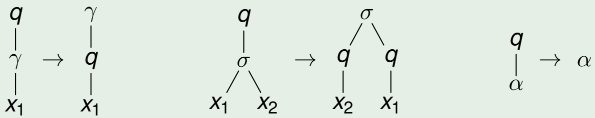
Transducer $(Q, \Sigma, \Delta, I, R)$ is

- linear if r is linear — for every $\ell \to r \in R$
- nondeleting if $\text{var}(r) = \text{var}(\ell)$ — for every $\ell \to r \in R$
- strict if $r \notin Q(X)$ — for every $\ell \to r \in R$

Example

Mirror-image transducer is linear, nondeleting, and strict

(Ins-TOP)



Top-down Tree Transducer

Properties [ENGELFRIET, 1975]

- T1 "Copying of an input tree and
  and processing the copies differently"
- T2 "Cannot inspect deleted input tree"

Top-down Tree Transducer

Properties [ENGELFRIET, 1975]

- T1 "Copying of an input tree and
  and processing the copies differently"
- T2 "Cannot inspect deleted input tree"

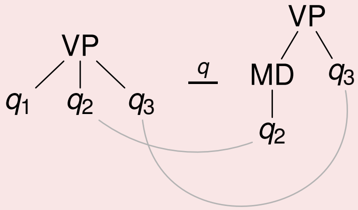
Remark

T2 has been addressed
$\rightsquigarrow$ top-down tree transducers with regular look-ahead
[ENGELFRIET, 1977]

Top-down Tree Transducer

Regular look-ahead

Can be simulated by allowing un-linked nonterminals on the input side



- these develop without effect on the output
- can generate any regular tree language

Top-down Tree Transducer

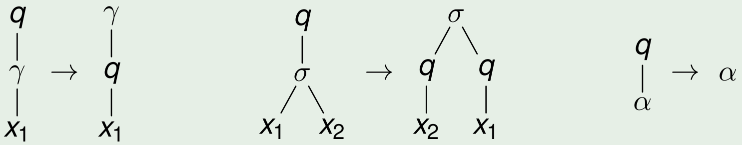
Definition (Comp)

$\tau \subseteq T_\Sigma \times T_\Delta$ and $\tau' \subseteq T_\Delta \times T_\Gamma$

$$\boxed{\tau \,;\, \tau' = \{(s, u) \mid \exists t \in T_\Delta : (s, t) \in \tau, (t, u) \in \tau'\}}$$

Example (Double mirror-image)

$N \,;\, N = \mathrm{id}$



Top-down Tree Transducer

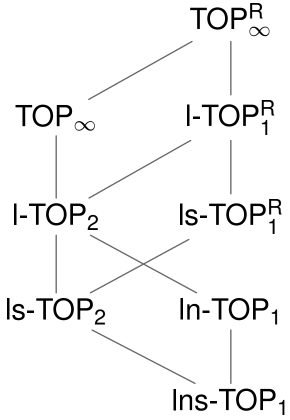
Notation

- TOP = class of tree transformations computable by top-down tree transducers
- $\mathrm{TOP}^{\mathrm{R}}$ = class of ... transducers with regular look-ahead
- x-$\mathrm{TOP}^{(\mathrm{R})}$ = class of ... transducers with properties x

Example

ln-TOP = class of tree transformations computable by linear and nondeleting top-down tree transducers

Top-down Tree Transducer

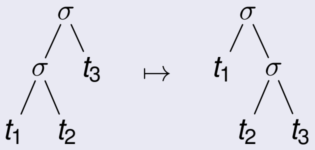


composition closure indicated in subscript

Top-down Tree Transducer

Rotations

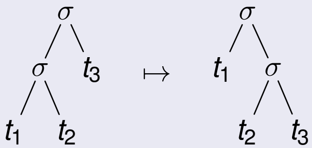
$\textsc{Rot} = \{\langle \sigma(\sigma(t_1, t_2), t_3),\ \sigma(t_1, \sigma(t_2, t_3)) \rangle \mid t_1, t_2, t_3 \in T_\Sigma\}$



Top-down Tree Transducer

Rotations

$\textsc{Rot} = \{\langle \sigma(\sigma(t_1, t_2), t_3),\ \sigma(t_1, \sigma(t_2, t_3))\rangle \mid t_1, t_2, t_3 \in T_\Sigma\}$



Preservation of regularity (Pres)

Given $\tau \subseteq T_\Sigma \times T_\Delta$ and $L \subseteq T_\Sigma$ regular, is $\tau(L)$ regular?

$$\boxed{\tau(L) = \{u \mid \exists t \in L \colon (t, u) \in \tau\}}$$

Top-down Tree Transducer

Model \ Criterion	Rot	Sym	Pres	Pres^{-1}	Comp
Ins-TOP	✗	✗	✓	✓	✓
ln-TOP	✗	✗	✓	✓	✓
ls-TOP	✗	✗	✓	✓	✗$_2$
l-TOP	✗	✗	✓	✓	✗$_2$
ls-TOP^R	✗	✗	✓	✓	✓
l-TOP^R	✗	✗	✓	✓	✓
TOP	✓	✗	✗	✓	✗$_\infty$
TOP^R	✓	✗	✗	✓	✗$_\infty$

(Sym = symmetric)

Extended Top-down Tree Transducers

Extended Top-down Tree Transducer

Definition (GRAEHL et al., 2009)

A top-down tree transducer is a tuple $(Q, \Sigma, \Delta, I, R)$

- finite set Q — states
- alphabets Σ and Δ — input/output symbols
- $I \subseteq Q$ — initial states
- finite set $R \subseteq Q(\Sigma(X)) \times T_\Delta(Q(X))$ — rules
  - $\text{var}(r) \subseteq \text{var}(\ell)$ — for all $(\ell, r) \in R$
  - ℓ is linear — for all $(\ell, r) \in R$

Extended Top-down Tree Transducer

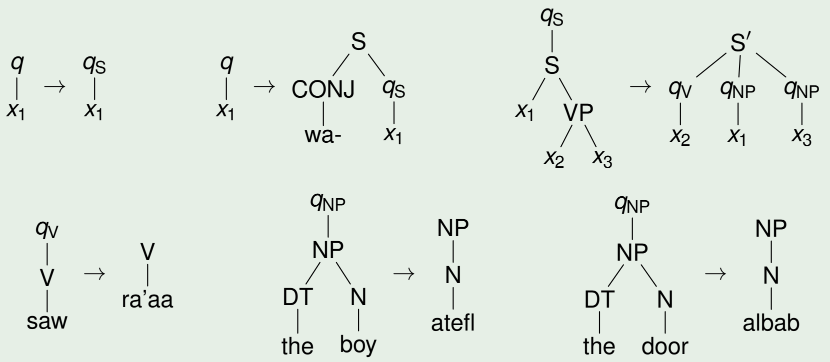
Definition (GRAEHL et al., 2009)

An extended top-down tree transducer is a tuple $(Q, \Sigma, \Delta, I, R)$

- finite set Q — states
- alphabets Σ and Δ — input/output symbols
- $I \subseteq Q$ — initial states
- finite set $R \subseteq Q(T_\Sigma(X)) \times T_\Delta(Q(X))$ — rules
  - $\text{var}(r) \subseteq \text{var}(\ell)$ — for all $(\ell, r) \in R$
  - ℓ is linear — for all $(\ell, r) \in R$

Extended Top-down Tree Transducer

Example



Extended Top-down Tree Transducer

Definition (same as before)

Sentential forms $\xi, \zeta \in T_\Delta(Q(T_\Sigma))$

$$\boxed{\xi \Rightarrow_M \zeta}$$

if there exist $\ell \to r \in R$, position $w \in \text{pos}(\xi)$, substitution $\theta \colon X \to T_\Sigma$

- $\xi = \xi[\ell\theta]_w$
- $\zeta = \xi[r\theta]_w$

Extended Top-down Tree Transducer

Definition (same as before)

Sentential forms $\xi, \zeta \in T_\Delta(Q(T_\Sigma))$

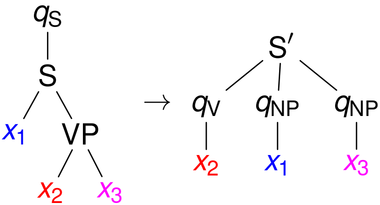
$$\boxed{\xi \Rightarrow_M \zeta}$$

if there exist $\ell \to r \in R$, position $w \in \mathrm{pos}(\xi)$, substitution $\theta \colon X \to T_\Sigma$

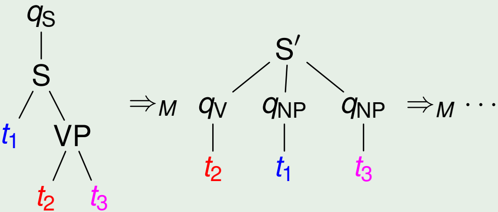
- $\xi = \xi[\ell\theta]_w$
- $\zeta = \xi[r\theta]_w$

$$M = \{\langle t, u\rangle \in T_\Sigma \times T_\Delta \mid \exists q \in I \colon q(t) \Rightarrow_M^* u\}$$

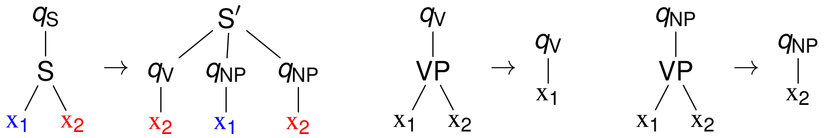
Extended Top-down Tree Transducer



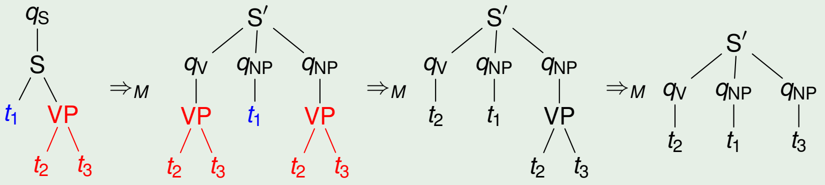
Application of the rule



Extended Top-down Tree Transducer



Simulation by top-down tree transducer



Extended Top-down Tree Transducer

Definition (Syntactic restrictions)

Extended top-down tree transducer $(Q, \Sigma, \Delta, I, R)$ is

- linear, nondeleting, strict as before

Extended Top-down Tree Transducer

Definition (Syntactic restrictions)

Extended top-down tree transducer $(Q, \Sigma, \Delta, I, R)$ is

- linear, nondeleting, strict as before
- ε-free if $\ell \notin Q(X)$ for every $\ell \to r \in R$

Extended Top-down Tree Transducer

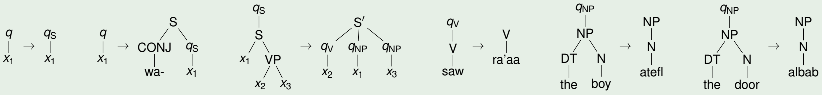
Definition (Syntactic restrictions)

Extended top-down tree transducer $(Q, \Sigma, \Delta, I, R)$ is

- linear, nondeleting, strict as before
- ε-free if $\ell \notin Q(X)$ for every $\ell \to r \in R$

Example

Our example transducer is linear, nondeleting, ~~strict~~, and ~~ε-free~~



Extended Top-down Tree Transducer

Properties [$\sim$ et al., 2009]

- X1 Finite look-ahead
- X2 Deep attachment of variables
- X3 Infinitely many outputs for one input

Extended Top-down Tree Transducer

Properties [$\sim$ et al., 2009]

X1 Finite look-ahead

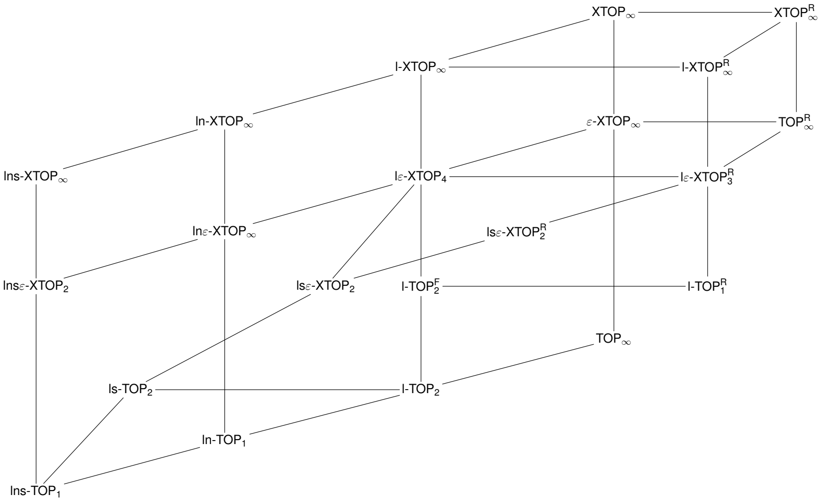
X2 Deep attachment of variables

X3 Infinitely many outputs for one input

Remarks

- T1 and T2 still apply
- XTOP = class notation

Extended Top-down Tree Transducer



composition closure indicated in subscript

Extended Top-down Tree Transducer

Model \ Criterion	Rot	Sym	Pres	Pres^{-1}	Comp
ln-TOP	✗	✗	✓	✓	✓
l-TOP	✗	✗	✓	✓	✗$_2$
l-TOP$^{\mathrm{R}}$	✗	✗	✓	✓	✓
TOP$^{\mathrm{R}}$	✓	✗	✗	✓	✗$_\infty$
lnsε-XTOP	✓	✓	✓	✓	✗$_2$
lns-XTOP	✓	✗	✓	✓	✗$_\infty$
lsε-XTOP$^{\mathrm{(R)}}$	✓	✗	✓	✓	✗$_2$
lε-XTOP	✓	✗	✓	✓	✗$_4$
lε-XTOP$^{\mathrm{R}}$	✓	✗	✓	✓	✗$_3$
(s)l-XTOP$^{\mathrm{(R)}}$	✓	✗	✓	✓	✗$_\infty$
XTOP	✓	✗	✗	✓	✗$_\infty$
XTOP$^{\mathrm{R}}$	✓	✗	✗	✓	✗$_\infty$

Extended Multi Bottom-up Tree Transducers

Extended Multi Bottom-up Tree Transducer

Definition (ENGELFRIET et al., 2009)

An extended multi bottom-up tree transducer $(Q, \Sigma, \Delta, F, R)$

- alphabet Q — states
- alphabets Σ and Δ — input/output symbols
- $F \subseteq Q$ — final states
- finite set $R \subseteq T_{\Sigma \cup Q}(X) \times T_{\Delta \cup Q}(X)$ — rules
  - $\text{var}(r) \subseteq \text{var}(\ell)$ — for all $(\ell, r) \in R$
  - ℓ is linear — for all $(\ell, r) \in R$
  - $q \in Q$ occur in ℓ only directly above elements of X
  - $q \in Q$ occurs in r only at the root

Extended Multi Bottom-up Tree Transducer

Definition (ENGELFRIET et al., 2009)

An extended multi bottom-up tree transducer $(Q, \Sigma, \Delta, F, R)$

- alphabet Q — states
- alphabets Σ and Δ — input/output symbols
- $F \subseteq Q$ — final states
- finite set $R \subseteq T_{\Sigma \cup Q}(X) \times T_{\Delta \cup Q}(X)$ — rules
  - $\text{var}(r) \subseteq \text{var}(\ell)$ — for all $(\ell, r) \in R$
  - ℓ is linear — for all $(\ell, r) \in R$
  - $q \in Q$ occur in ℓ only directly above elements of X
  - $q \in Q$ occurs in r only at the root

Properties

linear, nondeleting, strict, ε-free as before

Extended Multi Bottom-up Tree Transducer

Example (Duplication)

Extended multi bottom-up tree transducer $(Q, \Sigma, \Sigma, \{f\}, R)$

- $Q = \{q, f\}$ and $\Sigma = \{\sigma, a, b, e\}$
- R contains exactly:

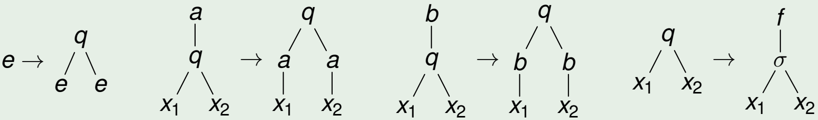
$$e \to q(e, e) \qquad a(q(x_1, x_2)) \to q(a(x_1), a(x_2)) \qquad b(q(x_1, x_2)) \to q(b(x_1), b(x_2)) \qquad q(x_1, x_2) \to f(\sigma(x_1, x_2))$$

Extended Multi Bottom-up Tree Transducer

Example (Duplication)

Extended multi bottom-up tree transducer $(Q, \Sigma, \Sigma, \{f\}, R)$

- $Q = \{q, f\}$ and $\Sigma = \{\sigma, a, b, e\}$
- R contains exactly:

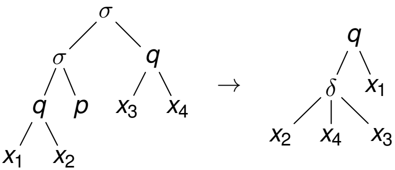


Properties

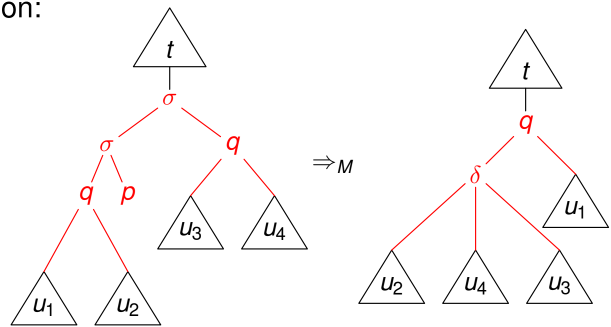
linear, nondeleting, strict, and ~~ε-free~~

Extended Multi Bottom-up Tree Transducer

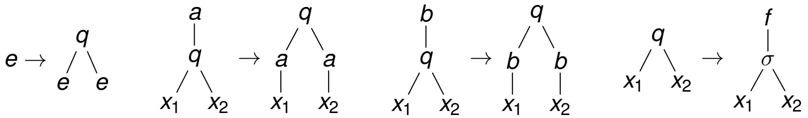
Rule:



Derivation:

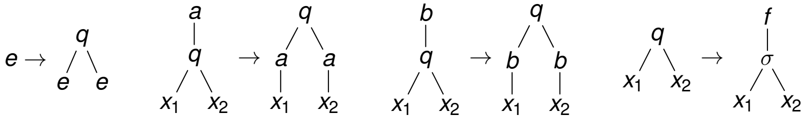


Extended Multi Bottom-up Tree Transducer

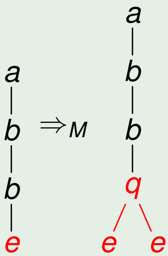


Derivation

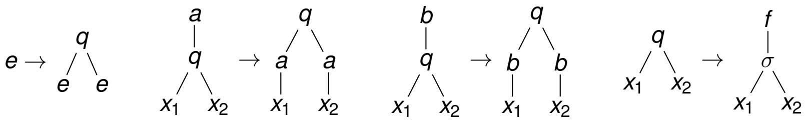
Extended Multi Bottom-up Tree Transducer



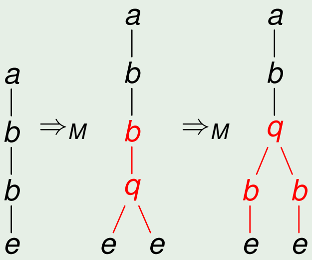
Derivation



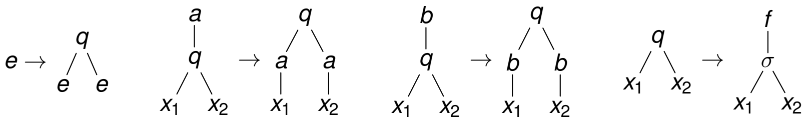
Extended Multi Bottom-up Tree Transducer



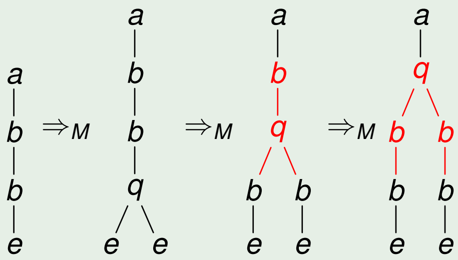
Derivation



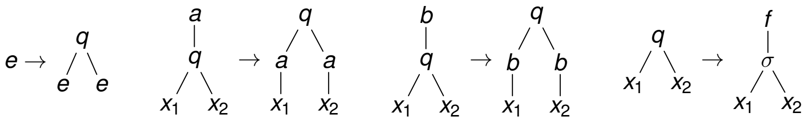
Extended Multi Bottom-up Tree Transducer



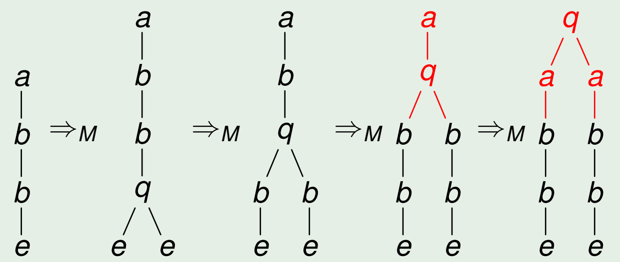
Derivation



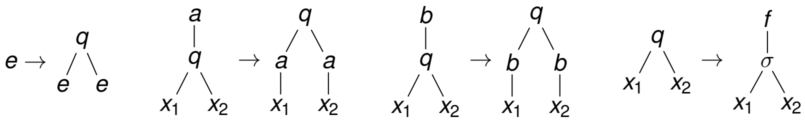
Extended Multi Bottom-up Tree Transducer



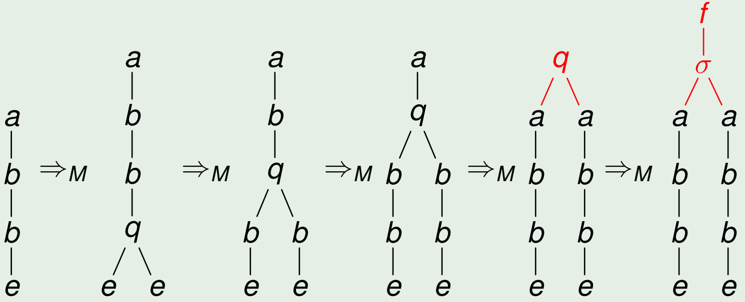
Derivation



Extended Multi Bottom-up Tree Transducer



Derivation



Extended Multi Bottom-up Tree Transducer

Definition

$$\tau_M = \{(t, u) \in T_\Sigma \times T_\Delta \mid \exists q \in F \colon t \Rightarrow^*_M q(u)\}$$

Extended Multi Bottom-up Tree Transducer

Definition

$$\tau_M = \{(t, u) \in T_\Sigma \times T_\Delta \mid \exists q \in F \colon t \Rightarrow^*_M q(u)\}$$

Example (Duplication)

It computes $\{(t, \sigma(t, t)) \mid t \in T_\Sigma\}$

Its image is not a regular tree language

Extended Multi Bottom-up Tree Transducer

Definition

Extended multi bottom-up tree transducer $(Q, \Sigma, \Delta, F, R)$ is

- extended bottom-up tree transducer (XBOT) if $R \subseteq T_\Sigma(Q(X)) \times Q(T_\Delta(X))$
- multi bottom-up tree transducer (MBOT) if $\ell \in \Sigma(T_Q(X))$ for all $\ell \to r \in R$
- bottom-up tree transducer (BOT) if both conditions hold

Extended Multi Bottom-up Tree Transducer

Definition

Extended multi bottom-up tree transducer $(Q, \Sigma, \Delta, F, R)$ is

- extended bottom-up tree transducer (XBOT)
  if $R \subseteq T_\Sigma(Q(X)) \times Q(T_\Delta(X))$
- multi bottom-up tree transducer (MBOT)
  if $\ell \in \Sigma(T_Q(X))$ for all $\ell \to r \in R$
- bottom-up tree transducer (BOT) if both conditions hold

Example (Duplication)

$$e \to q(e, e) \qquad a(q(x_1, x_2)) \to q(a(x_1), a(x_2)) \qquad b(q(x_1, x_2)) \to q(b(x_1), b(x_2)) \qquad q(x_1, x_2) \to f(\sigma(x_1, x_2))$$

Extended Multi Bottom-up Tree Transducer

Theorem [ENGELFRIET et al., 2009]

$$\text{l-XTOP}^{\text{R}} = \text{l-XBOT}$$

Proof.

Standard construction trading input-deletion for output-deletion
see l-TOP $\subseteq$ l-BOT by [ENGELFRIET '75] $\square$

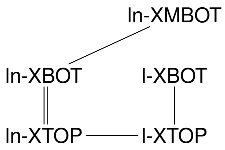
Extended Multi Bottom-up Tree Transducer

Theorem [ENGELFRIET et al., 2009]

$$\text{l-XTOP}^{\text{R}} = \text{l-XBOT}$$

Proof.

Standard construction trading input-deletion for output-deletion see l-TOP $\subseteq$ l-BOT by [ENGELFRIET '75] $\square$



Extended Multi Bottom-up Tree Transducer

Theorem [Engelfriet et al., 2009]

$$\text{XMBOT} = \text{n-XMBOT}$$

Extended Multi Bottom-up Tree Transducer

Theorem [Engelfriet et al., 2009]

$$\text{XMBOT} = \text{n-XMBOT}$$

Proof.

- guess subtrees that will be deleted
- process them using look-ahead $\square$

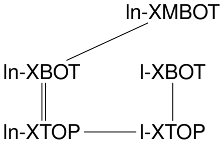
Extended Multi Bottom-up Tree Transducer

Theorem [Engelfriet et al., 2009]

$$\text{XMBOT} = \text{n-XMBOT}$$

Proof.

- guess subtrees that will be deleted
- process them using look-ahead □



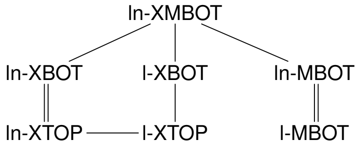
Extended Multi Bottom-up Tree Transducer

Theorem [ENGELFRIET et al., 2009]

$$\text{XMBOT} = \text{n-XMBOT}$$

Proof.

- guess subtrees that will be deleted
- process them using look-ahead $\square$



Extended Multi Bottom-up Tree Transducer

Theorem [Engelfriet et al., 2009]

$$\varepsilon\text{-XMBOT} = \text{MBOT}$$

Extended Multi Bottom-up Tree Transducer

Theorem [ENGELFRIET et al., 2009]

$$\varepsilon\text{-XMBOT} = \text{MBOT}$$

Proof.

- decompose large left-hand sides using “multi”-states
- attach finite effect of ε-rules $\square$

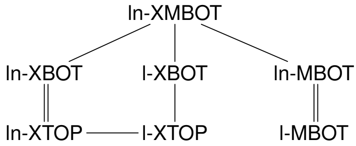
Extended Multi Bottom-up Tree Transducer

Theorem [Engelfriet et al., 2009]

$$\varepsilon\text{-XMBOT} = \text{MBOT}$$

Proof.

- decompose large left-hand sides using “multi”-states
- attach finite effect of ε-rules $\square$



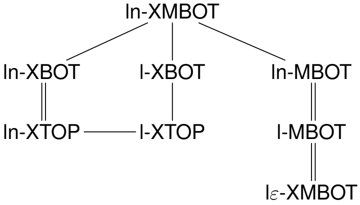
Extended Multi Bottom-up Tree Transducer

Theorem [Engelfriet et al., 2009]

$$\varepsilon\text{-XMBOT} = \text{MBOT}$$

Proof.

- decompose large left-hand sides using “multi”-states
- attach finite effect of ε-rules $\square$



Extended Multi Bottom-up Tree Transducer

Definition

tree transformation τ sensible
if $|\text{pos}(u)| \in \mathcal{O}(|\text{pos}(t)|)$ for all $(t, u) \in \tau$ (linear i-o size relation)

Extended Multi Bottom-up Tree Transducer

Definition

tree transformation τ sensible
if $|\text{pos}(u)| \in \mathcal{O}(|\text{pos}(t)|)$ for all $(t, u) \in \tau$ (linear i-o size relation)

Theorem [$\sim$, 2012]

$$\text{sensible XTOP} \subseteq \text{ln-MBOT}$$

Extended Multi Bottom-up Tree Transducer

Definition

tree transformation τ sensible
if $|\text{pos}(u)| \in \mathcal{O}(|\text{pos}(t)|)$ for all $(t, u) \in \tau$ (linear i-o size relation)

Theorem [$\sim$, 2012]

sensible XTOP $\subseteq$ ln-MBOT

Proof.

- use construction of [Engelfriet, Maneth, 2003]
- obtain finitely copying ε-XTOP
- apply [Engelfriet et al., 2009] to obtain lε-XMBOT
- previous theorems yield ln-MBOT $\square$

Extended Multi Bottom-up Tree Transducer

Corollary

All SMT-relevant extended top-down tree transducers can be simulated by linear and nondeleting extended multi bottom-up tree transducers

Extended Multi Bottom-up Tree Transducer

Theorem

$$\text{ln-MBOT} \not\subseteq \text{XTOP}^{\text{R}}$$

Extended Multi Bottom-up Tree Transducer

Theorem

$$\text{ln-MBOT} \not\subseteq \text{XTOP}^{\text{R}}$$

Theorem [Gildea, 2012]

$$\text{yd}_{\text{out}}(\text{ln-MBOT}) = \text{LCFRS}$$

Summary

Model \ Criterion	Rot	Sym	Pres	Pres^{-1}	Comp
ln-TOP	✗	✗	✓	✓	✓
l-TOP	✗	✗	✓	✓	$✗_2$
l-TOP^{R}	✗	✗	✓	✓	✓
TOP^{R}	✓	✗	✗	✓	$✗_\infty$
lnsε-XTOP	✓	✓	✓	✓	$✗_2$
lns-XTOP	✓	✗	✓	✓	$✗_\infty$
lsε-XTOP$^{(R)}$	✓	✗	✓	✓	$✗_2$
lε-XTOP	✓	✗	✓	✓	$✗_4$
lε-XTOP^{R}	✓	✗	✓	✓	$✗_3$
(s)l-XTOP$^{(R)}$	✓	✗	✓	✓	$✗_\infty$
XTOP$^{(R)}$	✓	✗	✗	✓	$✗_\infty$
l(n)-XMBOT	✓	✗	✗	✓	✓
XMBOT	✓	✗	✗	✓	$✗_\infty$
reg.-preserving l-XMBOT	✓	✗	✓	✓	✓
invertable l-XMBOT	✓	✓	✓	✓	✓

Literature

Selected references

- ARNOLD, DAUCHET: *Morphismes et Bimorphismes d'Arbres*
  Theoret. Comput. Sci. 20, 1982
- ENGELFRIET: *Bottom-up and Top-down Tree Transformations — A Comparison.* Math. Systems Theory 9, 1975
- ENGELFRIET, MANETH: *Macro Tree Translations of Linear Size Increase are MSO Definable.* SIAM J. Comput. 32, 2003
- ENGELFRIET, LILIN, ∼: *Extended Multi Bottom-up Tree Transducers — Composition and Decomposition.* Acta Inf. 46, 2009
- ROUNDS: *Mappings and Grammars on Trees*
  Math. Systems Theory 4, 1970
- THATCHER: *Generalized^2 Sequential Machine Maps*
  J. Comput. System Sci. 4, 1970