Applications of Tree Automata Theory
Lecture III: Parsing — Advanced Topics

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 24, 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!

Overview

Topics

- Foundations of Tree Automata
- Applications of TA in NLP
  (yielding TAs and algorithms for further study)
- Application of TA theory in NLP
  (solving NLP problems)

Overview

Lexicalized Grammars

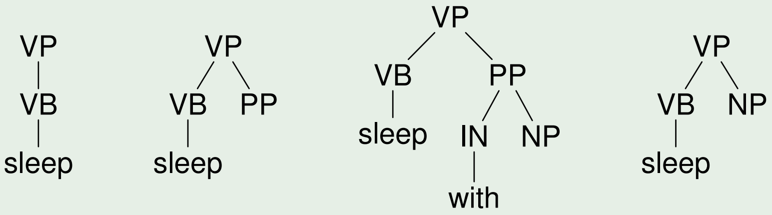
Lexicon

MERRIAM-WEBSTER entry for *sleep*

- intransitive verb
  1. to rest in a state of sleep
  2. to be in a state (as of quiescence or death) resembling sleep
  3. to have sexual relations — usually used with *with*
- transitive verb
  1. to be slumbering in — slept the sleep of the dead
  2. to get rid of or spend in or by sleep — sleep away the hours, sleep off a headache
  3. to provide sleeping accommodations for — the boat sleeps six

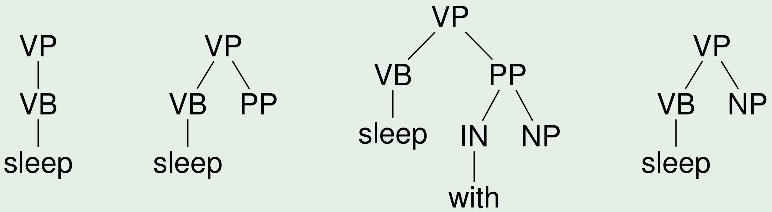
Linguistic Constructions

Fragments



Linguistic Constructions

Fragments



Constructions

- major research area for traditional linguistics

Lexicalized Grammars

Definition (Meta)

A grammar with productions *P* is lexicalized
if each production $\rho \in P$ contains at least one lexical item

Lexicalized Grammars

Definition (Meta)

A grammar with productions P is lexicalized
if each production $\rho \in P$ contains at least one lexical item

Example

- NFAs are lexicalized
- CFGs in GREIBACH normal form are lexicalized

Lexicalized Grammars

Benefits

- ideal for parsing
  - bound on number of applied productions
  - grammar pruning based on occurring lexical items
- linguistic constructions evident
- theoretical advantages

Lexicalized Grammars

Benefits

- ideal for parsing
  - bound on number of applied productions
  - grammar pruning based on occurring lexical items
- linguistic constructions evident
- theoretical advantages

Disadvantages

- have only finite ambiguity theoretical disadvantage
  (for every sentence there are only finitely many parses)

Overview

Lexicalization

Weak Lexicalization

classes $\mathcal{G}, \mathcal{G}'$ of string grammars

Definition

$\mathcal{G}'$ (weakly) lexicalizes $\mathcal{G}$ if for every $G \in \mathcal{G}$
there exists an equivalent lexicalized $G' \in \mathcal{G}'$

Example

- CFGs weakly lexicalize themselves

(via the GREIBACH normal form)

Weak Lexicalization

Problem

- weak lexicalization only covers generated string language
- parses can look totally different

  (see GREIBACH normalization)
- weak lexicalization destroys linguistic knowledge present in treebanks

Tree Yield

Definition (Yield)

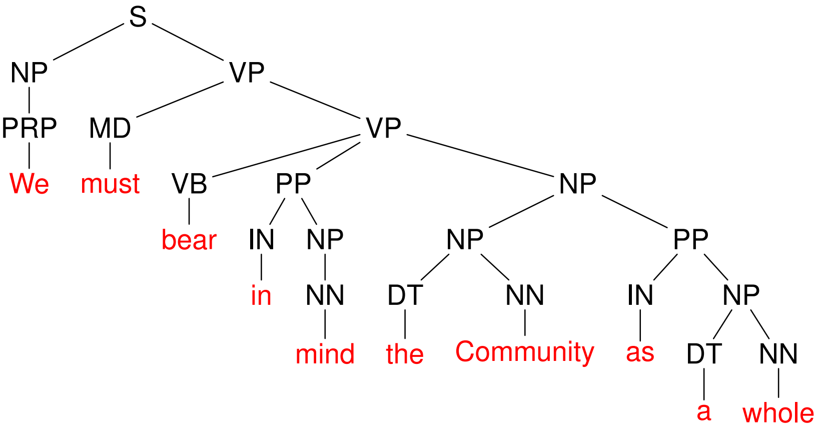
mapping $\mathrm{yd}\colon T_\Sigma(Q) \to Q^*$

$$\mathrm{yd}(q) = q$$
$$\mathrm{yd}(\sigma(t_1, \ldots, t_k)) = \mathrm{yd}(t_1) \cdots \mathrm{yd}(t_k)$$

for all $q \in Q$, $\sigma \in \Sigma$, and $t_1, \ldots, t_k \in T_\Sigma(Q)$

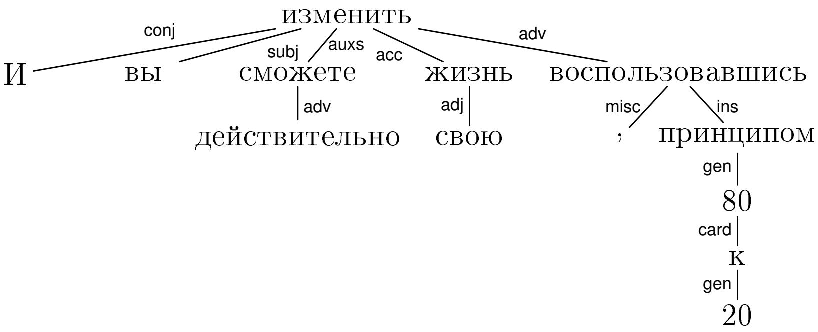
Tree Yield

Yield: We must bear in mind the Community as a whole



Tree Yield

Yield: И вы действительно свою, 20



Finite Ambiguity

Definition

A tree grammar G is finitely ambiguous if for every $w \in Q^*$

$$\text{Parses}(w) = \{t \in L(G) \mid \text{yd}(t) = w\}$$

is finite

Strong Lexicalization

classes $\mathcal{G}, \mathcal{G}'$ of tree grammars

Definition

$\mathcal{G}'$ (strongly) lexicalizes $\mathcal{G}$ if for every finitely ambiguous $G \in \mathcal{G}$ there exists an equivalent lexicalized $G' \in \mathcal{G}'$

Strong Lexicalization

classes $\mathcal{G}, \mathcal{G}'$ of tree grammars

Definition

$\mathcal{G}'$ (strongly) lexicalizes $\mathcal{G}$ if for every finitely ambiguous $G \in \mathcal{G}$ there exists an equivalent lexicalized $G' \in \mathcal{G}'$

Example

- LTGs strongly lexicalize themselves

[SCHABES, 1990]

Strong Lexicalization

classes $\mathcal{G}, \mathcal{G}'$ of tree grammars

Definition

$\mathcal{G}'$ (strongly) lexicalizes $\mathcal{G}$ if for every finitely ambiguous $G \in \mathcal{G}$ there exists an equivalent lexicalized $G' \in \mathcal{G}'$

Example

- LTGs cannot strongly lexicalize themselves

[Schabes, 1990]

Strong Lexicalization

classes $\mathcal{G}, \mathcal{G}'$ of tree grammars

Definition

$\mathcal{G}'$ (strongly) lexicalizes $\mathcal{G}$ if for every finitely ambiguous $G \in \mathcal{G}$ there exists an equivalent lexicalized $G' \in \mathcal{G}'$

Example

- LTGs cannot strongly lexicalize themselves [Schabes, 1990]
- TSGs strongly lexicalize LTGs [Schabes, 1990]

Strong Lexicalization

classes $\mathcal{G}, \mathcal{G}'$ of tree grammars

Definition

$\mathcal{G}'$ (strongly) lexicalizes $\mathcal{G}$ if for every finitely ambiguous $G \in \mathcal{G}$ there exists an equivalent lexicalized $G' \in \mathcal{G}'$

Example

- LTGs cannot strongly lexicalize themselves
  [SCHABES, 1990]
- TSGs cannot strongly lexicalize LTGs
  [SCHABES, 1990]

Strong Lexicalization

classes $\mathcal{G}, \mathcal{G}'$ of tree grammars

Definition

$\mathcal{G}'$ (strongly) lexicalizes $\mathcal{G}$ if for every finitely ambiguous $G \in \mathcal{G}$ there exists an equivalent lexicalized $G' \in \mathcal{G}'$

Example

- LTGs cannot strongly lexicalize themselves [Schabes, 1990]
- TSGs cannot strongly lexicalize LTGs [Schabes, 1990]
- RTGs strongly lexicalize LTGs

Strong Lexicalization

classes $\mathcal{G}, \mathcal{G}'$ of tree grammars

Definition

$\mathcal{G}'$ (strongly) lexicalizes $\mathcal{G}$ if for every finitely ambiguous $G \in \mathcal{G}$ there exists an equivalent lexicalized $G' \in \mathcal{G}'$

Example

- LTGs cannot strongly lexicalize themselves [SCHABES, 1990]
- TSGs cannot strongly lexicalize LTGs [SCHABES, 1990]
- RTGs cannot strongly lexicalize LTGs

Overview

Tree Adjoining Grammars

Tree-Adjoining Grammars

Motivation

- mildly context-sensitive formalism
- productions express local dependencies

Tree-Adjoining Grammars

Motivation

- mildly context-sensitive formalism
- productions express local dependencies
- but can realize global dependencies

Tree-Adjoining Grammars

Motivation

- mildly context-sensitive formalism
- productions express local dependencies
- but can realize global dependencies

Applications

- TAG for English [XTAG Research Group, 2001]
- lexicalized TAG for German [Kallmeyer et al., 2010]

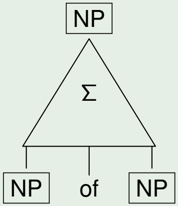
Tree-Adjoining Grammars

Definition (JOSHI et al., 1969)

$G = (N, \Sigma, S, R)$ tree-adjoining grammar (TAG) with finite set R

- substitution productions

Substitution production



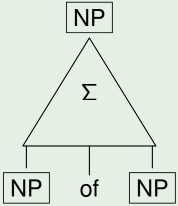
Tree-Adjoining Grammars

Definition (JOSHI et al., 1969)

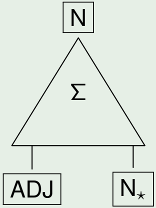
$G = (N, \Sigma, S, R)$ tree-adjoining grammar (TAG) with finite set R

- substitution productions
- adjunction productions

Substitution production



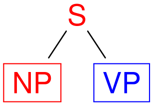
Adjunction production



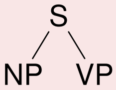
Tree-Adjoining Grammars

$\boxed{\text{S}}$

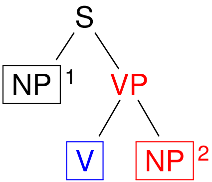
Tree-Adjoining Grammars



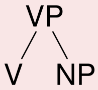
Used substitution production



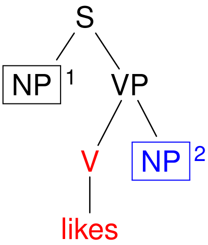
Tree-Adjoining Grammars



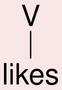
Used substitution production

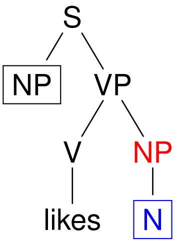


Tree-Adjoining Grammars



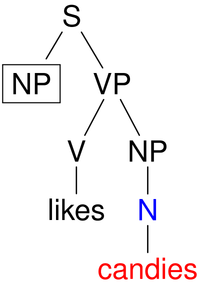
Used substitution production



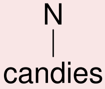


Used substitution production

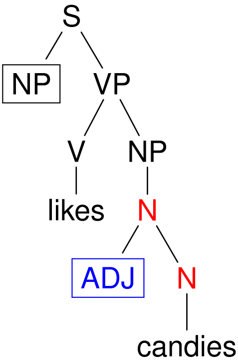
Tree-Adjoining Grammars



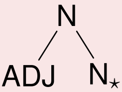
Used substitution production



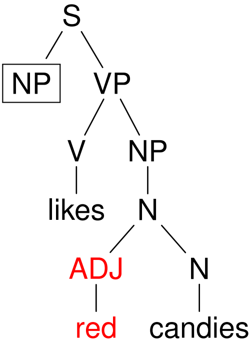
Tree-Adjoining Grammars



Used adjunction production



Tree-Adjoining Grammars



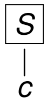
Used substitution production

Tree-Adjoining Grammars

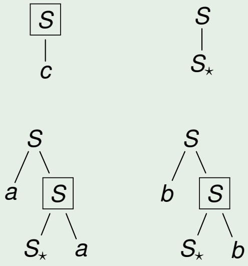
Definition (generated tree language)

$$L(G) = \bigcup_{A \in S} \{t \in T_\Sigma \mid A \Rightarrow_G^* t\}$$

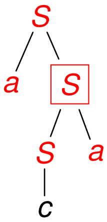
Tree-Adjoining Grammars



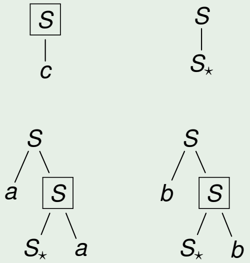
Productions



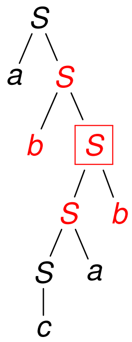
Tree-Adjoining Grammars



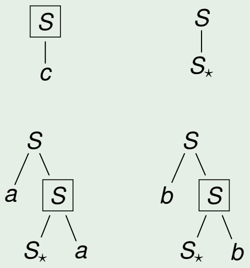
Productions



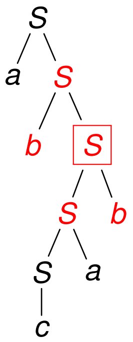
Tree-Adjoining Grammars



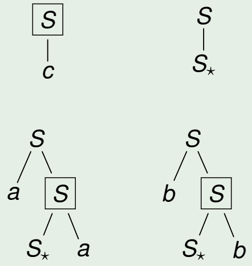
Productions



Tree-Adjoining Grammars



Productions



String language

$$\text{yd}(L(G)) = \{wcw \mid w \in \{a, b\}^*\}$$

Tree-Adjoining Grammars

Theorem (SCHABES, 1990)

TAGs can strongly lexicalize LTGs and themselves

Tree-Adjoining Grammars

Theorem (SCHABES, 1990 and KUHLMANN, SATTA, 2012)

TAGs can strongly lexicalize LTGs ~~and themselves~~
but not themselves

Tree-Adjoining Grammars

Theorem (SCHABES, 1990 and KUHLMANN, SATTA, 2012)

TAGs can strongly lexicalize LTGs ~~and themselves~~
but not themselves

Widespread myth

- KALLMEYER: *Parsing beyond context-free grammars*
  Springer, 2010
- JOSHI, SCHABES
  Tree-adjoining grammars
  In Handbook of Formal Languages, vol. 3, Springer, 1997
- JOSHI, SCHABES
  Tree-adjoining grammars and lexicalized grammars
  In Tree Automata and Languages, North-Holland, 1992

Overview

Context-free Tree Grammars

Context-free Tree Grammar

Definition (Rounds, 1969)

(N, Σ, S, P) context-free tree grammar (CFTG)

- alphabet N — nonterminals
- alphabet Σ — terminals
- $S \subseteq N$ — start nonterminals
- P is a finite set of $A(x_1, \ldots, x_k) \to r$ — productions
  - $A \in N$
  - $r \in C_{N \cup \Sigma}(\{x_1, \ldots, x_k\})$

Context-free Tree Grammar

Example

CFTG $(N, \Sigma, \{S\}, P)$
with
- $N = \{S, A\}$
- $\Sigma = \{\alpha, \beta, \sigma\}$

Productions

$$S \to A(\alpha, \alpha) \mid A(\beta, \beta) \mid \sigma(\alpha, \beta)$$
$$A(x_1, x_2) \to A(\sigma(x_1, S), \sigma(x_2, S))$$
$$A(x_1, x_2) \to \sigma(x_1, x_2)$$

Context-free Tree Grammar

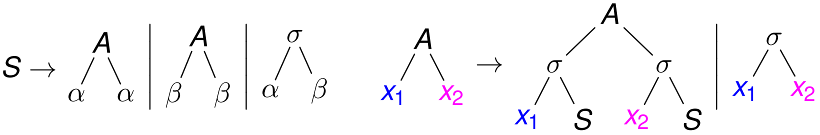
Example

CFTG $(N, \Sigma, \{S\}, P)$
with

- $N = \{S, A\}$
- $\Sigma = \{\alpha, \beta, \sigma\}$

Productions

$$
\begin{aligned}
S &\to A(\alpha, \alpha) \mid A(\beta, \beta) \mid \sigma(\alpha, \beta) \\
A(x_1, x_2) &\to A(\sigma(x_1, S), \sigma(x_2, S)) \\
A(x_1, x_2) &\to \sigma(x_1, x_2)
\end{aligned}
$$



Context-free Tree Grammar

Example

CFTG $(N, \Sigma, \{S\}, P)$
with

- $N = \{S, A\}$
- $\Sigma = \{\alpha, \beta, \sigma\}$

$$S \to A(\alpha, \alpha) \;\Big|\; A(\beta, \beta) \;\Big|\; \sigma(\alpha, \beta) \qquad A(x_1, x_2) \to A(\sigma(x_1, S), \sigma(x_2, S)) \;\Big|\; \sigma(x_1, x_2)$$

Context-free Tree Grammar

Productions

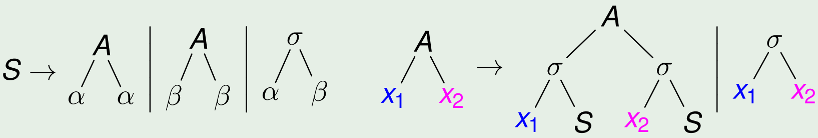
$$S \to A(\alpha, \alpha) \;\Big|\; A(\beta, \beta) \;\Big|\; \sigma(\alpha, \beta) \qquad A(\textcolor{blue}{x_1}, \textcolor{magenta}{x_2}) \to A\big(\sigma(\textcolor{blue}{x_1}, S), \sigma(\textcolor{magenta}{x_2}, S)\big) \;\Big|\; \sigma(\textcolor{blue}{x_1}, \textcolor{magenta}{x_2})$$

Derivation:

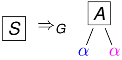
$\boxed{S}$

Context-free Tree Grammar

Productions

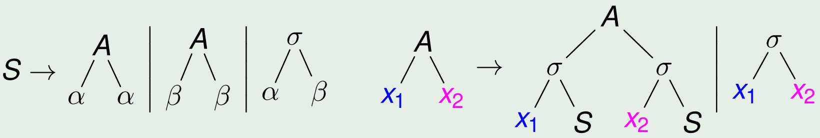


Derivation:

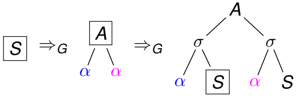


Context-free Tree Grammar

Productions

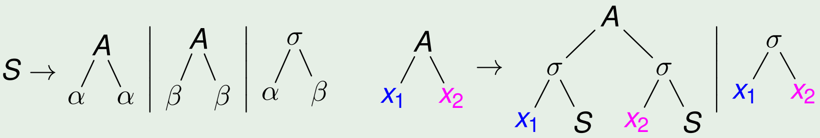


Derivation:

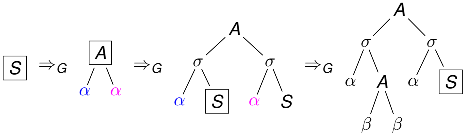


Context-free Tree Grammar

Productions

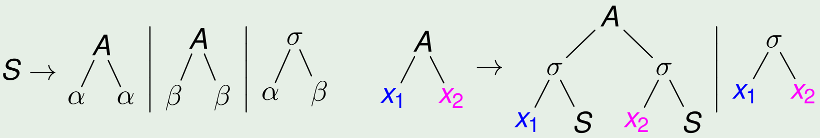


Derivation:

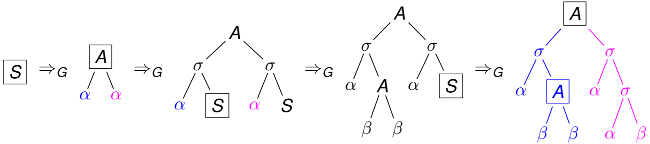


Context-free Tree Grammar

Productions

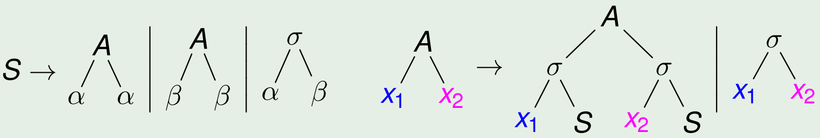


Derivation:

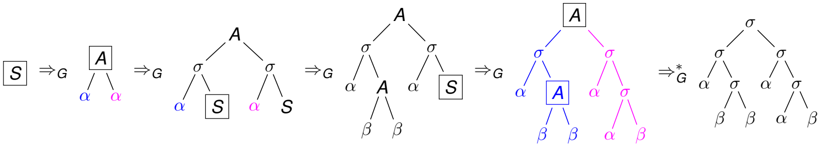


Context-free Tree Grammar

Productions



Derivation:



Context-free Tree Grammar

Definition (generated language)

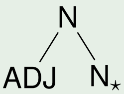
$$L(G) = \bigcup_{A \in S} \{t \in T_\Sigma \mid A \Rightarrow_G^* t\}$$

Context-free Tree Grammar

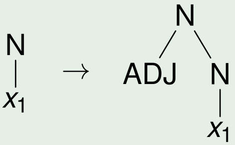
Theorem (JOSHI et al., 1975 and MÖNNICH, 1997)

For every (non-strict) TAG there is an equivalent CFTG

Adjunction production



Corresponding CFTG production



Overview

Lexicalization of TAG

Context-free Tree Grammar

Definition

CFTG (N, Σ, S, P) start-separated if

- it has a single start nonterminal $|S| = 1$
- the start nonterminal $A \in S$ does *not* occur in the right-hand sides of P

Context-free Tree Grammar

Definition

CFTG (N, Σ, S, P) start-separated if

- it has a single start nonterminal $|S| = 1$
- the start nonterminal $A \in S$ does *not* occur in the right-hand sides of P

Theorem

For every CFTG there is an equivalent start-separated CFTG

Context-free Tree Grammar

Definition

CFTG (N, Σ, S, P) start-separated if

- it has a single start nonterminal $|S| = 1$
- the start nonterminal $A \in S$ does *not* occur in the right-hand sides of P

Theorem

For every CFTG there is an equivalent start-separated CFTG

Proof.

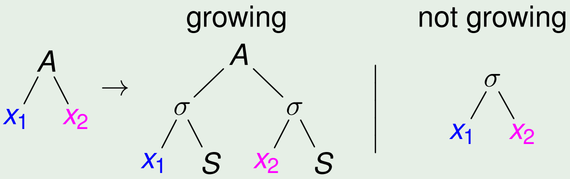
- add new start nonterminal A' $S' = \{A'\}$
- add new production $A' \to A$ for every $A \in S$ $\square$

Context-free Tree Grammar

Definition

CFTG growing if non-initial productions contain ≥ 3 non-variables

Example

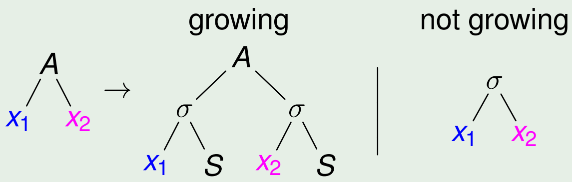


Context-free Tree Grammar

Definition

CFTG growing if non-initial productions contain ≥ 3 non-variables

Example



Theorem (Stamer, Otto, 2007)

For every CFTG there is an equivalent growing CFTG

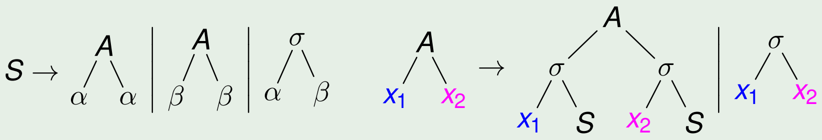
Context-free Tree Grammar

growing CFTG

CFTG

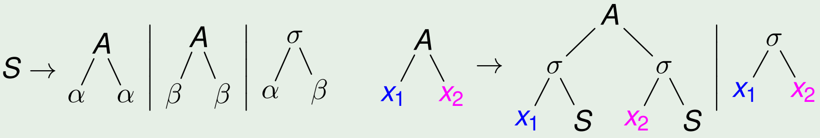
Context-free Tree Grammar

Example

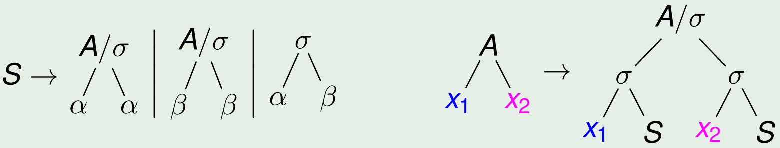


Context-free Tree Grammar

Example



Eliminate last production:



Context-free Tree Grammar

CFTG (N, Σ, S, P)

Definition

Production $\ell \to r \in P$

- monadic if r contains ≤ 1 nonterminals
- terminal if r contains 0 nonterminals

Context-free Tree Grammar

CFTG (N, Σ, S, P)

Definition

Production $\ell \to r \in P$

- monadic if r contains ≤ 1 nonterminals
- terminal if r contains 0 nonterminals
- lexicalized if r contains ≥ 1 lexical items
- doubly lexicalized if r contains ≥ 2 lexical items

Context-free Tree Grammar

Theorem (Engelfriet, $\sim$, 2012)

For every CFTG with finite ambiguity
there is an equivalent CFTG such that

- all (non-initial) monadic productions are lexicalized
- all (non-initial) terminal productions are doubly lexicalized

Context-free Tree Grammar

Theorem (ENGELFRIET, ∼, 2012)

For every CFTG with finite ambiguity
there is an equivalent CFTG such that

- all (non-initial) monadic productions are lexicalized
- all (non-initial) terminal productions are doubly lexicalized

Proof.

- similar to removal of ε-productions [HOPCROFT et al., 2001]
- compute closure under non-lexicalized productions □

Context-free Tree Grammar

$$\begin{tikzcd}\text{fully normalized CFTG} \\ \text{growing CFTG} \arrow[u] \\ \text{CFTG} \arrow[u]\end{tikzcd}$$

Lexicalization

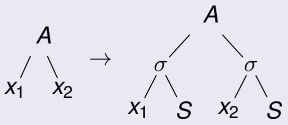
Theorem (ENGELFRIET, $\sim$, 2012)

Every CFTG with finite ambiguity can be strongly lexicalized

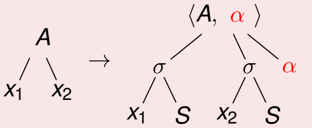
Lexicalization

1. guess lexical item in non-lexicalized production

Input



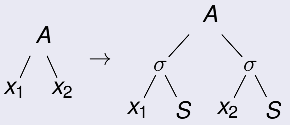
Output



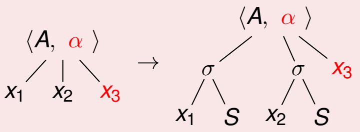
Lexicalization

1. guess lexical item in non-lexicalized production
2. transport guessed lexical item

Input



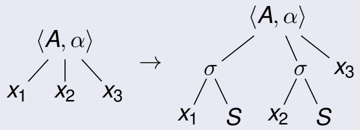
Output



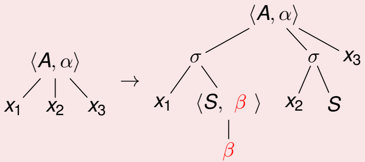
Lexicalization

1. guess lexical item in non-lexicalized production
2. transport guessed lexical item
3. potentially guess again

Input



Output



Lexicalization

1. guess lexical item in non-lexicalized production
2. transport guessed lexical item
3. potentially guess again
4. cancel in terminal production

Input

$$S \to \sigma(\alpha, \alpha)$$

Output

$$\langle S, \alpha(x_1) \rangle \to \sigma(x_1, \alpha)$$

Lexicalization

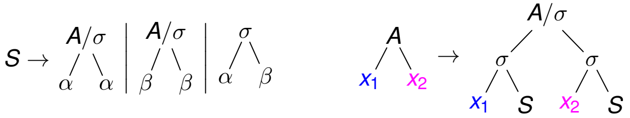
lexicalized CFTG

fully normalized CFTG

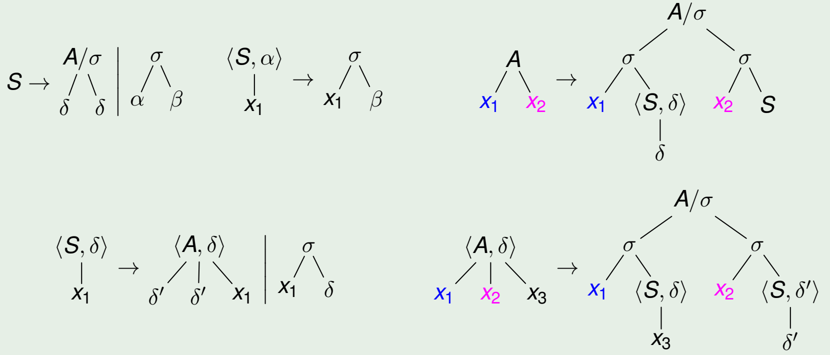
growing CFTG

CFTG

Lexicalization



After lexicalization (with $\delta, \delta' \in \{\alpha, \beta\}$)



Summary

CFTG(k): CFTG with nonterminals of rank $\leq k$

Theorem (ENGELFRIET et al. 1980)

CFTG(k) induces infinite hierarchy of string languages

Summary

CFTG(k): CFTG with nonterminals of rank $\leq k$

$$
\begin{array}{ll}
\text{CFTG}(k+1) & \\
\uparrow & \\
\text{CFTG}(k) & \\
\uparrow & \\
\cdots & \\
\uparrow & \\
\text{CFTG}(2) & \\
\uparrow & \\
\text{CFTG}(1) & \qquad \text{includes TAG} \\
\uparrow & \\
\text{CFTG}(0) & \qquad \text{regular tree grammar}
\end{array}
$$

Summary

CFTG(k): CFTG with nonterminals of rank $\leq k$

CFTG($k+1$)

CFTG(k) strongly lexicalized by

. . .

CFTG(2)

CFTG(1) includes TAG

CFTG(0) regular tree grammar

Corollary

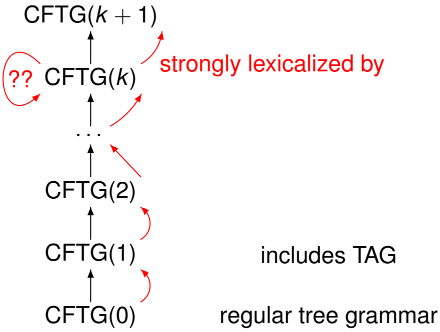
CFTG($k+1$) strongly lexicalize CFTG(k)

Corollary

CFTG(2) strongly lexicalize TAGs

Summary

CFTG(k): CFTG with nonterminals of rank $\leq k$



Corollary

CFTG($k+1$) strongly lexicalize CFTG(k)

Corollary

CFTG(2) strongly lexicalize TAGs

Open problem

Rank increase necessary?

Literature

Selected references

- JOSHI, LEVY, TAKAHASHI: *Tree Adjunct Grammars*
  J. Comput. System Sci. 10, 1975
- KUHLMANN, SATTA
  Tree-adjoining Grammars are not Closed under Strong Lexicalization. Comput. Linguist. 38, 2012
- ROUNDS: *Context-free Grammars on Trees*
  Proc. 1st STOC 1969
- SCHABES: *Mathematical and Computational Aspects of Lexicalized Grammars*
  Ph.D. thesis. University of Pennsylvania, 1990