Extended Multi Bottom-up Tree Transducers

Joost Engelfriet[1], Eric Lilin[2], and Andreas Maletti[3]

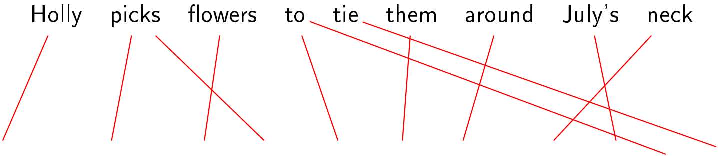
[1] LIACS, Leiden, The Netherlands
[2] Université de Lille, France
[3] ICSI, Berkeley, CA, USA

maletti@icsi.berkeley.edu

Kyoto — September 16, 2008

Classic Approach (simplified)

Source: Google Translate

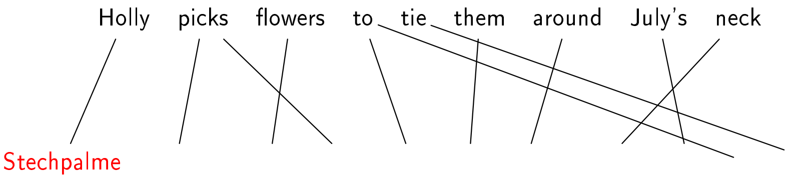


Steps

1. Select scheme
2. Translate words individually
3. Rescore using word sequences (bigrams, trigrams, etc.)

Classic Approach (simplified)

Source: Google Translate

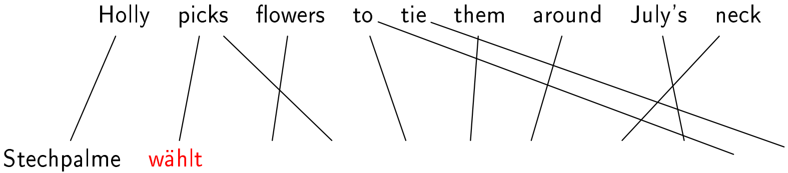


Steps

1. Select scheme
2. Translate words individually
3. Rescore using word sequences (bigrams, trigrams, etc.)

Classic Approach (simplified)

Source: Google Translate

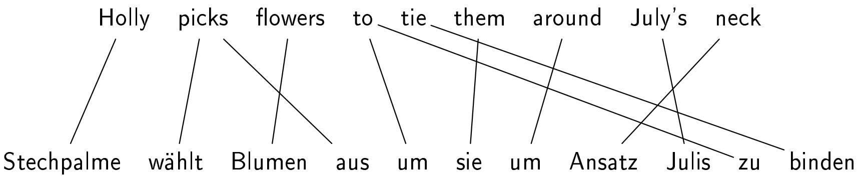


Steps

1. Select scheme
2. Translate words individually
3. Rescore using word sequences (bigrams, trigrams, etc.)

Classic Approach (simplified)

Source: Google Translate

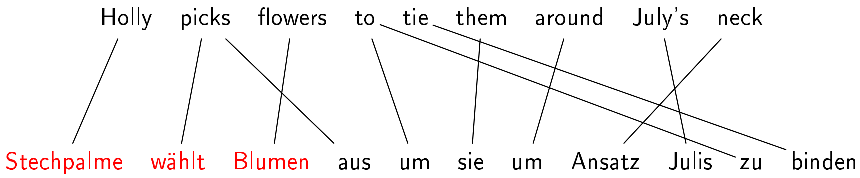


Steps

1. Select scheme
2. Translate words individually
3. Rescore using word sequences (bigrams, trigrams, etc.)

Classic Approach (simplified)

Source: Google Translate

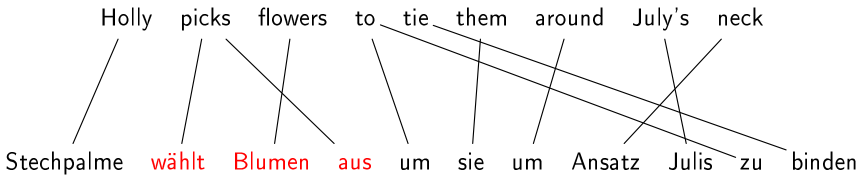


Steps

1. Select scheme
2. Translate words individually
3. Rescore using word sequences (bigrams, trigrams, etc.)

Classic Approach (simplified)

Source: Google Translate

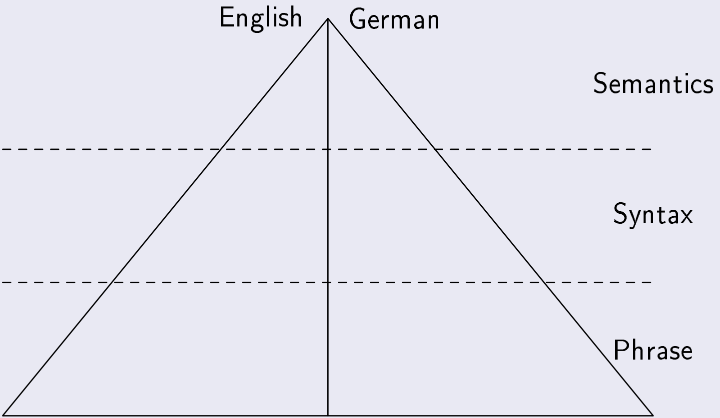


Steps

1. Select scheme
2. Translate words individually
3. Rescore using word sequences (bigrams, trigrams, etc.)

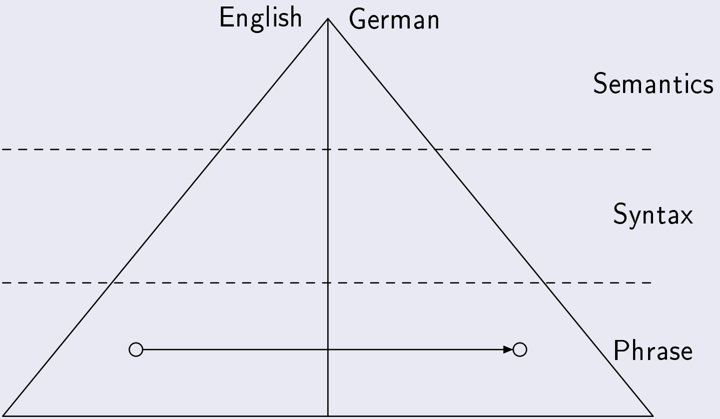
Syntax-based Approach

Overview



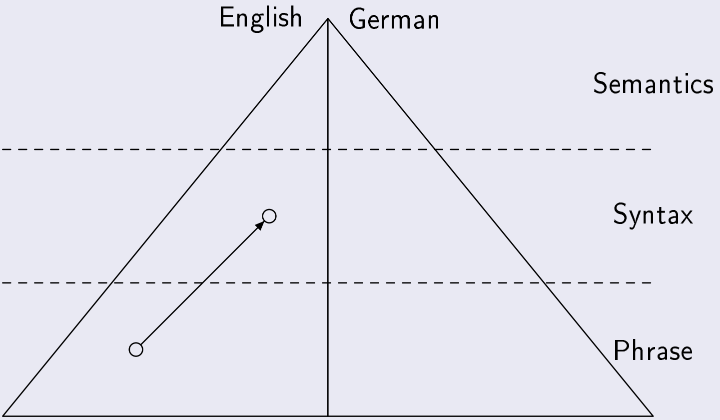
Syntax-based Approach

Overview



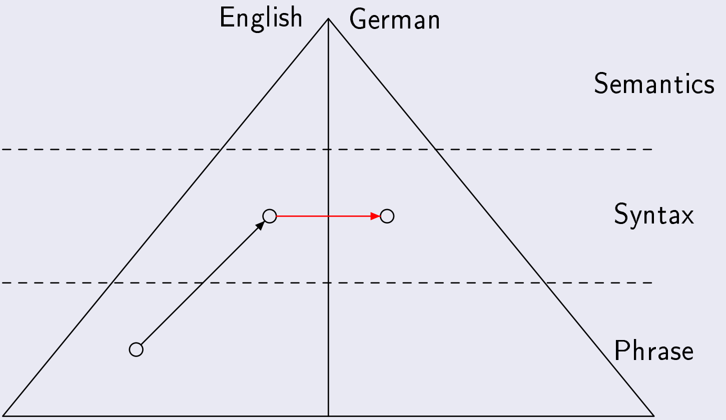
Syntax-based Approach

Overview



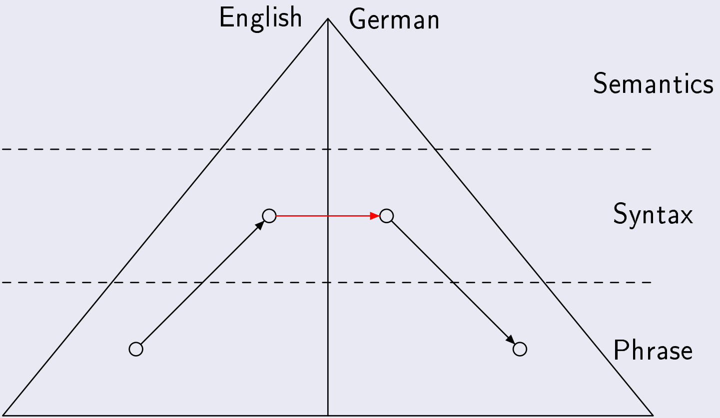
Syntax-based Approach

Overview



Syntax-based Approach

Overview



Syntax-based Approach (cont'd)

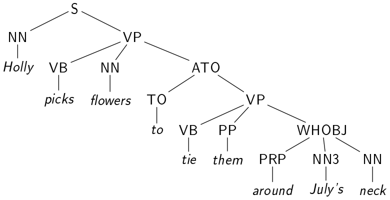
Holly picks flowers to tie them around July's neck

$\Downarrow$

Syntax-based Approach (cont'd)

Holly picks flowers to tie them around July's neck

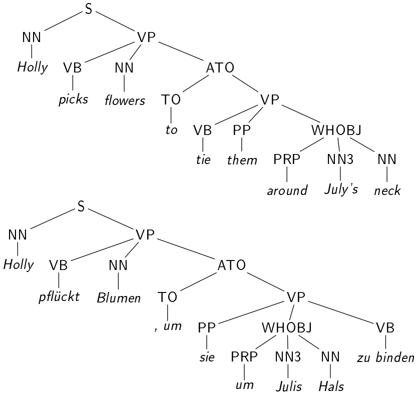
$\Downarrow$



Syntax-based Approach (cont'd)

Holly picks flowers to tie them around July's neck

$\Downarrow$

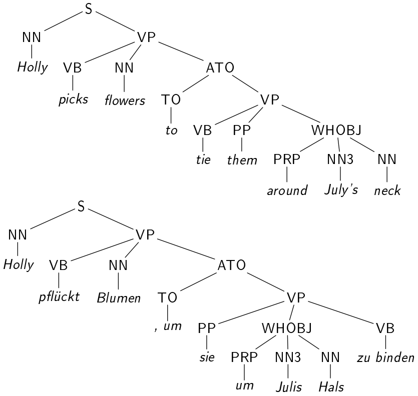


$\Downarrow$

Syntax-based Approach (cont'd)

Holly picks flowers to tie them around July's neck

$\Downarrow$



$\Downarrow$

Holly pflückt Blumen, um sie um Julis Hals zu binden

Reasonable formal models (ala Knight)

Required properties

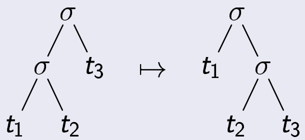
(a) generalize finite-state transducers (the string case)
(b) efficiently trainable
(c) able to handle rotations
(d) class of transformations closed under composition

Reasonable formal models (ala Knight)

Required properties

(a) generalize finite-state transducers (the string case)
(b) efficiently trainable
(c) able to handle rotations
(d) class of transformations closed under composition

Example rotation



Models of tree transformations

Overview

Model \ Criterion	(a)	(b)	(c)	(d)
Linear nondeleting top-down tt				
Quasi-alphabetic tree bimorphism		?		
Synchronous context-free grammar				
Synchronous tree substitution grammar				
Synchronous tree adjoining grammar				
Linear complete tree bimorphism				
Linear extended top-down tt				
Linear multi bottom-up tt		?		

Models of tree transformations

Overview

Model \ Criterion	(a)	(b)	(c)	(d)
Linear nondeleting top-down tt				
Quasi-alphabetic tree bimorphism		?		
Synchronous context-free grammar				
Synchronous tree substitution grammar				
Synchronous tree adjoining grammar				
Linear complete tree bimorphism				
Linear extended top-down tt				
Linear multi bottom-up tt		?		
Linear extended multi bottom-up tt		?		

Table of Contents

Syntax

Definition

eXtended Multi Bottom-Up Tree Transducer $(Q, \Sigma, \Delta, F, R)$

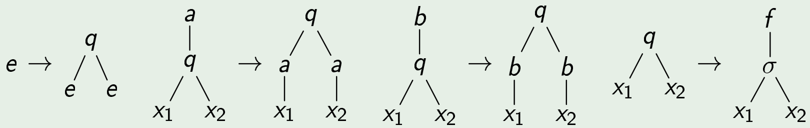
- Q ranked alphabet of *states*
- Σ and Δ ranked alphabets of input and output symbols
- $F \subseteq Q$ final states (non-zero ranked)
- R finite set of (rewrite) rules $l \to r$ with $l \in T_\Sigma(Q(X))$ and $r \in Q(T_\Delta(X))$

A full example

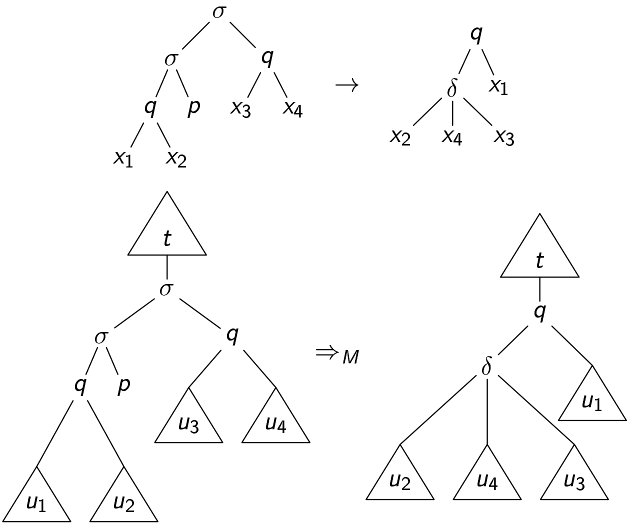
Example

xmbutt $(Q, \Sigma, \Delta, F, R)$ with

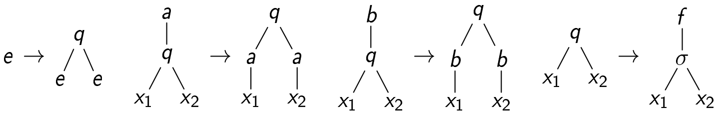
- $Q = \{f, q\}$
- $\Sigma = \{a^{(1)}, b^{(1)}, e^{(0)}\}$ and $\Delta = \Sigma \cup \{\sigma^{(2)}\}$
- $F = \{f\}$
- the following rules in R



Semantics

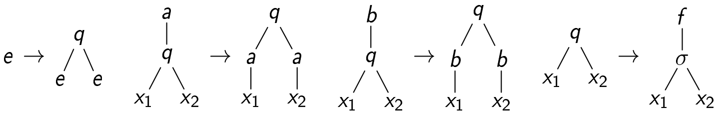


A full example (cont'd)

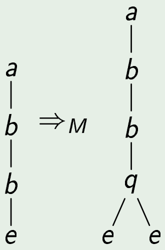


Example

A full example (cont'd)



Example



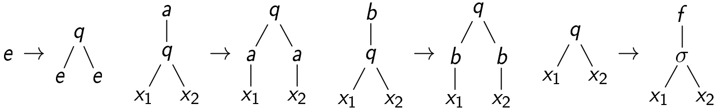
A full example (cont'd)

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

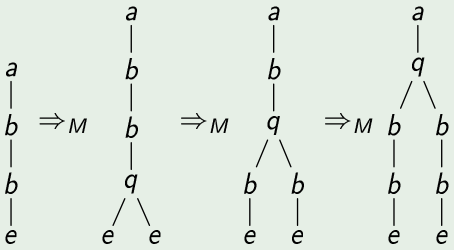
Example

$$a(b(b(e))) \Rightarrow_M a(b(b(q(e, e)))) \Rightarrow_M a(b(q(b(e), b(e))))$$

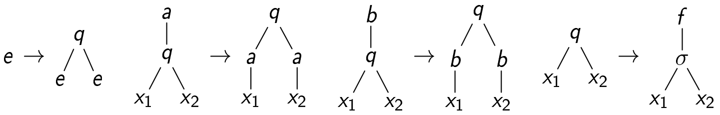
A full example (cont'd)



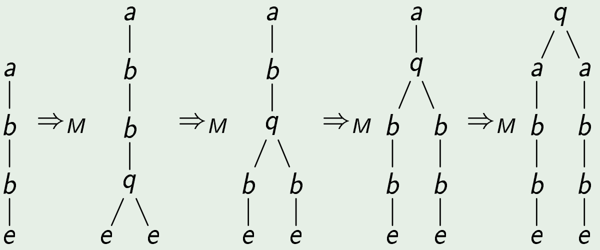
Example



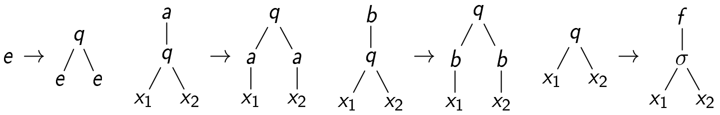
A full example (cont'd)



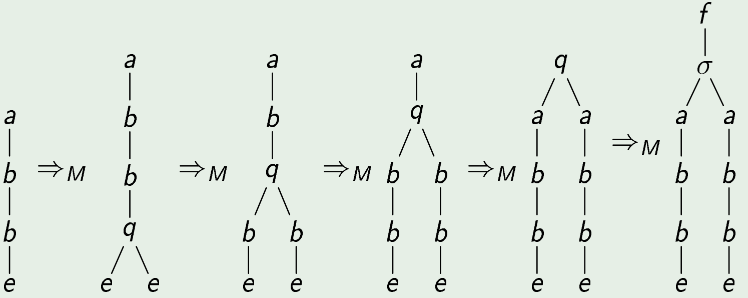
Example



A full example (cont'd)



Example



Semantics (cont'd)

Definition

$M = (Q, \Sigma, \Delta, F, R)$ xmbutt

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

Example

Our example xmbutt M computes:

$$\tau_M = \{(t, \sigma(t, t)) \mid t \in T_\Sigma\}$$

Note that the image is not recognizable.

Semantics (cont'd)

Definition

$M = (Q, \Sigma, \Delta, F, R)$ xmbutt

$$\tau_M = \{(t, u) \mid \exists q \in F \colon t \Rightarrow_M^* q(u, \dots)\}$$

Example

Our example xmbutt M computes:

$$\tau_M = \{(t, \sigma(t, t)) \mid t \in T_\Sigma\}$$

Note that the image is not recognizable.

Syntactical restrictions

Definition

xmbutt $(Q, \Sigma, \Delta, F, R)$ is

- linear if every right-hand side (of a rule) does not contain duplicate variables
- nondeleting, if every right-hand side contains all variables of its left-hand side

Example

Our example xmbutt is linear and nondeleting.

Syntactical restrictions

Definition

xmbutt $(Q, \Sigma, \Delta, F, R)$ is

- linear if every right-hand side (of a rule) does not contain duplicate variables
- nondeleting, if every right-hand side contains all variables of its left-hand side

Example

Our example xmbutt is linear and nondeleting.

Table of Contents

Generalization of the string case

Theorem

Every fst (including epsilon-rules) can be simulated by an xmbutt.

Property summary

generalize finite-state transducers (the string case)

(b) efficiently trainable

(c) able to handle rotations

(d) class of transformations closed under composition

Generalization of the string case

Theorem

Every fst (including epsilon-rules) can be simulated by an xmbutt.

Property summary

generalize finite-state transducers (the string case)

(b) efficiently trainable

(c) able to handle rotations

(d) class of transformations closed under composition

Nondeletion

Theorem

For every (linear) xmbutt there exists an equivalent (linear) nondeleting xmbutt.

Proof.

Nondeterministically guess which subtrees will be deleted and process those in nullary states. □

Note

Nullary states act like look-ahead.

Nondeletion

Theorem

For every (linear) xmbutt there exists an equivalent (linear) nondeleting xmbutt.

Proof.

Nondeterministically guess which subtrees will be deleted and process those in nullary states. □

Note

Nullary states act like look-ahead.

Nondeletion

Theorem

For every (linear) xmbutt there exists an equivalent (linear) nondeleting xmbutt.

Proof.

Nondeterministically guess which subtrees will be deleted and process those in nullary states. □

Note

Nullary states act like look-ahead.

Training

Theorem (Decomposition)

Every (linear) xmbutt can be simulated by a composition of a linear and nondeleting extended top-down tree transducer and a deterministic (single-use) top-down tree transducer.

Property summary

- generalize finite-state transducers (the string case)
- ? efficiently trainable
- (c) able to handle rotations
- (d) class of transformations closed under composition

Training

Theorem (Decomposition)

Every (linear) xmbutt can be simulated by a composition of a linear and nondeleting extended top-down tree transducer and a deterministic (single-use) top-down tree transducer.

Property summary

- generalize finite-state transducers (the string case)
- ? efficiently trainable
- (c) able to handle rotations
- (d) class of transformations closed under composition

Rotations

Theorem

Every linear extended top-down tree transducer can be simulated by a linear xmbutt.

Proof.

Similar as the proof for the non-extended case. □

Corollary

Xmbutt can handle rotations.

Rotations

Theorem

Every linear extended top-down tree transducer can be simulated by a linear xmbutt.

Proof.

Similar as the proof for the non-extended case. □

Corollary

Xmbutt can handle rotations.

Rotations

Theorem

Every linear extended top-down tree transducer can be simulated by a linear xmbutt.

Proof.

Similar as the proof for the non-extended case. □

Corollary

Xmbutt can handle rotations.

Rotations (cont'd)

Property summary

- generalize finite-state transducers (the string case)
- ? efficiently trainable
- able to handle rotations
- (d) class of transformations closed under composition

One-symbol normal form

Definition

xmbutt $(Q, \Sigma, \Delta, F, R)$ in one-symbol normal form if exactly one input or output symbol occurs in each rule.

Theorem

For every (linear, nondeleting) xmbutt there exists an equivalent (linear, nondeleting) xmbutt in one-symbol normal form.

One-symbol normal form

Definition

xmbutt $(Q, \Sigma, \Delta, F, R)$ in one-symbol normal form if exactly one input or output symbol occurs in each rule.

Theorem

For every (linear, nondeleting) xmbutt there exists an equivalent (linear, nondeleting) xmbutt in one-symbol normal form.

One-symbol normal form (cont'd)

Example

Rule not in one-symbol normal form:

$$b(q(x_1, x_2)) \to q(b(x_1), b(x_2))$$

Replacement rules for this rule:

$$b(q(x_1, x_2)) \to q_1(x_1, x_2) \qquad q_1(x_1, x_2) \to q_2(b(x_1), x_2) \qquad q_2(x_1, x_2) \to q(x_1, b(x_2))$$

Composition construction

Definition

from xmbutt $M = (Q, \Sigma, \Gamma, F, R)$ and $N = (Q', \Gamma, \Delta, G, P)$ construct

$$M \,;\, N = (Q\langle Q'\rangle, \Sigma, \Delta, F\langle G\rangle, R')$$

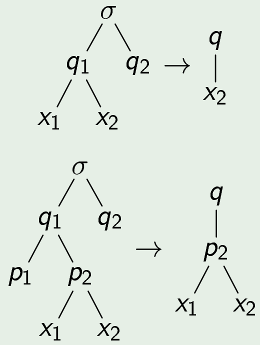
with three types of rules:

1. input-consuming rules constructed from input-consuming rules of R
2. epsilon rules constructed from epsilon-rules of P
3. epsilon rules constructed from an epsilon rule of R followed by an input consuming rule of P

Composition construction (cont'd)

Example

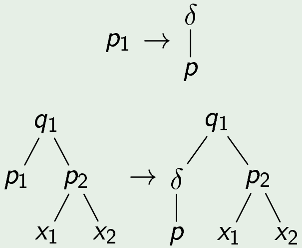
Input consuming rule of R and resulting rule:



Composition construction (cont'd)

Example

Epsilon rule of P and resulting rule:



Composition construction (cont'd)

Example

Epsilon rule of R and input consuming of P and resulting rule:

$$q_1(x_1, x_2) \to q(\gamma(x_2)) \qquad \gamma(p_2(x_1, x_2)) \to p(x_2)$$

$$q_1(p_1, p_2(x_1, x_2)) \to q(\gamma(p_2(x_1, x_2))) \to q(p(x_2))$$

Closure under composition

Theorem

For every linear xmbutt M and xmbutt N, there exists an xmbutt that computes the composition of the transformations computed by M and N.

Corollary

The class of transformations computed by linear xmbutt is closed under composition.

Closure under composition

Theorem

For every linear xmbutt M and xmbutt N, there exists an xmbutt that computes the composition of the transformations computed by M and N.

Corollary

The class of transformations computed by linear xmbutt is closed under composition.

Closure under composition (cont'd)

Property summary

- generalize finite-state transducers (the string case)
- ? efficiently trainable
- able to handle rotations
- class of transformations closed under composition

References

- Lilin: Une généralisation des transducteurs d'états finis d'arbres: les S-transducteurs. Université de Lille. *Thèse 3ème cycle*, 1978
- Lilin: Propriétés de clôture d'une extension de transducteurs d'arbres déterministes. In *CAAP*, LNCS 112, p. 280–289, 1981
- Engelfriet: Bottom-up and top-down tree transformations — a comparison. *Math. Systems Theory* 9(3), 1975
- Gécseg, Steinby: *Tree Automata*. Akadémiai Kiadó, Budapest 1984

Thank you for your attention!