Input Products for Weighted Extended Top-down Tree Transducers

Andreas Maletti

Universitat Rovira i Virgili
Tarragona, Spain

andreas.maletti@urv.cat

London, ON — August 17, 2010

Machine translation

Schema

Input $\longrightarrow$ Machine translation system $\longrightarrow$ Output

Question

How does the system handle input sentences containing “system”?

Some answer

- take regular language $L = * \text{ system } *$
- turn into a regular tree language
- use forward application

Machine translation

Schema

Input $\longrightarrow$ WXTT $\longrightarrow$ Output

Question

How does the system handle input sentences containing “system”?

Some answer

- take regular language $L = * \text{ system } *$
- turn into a regular tree language
- use forward application

Machine translation

Schema

Input $\longrightarrow$ **WXTT** $\longrightarrow$ Output

Question

How does the system handle input sentences containing “system”?

Some answer

- take regular language $L = *\ \text{system}\ *$
- turn into a regular tree language
- use forward application

Machine translation

Schema

Input $\longrightarrow$ WXTT $\longrightarrow$ Output

Question

How does the system handle input sentences containing “system”?

Some answer

- take regular language $L = *\ \text{system}\ *$
- turn into a regular tree language
- use forward application

Machine translation (cont'd)

Question

How does the system handle input sentences containing "system"?

Forward application

- Problem: we obtain only output trees

$\Rightarrow$ not informative enough

Another answer

- regular language, regular tree language as before
- input product restricts input to regular tree language

$\Rightarrow$ retains full translation information

Machine translation (cont'd)

Question

How does the system handle input sentences containing "system"?

Forward application

- Problem: we obtain only output trees

$\Rightarrow$ not informative enough

Another answer

- regular language, regular tree language as before
- input product restricts input to regular tree language

$\Rightarrow$ retains full translation information

Input product

Applications

- parsing (one-sided, both-sided)
- translation
- forward application (input product + domain projection)
- regular look-ahead
- computation of interesting parameters (inside/outside weights)

Input product

Applications

- parsing (one-sided, both-sided)
- translation
- forward application (input product + domain projection)
- regular look-ahead
- computation of interesting parameters (inside/outside weights)

Input product

Applications

- parsing (one-sided, both-sided)
- translation
- forward application (input product + domain projection)
- regular look-ahead
- computation of interesting parameters (inside/outside weights)

Contents

Weight structure

Definition

Commutative semiring $(C, +, \cdot, 0, 1)$ if

- $(C, +, 0)$ and $(C, \cdot, 1)$ commutative monoids
- $\cdot$ distributes over finite (incl. empty) sums

Idempotent if $c + c = c$

Example

- BOOLEAN semiring $(\{0, 1\}, \max, \min, 0, 1)$ (idempotent)
- Semiring $(\mathbb{N}, +, \cdot, 0, 1)$ of natural numbers
- Tropical semiring $(\mathbb{N} \cup \{\infty\}, \min, +, \infty, 0)$ (idempotent)
- Any field, ring, etc.

Weight structure

Definition

Commutative semiring $(C, +, \cdot, 0, 1)$ if

- $(C, +, 0)$ and $(C, \cdot, 1)$ commutative monoids
- $\cdot$ distributes over finite (incl. empty) sums

Idempotent if $c + c = c$

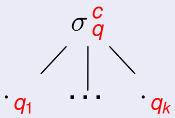
Example

- BOOLEAN semiring $(\{0, 1\}, \max, \min, 0, 1)$ (idempotent)
- Semiring $(\mathbb{N}, +, \cdot, 0, 1)$ of natural numbers
- Tropical semiring $(\mathbb{N} \cup \{\infty\}, \min, +, \infty, 0)$ (idempotent)
- Any field, ring, etc.

Weighted tree automaton

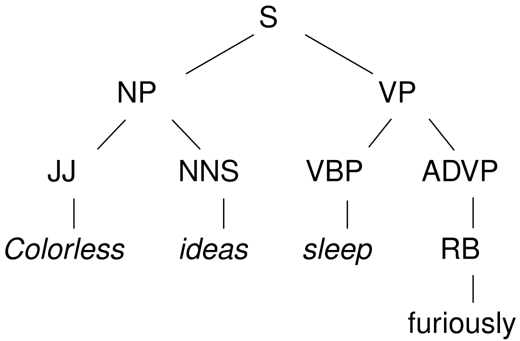
Definition (Berstel, Reutenauer 1982)

Weighted tree automaton (WTA) $A = (Q, \Sigma, I, \delta)$ with rules

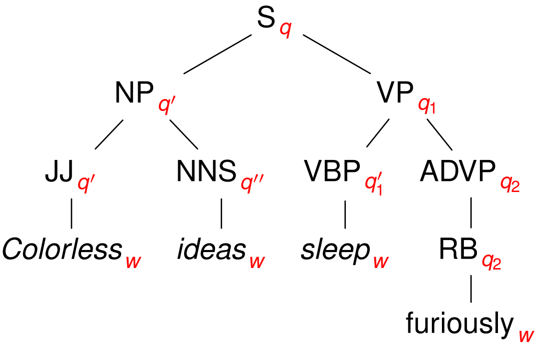


- $q, q_1, \ldots, q_k \in Q$ are states
- $c \in C$ is a weight
- $\sigma \in \Sigma_k$ is a k-ary input symbol

Run

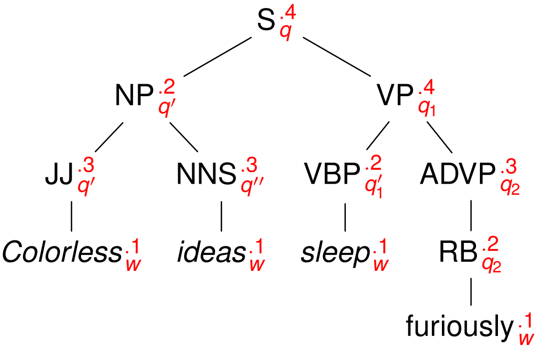


Run



S q
NP q′
VP q1
JJ q′
NNS q′′
VBP q′1
ADVP q2
Colorless w
ideas w
sleep w
RB q2
furiously w

Run

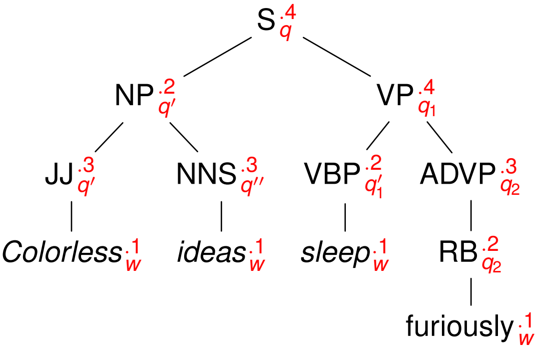


Definition

Weight wt(r) of a run r

= product of its weights

Run



Example (Weight of the run)

$\mathrm{wt}(r) = 0.4 \cdot 0.2 \cdot 0.3 \cdot 0.1 \cdot 0.3 \cdot 0.1 \cdot 0.4 \cdot 0.2 \cdot 0.1 \cdot 0.3 \cdot 0.2 \cdot 0.1$

Semantics

Definition

The weight $A(t)$ of input tree t
= sum of weights of all runs ending in initial state

$$A(t) = \sum_{\substack{r \text{ run on } t \\ \text{root}(r) \in I}} \text{wt}(r)$$

Note

Weighted tree language regular if computable by WTA

Semantics

Definition

The weight $A(t)$ of input tree t
= sum of weights of all runs ending in initial state

$$A(t) = \sum_{\substack{r \text{ run on } t \\ \text{root}(r) \in I}} \text{wt}(r)$$

Note

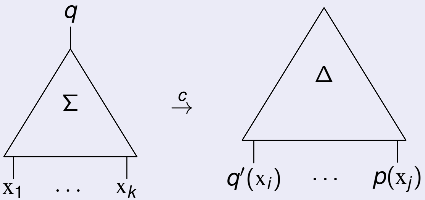
Weighted tree language regular if computable by WTA

Contents

Syntax

Definition (Arnold, Dauchet 1976, Graehl, Knight 2004)

Weighted extended top-down tree transducer (WXTT)
$M = (Q, \Sigma, \Delta, I, R)$ with finitely many rules

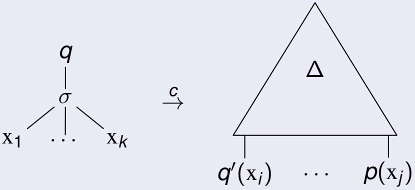


- $q, q', p \in Q$ are states
- $i, j \in \{1, \dots, k\}$

Syntax (cont'd)

Definition (Rounds 1970, Thatcher 1970)

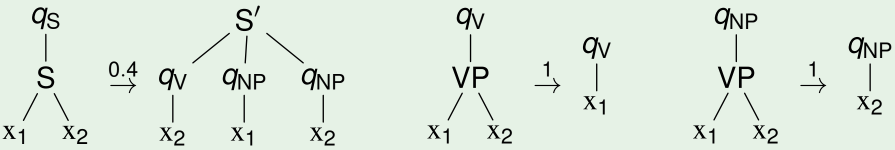
Weighted top-down tree transducer (WTT) if all rules



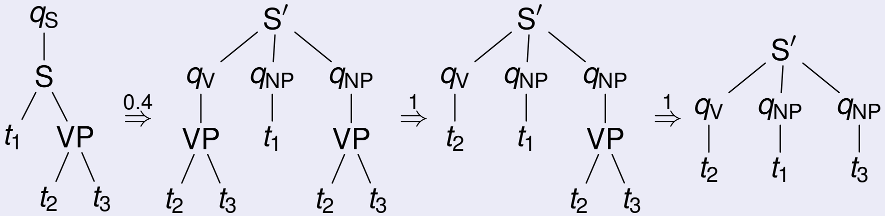
Semantics

Example

States $\{q_S, q_V, q_{NP}\}$ of which only q_S is initial



Derivation



Semantics (cont'd)

Definition

Computed transformation ($t \in T_\Sigma$ and $u \in T_\Delta$):

$$M(t, u) = \sum_{\substack{q \in I \\ q(t) \overset{c_1}{\Rightarrow} \cdots \overset{c_n}{\Rightarrow} u \\ \text{left-most derivation}}} c_1 \cdot \ldots \cdot c_n$$

Contents

Input product

Definition

Given WTA A and WTT M, their input product is WTT N with

$$N(t, u) = M(t, u) \cdot A(t)$$

Notes

Input product . . .

- is special composition
- is like regular look-ahead
- can be used for parsing
- can be used for preservation of regularity

Input product

Definition

Given WTA A and WTT M, their input product is WTT N with

$$N(t, u) = M(t, u) \cdot A(t)$$

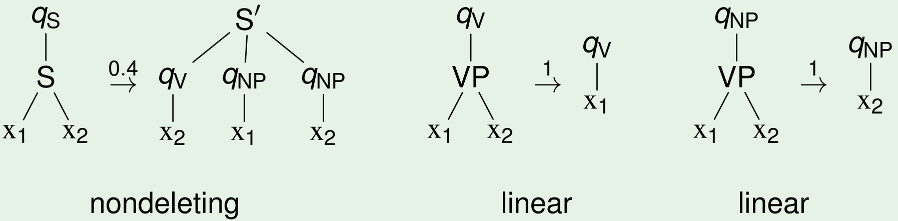
Notes

Input product ...

- is special composition
- is like regular look-ahead
- can be used for parsing
- can be used for preservation of regularity

Nondeletion

Example



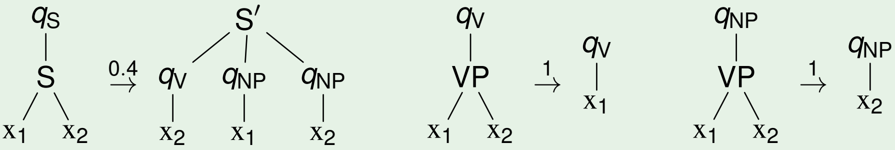
Definition

WTT M is

- nondeleting if $\text{var}(l) = \text{var}(r)$ for all rules $l \to r$
- linear if no variable appears twice in r for all rules $l \to r$

Nondeletion

Example



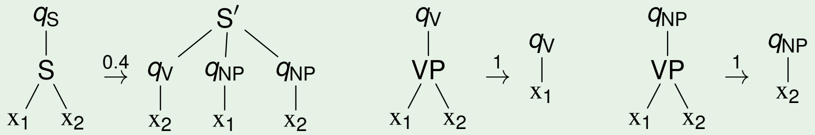
nondeleting | deletes x_2 | deletes x_1

Definition

- all-copies nondeleting = nondeleting
  = every copy of an input subtree is fully explored
- some-copy nondeleting
  = one copy of each input subtree is fully explored

Nondeletion (cont'd)

Example



is not some-copy nondeleting

Nondeletion (cont'd)

Example

$$q_{S}(S(x_1, x_2)) \xrightarrow{0.4} S'(q_{V}(x_2), q_{NP}(x_1), q_{NP}(x_2))$$

$$q_{V}(VP(x_1, x_2)) \xrightarrow{1} q_{V}(x_1)$$

$$q_{NP}(VP(x_1, x_2)) \xrightarrow{1} VP(q_{NP}(x_2), q_{V}(x_1))$$

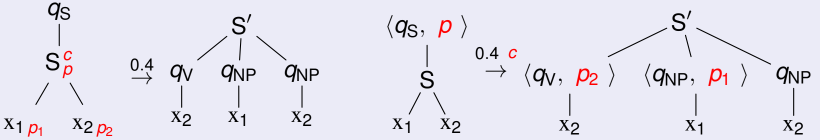
can be some-copy nondeleting

Scenario 1

Theorem (Engelfriet 1977)

For nondeleting WTT and WTA we can construct their input product.

Proof.



original rules — constructed rule

- for original nondeleting rules construct new rules
- mark one state for each variable; one possibility
- $x_2\,a \quad x_1\,b \quad x_2\,d \quad \rightarrow \quad \boxed{x_2}^{e}_{a} \quad \boxed{x_1}^{f}_{b} \quad x_2\,d$

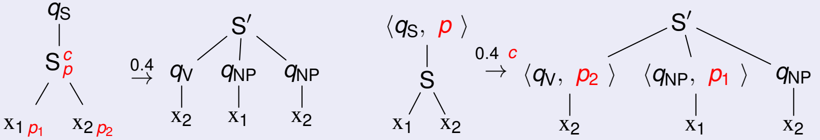
□

Scenario 1

Theorem (Engelfriet 1977)

For nondeleting WTT and WTA we can construct their input product.

Proof.



original rules   constructed rule

- for original nondeleting rules construct new rules
- mark one state for each variable; one possibility
- $x_2\,a \quad x_1\,b \quad x_2\,d \quad \rightarrow \quad \boxed{x_2}^{e}_{a} \quad \boxed{x_1}^{f}_{b} \quad x_2\,d$

Scenario 2

Theorem

For some-copy nondeleting WXTT and WTA over idempotent semiring we can construct their input product.

Proof.

- for original nondeleting rules construct new rules
- mark one state for each variable; all possibilities

$$x_{2}a \quad x_{1}b \quad x_{2}d \quad \to \quad \boxed{x_2}^{e}_{a} \quad \boxed{x_1}^{f}_{b} \quad x_{2}d \quad \mid \quad x_{2}a \quad \boxed{x_1}^{f}_{b} \quad \boxed{x_2}^{e}_{d}$$

- at least one exploration will succeed (somy-copy nondeletion)
- $aebfd + abfde = abdef$ if several succeed (idempotency)

□

Scenario 2

Theorem

For some-copy nondeleting WXTT and WTA over idempotent semiring we can construct their input product.

Proof.

- for original nondeleting rules construct new rules
- mark one state for each variable; all possibilities

$$x_{2\,a} \quad x_{1\,b} \quad x_{2\,d} \quad \to \quad \boxed{x_2}^{e}_{a} \quad \boxed{x_1}^{f}_{b} \quad x_{2\,d} \quad \Big| \quad x_{2\,a} \quad \boxed{x_1}^{f}_{b} \quad \boxed{x_2}^{e}_{d}$$

- at least one exploration will succeed (somy-copy nondeletion)
- $aebfd + abfde = abdef$ if several succeed (idempotency)

□

Scenario 2

Theorem

For some-copy nondeleting WXTT and WTA over idempotent semiring we can construct their input product.

Proof.

- for original nondeleting rules construct new rules
- mark one state for each variable; all possibilities

$$x_2 a \quad x_1 b \quad x_2 d \quad \to \quad \boxed{x_2}^{e}_{a} \quad \boxed{x_1}^{f}_{b} \quad x_2 d \quad \mid \quad x_2 a \quad \boxed{x_1}^{f}_{b} \quad \boxed{x_2}^{e}_{d}$$

- at least one exploration will succeed (somy-copy nondeletion)
- $aebfd + abfde = abdef$ if several succeed (idempotency)

□

Scenario 3

Theorem

For some-copy nondeleting WXTT and WTA over ring we can construct their input product.

Proof.

- for original nondeleting rules construct several new rules
- mark states according to elimination scheme
- $x_{2a} \quad x_{1b} \quad x_{2d} \quad \to$

$$\boxed{x_2}^{e}_{a} \; \boxed{x_1}^{f}_{b} \; x_{2d} \quad \Big| \quad x_{2a} \; \boxed{x_1}^{f}_{b} \; \boxed{x_2}^{e}_{d} \quad \Big| \quad \boxed{x_2}^{e}_{a} \; \boxed{x_1}^{f}_{b} \; \boxed{x_2}^{-1}_{d}$$

- at least one exploration will succeed

□

Scenario 3

Theorem

For some-copy nondeleting WXTT and WTA over ring we can construct their input product.

Proof.

- for original nondeleting rules construct several new rules
- mark states according to elimination scheme
- $x_{2\,a} \quad x_{1\,b} \quad x_{2\,d} \quad \to$

$$\boxed{x_2}^{e}_{a} \quad \boxed{x_1}^{f}_{b} \quad x_{2\,d} \quad \Big| \quad x_{2\,a} \quad \boxed{x_1}^{f}_{b} \quad \boxed{x_2}^{e}_{d} \quad \Big| \quad \boxed{x_2}^{e}_{a} \quad \boxed{x_1}^{f}_{b} \quad \boxed{x_2}^{-1}_{d}$$

- at least one exploration will succeed

□

Scenario 3

Theorem

For some-copy nondeleting WXTT and WTA over ring we can construct their input product.

Proof.

- if several succeed, then

$\boxed{x_2}_a^e \ \boxed{x_1}_b^f \ x_{2\,d}$	$x_{2\,a} \ \boxed{x_1}_b^f \ \boxed{x_2}_d^e$	$\boxed{x_2}_a^e \ \boxed{x_1}_b^f \ \boxed{x_2}_d^{-1}$
$aebfd$	0	0
0	$abfde$	0
$aebfd$	$abfde$	$-aebfd$

□

Elimination schemes

Question

Do elimination schemes exist?

Answer

	001	010	100	011	101	110	111	$\sum$
	$+$	$+$	$+$	$-$	$-$	$-$	$+$	
001	a	0	0	0	0	0	0	a
010	0	a	0	0	0	0	0	a
100	0	0	a	0	0	0	0	a
011	a	a	0	$-a$	0	0	0	a
101	a	0	a	0	$-a$	0	0	a
110	0	a	a	0	0	$-a$	0	a
111	a	a	a	$-a$	$-a$	$-a$	a	a

Elimination schemes

Question

Do elimination schemes exist?

Answer

	001 $+$	010 $+$	100 $+$	011 $-$	101 $-$	110 $-$	111 $+$	$\sum$
001	a	0	0	0	0	0	0	a
010	0	a	0	0	0	0	0	a
100	0	0	a	0	0	0	0	a
011	a	a	0	$-a$	0	0	0	a
101	a	0	a	0	$-a$	0	0	a
110	0	a	a	0	0	$-a$	0	a
111	a	a	a	$-a$	$-a$	$-a$	a	a

Elimination schemes

Question

Do elimination schemes exist?

Answer

	001	010	100	011	101	110	111	$\sum$
	$+$	$+$	$+$	$-$	$-$	$-$	$+$	
001	a	0	0	0	0	0	0	a
010	0	a	0	0	0	0	0	a
100	0	0	a	0	0	0	0	a
011	a	a	0	$-a$	0	0	0	a
101	a	0	a	0	$-a$	0	0	a
110	0	a	a	0	0	$-a$	0	a
111	a	a	a	$-a$	$-a$	$-a$	a	a

References

- ARNOLD, DAUCHET: *Bi-transductions de forêts.* ICALP 1976
- BERSTEL, REUTENAUER: *Recognizable formal power series on trees.* Theor. Comput. Sci. 18, 1982
- ENGELFRIET: *Top-down tree transducers with regular look-ahead.* Math. Systems Theory 10, 1977
- GRAEHL, KNIGHT: *Training tree transducers.* HLT-NAACL 2004
- MALETTI, SATTA: *Parsing and translation algorithms based on weighted extended tree transducers.* ATANLP 2010
- MAY, KNIGHT: TIBURON — *a weighted tree automata toolkit.* CIAA 2006
- ROUNDS: *Mappings and grammars on trees.* Math. Systems Theory 4, 1970
- THATCHER *Generalized*^{2} *sequential machine maps.* J. Comput. System Sci. 4, 1970

Thank you for your attention!