Strong Lexicalization of Tree-Adjoining Grammars

Andreas Maletti[1] and Joost Engelfriet[2]

[1] IMS, Universität Stuttgart, Germany

[2] LIACS, Leiden University, The Netherlands

maletti@ims.uni-stuttgart.de

Jeju, Korea — July 10, 2012

Universität Stuttgart

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

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

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]

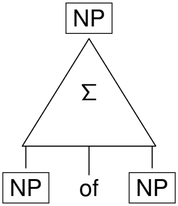
TAG — Syntax

Definition (JOSHI et al. 1969)

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

- substitution productions
- adjunction productions

Example (substitution production)



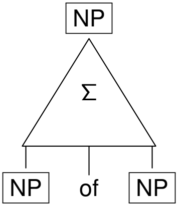
TAG — Syntax

Definition (Joshi et al. 1969)

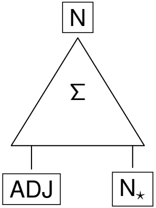
$G = (N, \Sigma, S, R)$ tree-adjoining grammar (TAG) with finite set R

- substitution productions
- adjunction productions

Example (substitution production)



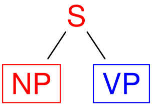
Example (adjunction production)



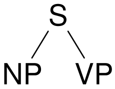
TAG — Example Derivation

S

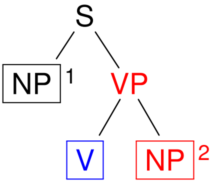
TAG — Example Derivation



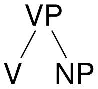
Used substitution production



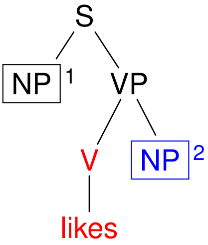
TAG — Example Derivation



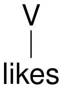
Used substitution production



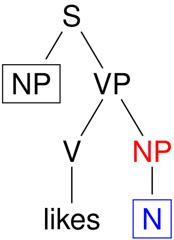
TAG — Example Derivation



Used substitution production

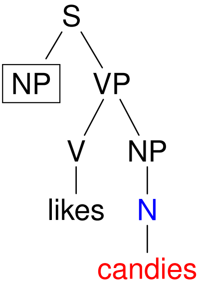


TAG — Example Derivation

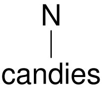


Used substitution production

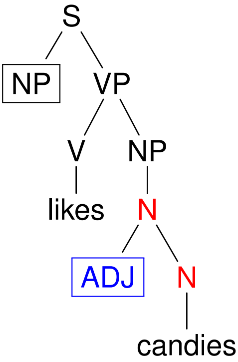
TAG — Example Derivation



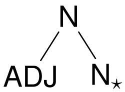
Used substitution production



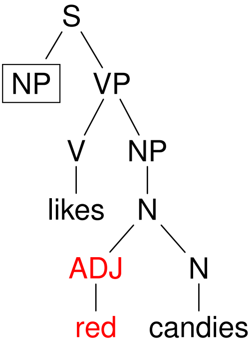
TAG — Example Derivation



Used adjunction production



TAG — Example Derivation



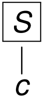
Used substitution production

TAG — Semantics

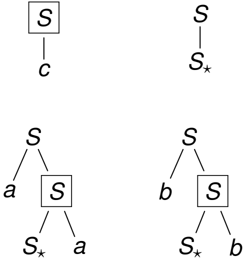
Definition (generated language)

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

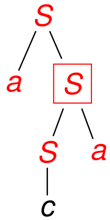
TAG — More Than CFG



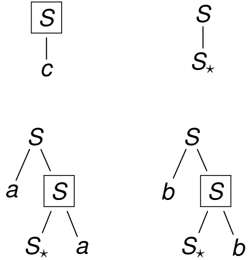
Example (productions)



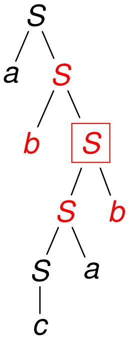
TAG — More Than CFG



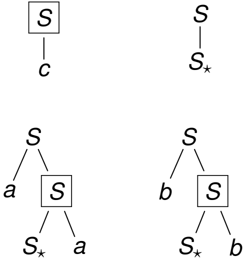
Example (productions)



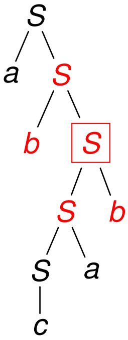
TAG — More Than CFG



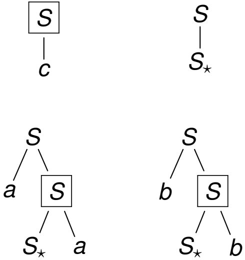
Example (productions)



TAG — More Than CFG



Example (productions)



String language

$$\{wcw \mid w \in \Sigma^*\}$$

TAG — Lexicalization

Definition

A TAG is lexicalized if each production contains a lexical item

Theorem (SCHABES 1990)

TAG can strongly lexicalize CFG and themselves

Widespread myth

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

TAG — Lexicalization

Definition

A TAG is lexicalized if each production contains a lexical item

Theorem (SCHABES 1990)

TAG can strongly lexicalize CFG and themselves

Widespread myth

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

TAG — Lexicalization

Definition

A TAG is lexicalized if each production contains a lexical item

Theorem (SCHABES 1990 and KUHLMANN, SATTA 2012)

TAG can strongly lexicalize CFG ~~and themselves~~ but not themselves

Widespread myth

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

TAG — Lexicalization

Definition

A TAG is lexicalized if each production contains a lexical item

Theorem (SCHABES 1990 and KUHLMANN, SATTA 2012)

TAG can strongly lexicalize CFG ~~and themselves~~ but not themselves

Widespread myth

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

Overview

Context-free Tree Grammar

Definition (Rounds 1969)

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

- ranked alphabet N — *nonterminals*
- ranked alphabet Σ — *terminals*
- $S \in N_0$ — *start nonterminal*
- P is a finite set of $A(x_1, \dots, x_k) \to r$ — *productions*
    - $A \in N_k$
    - $r \in C_{N \cup \Sigma}(\{x_1, \dots, x_k\})$

CFTG — Example

Example

CFTG (N, Σ, S, P)

- $N = \{S^{(0)}, A^{(2)}\}$
- $\Sigma = \{\alpha^{(0)}, \beta^{(0)}, \sigma^{(2)}\}$

productions

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

$$A(x_1, x_2) \to A(\sigma(x_1, S), \sigma(x_2, S))$$

$$A(x_1, x_2) \to \sigma(x_1, x_2)$$

CFTG — Example

Example

CFTG (N, Σ, S, P)

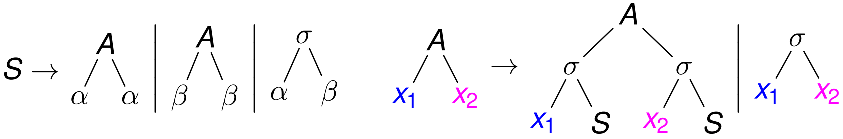
- $N = \{S^{(0)}, A^{(2)}\}$
- $\Sigma = \{\alpha^{(0)}, \beta^{(0)}, \sigma^{(2)}\}$

productions

$$S \rightarrow A(\alpha, \alpha) \mid A(\beta, \beta) \mid \sigma(\alpha, \beta)$$

$$A(x_1, x_2) \rightarrow A(\sigma(x_1, S), \sigma(x_2, S))$$

$$A(x_1, x_2) \rightarrow \sigma(x_1, x_2)$$



CFTG — Example

Example

CFTG (N, Σ, S, P)

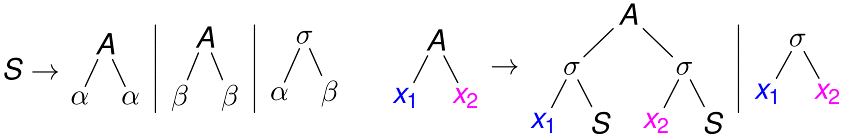
- $N = \{S^{(0)}, A^{(2)}\}$
- $\Sigma = \{\alpha^{(0)}, \beta^{(0)}, \sigma^{(2)}\}$

productions

$$S \rightarrow A(\alpha, \alpha) \mid A(\beta, \beta) \mid \sigma(\alpha, \beta)$$

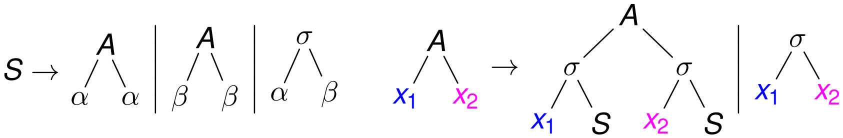
$$A(x_1, x_2) \rightarrow A(\sigma(x_1, S), \sigma(x_2, S))$$

$$A(x_1, x_2) \rightarrow \sigma(x_1, x_2)$$



CFTG — Derivation Example

Example



S

CFTG — Derivation Example

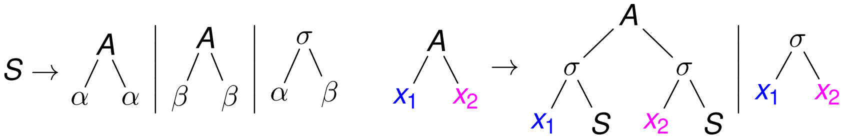
Example

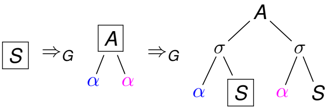
$$S \to A(\alpha, \alpha) \mid A(\beta, \beta) \mid \sigma(\alpha, \beta) \qquad A(x_1, x_2) \to A(\sigma(x_1, S), \sigma(x_2, S)) \mid \sigma(x_1, x_2)$$

$$\boxed{S} \Rightarrow_G \boxed{A}(\alpha, \alpha)$$

CFTG — Derivation Example

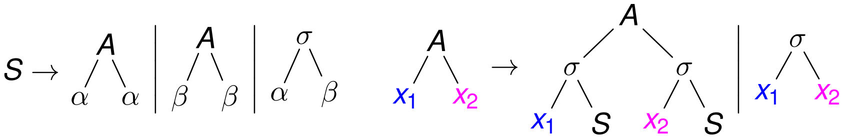
Example

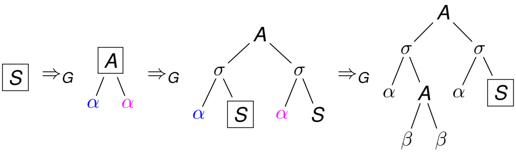




CFTG — Derivation Example

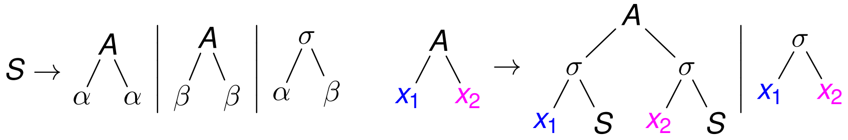
Example

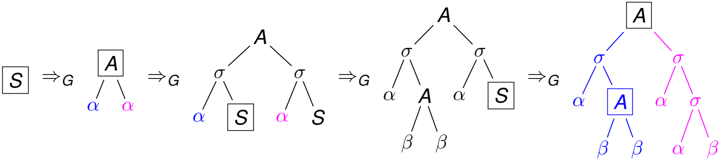




CFTG — Derivation Example

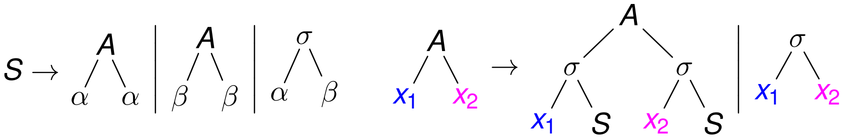
Example

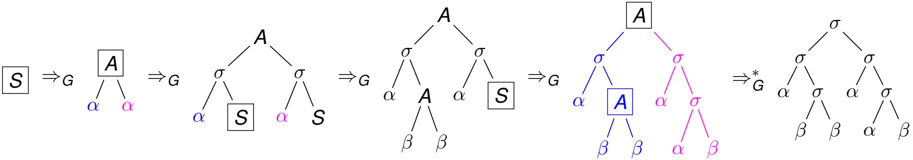




CFTG — Derivation Example

Example





CFTG — Semantics

Definition (generated language)

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

CFTG — Semantics

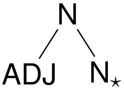
Definition (generated language)

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

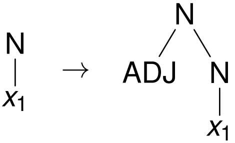
Theorem (Joshi et al. 1975 and Mönnich 1997)

Every (non-strict) TAG can be simulated by a CFTG

Example (adjunction production)



Example (CFTG production)



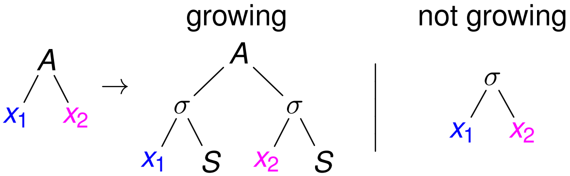
Overview

Growing Normal Form

Definition

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

Example



Theorem (Stamer, Otto 2007)

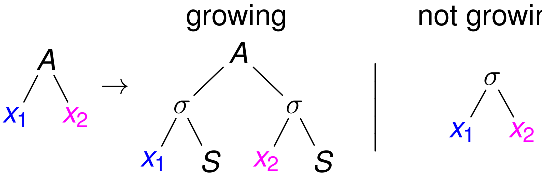
Every CFTG can be simulated by a growing CFTG

Growing Normal Form

Definition

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

Example



Theorem (Stamer, Otto 2007)

Every CFTG can be simulated by a growing CFTG

Growing Normal Form

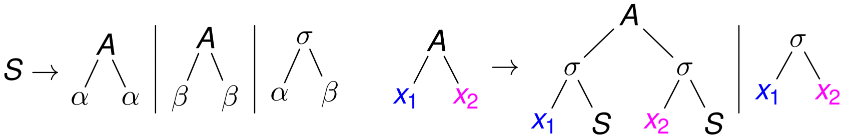
growing CFTG

↑

CFTG

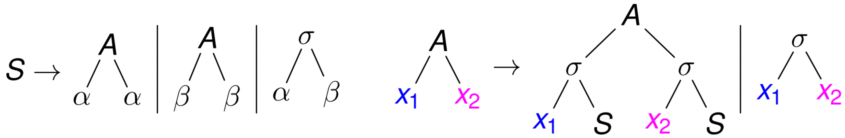
Growing Normal Form

Example

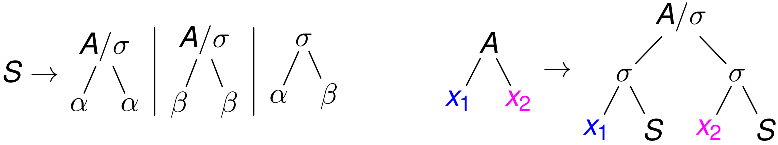


Growing Normal Form

Example



Eliminate last production:



Limited Ambiguity

Definition (frontier yield)

$\text{yd}\colon T_\Sigma \to \Sigma_0^*$

$$\text{yd}(\alpha) = \alpha$$

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

Definition (Schabes 1990)

$L \subseteq T_\Sigma$ finite ambiguity if $\{t \in L \mid \text{yd}(t) = w\}$ finite for all $w \in \Sigma_0^*$

Limited Ambiguity

Definition (frontier yield)

$\mathrm{yd}\colon T_\Sigma \to \Sigma_0^*$

$$\mathrm{yd}(\alpha) = \alpha$$

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

Definition (Schabes 1990)

$L \subseteq T_\Sigma$ finite ambiguity if $\{t \in L \mid \mathrm{yd}(t) = w\}$ finite for all $w \in \Sigma_0^*$

Monadic, Terminal, and Lexicalized Productions

Definition

Production $\ell \to r$

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

Theorem

Every CFTG with finite ambiguity can be simulated by a CFTG

- *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 □

Monadic, Terminal, and Lexicalized Productions

Definition

Production $\ell \to r$

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

Theorem

Every CFTG with finite ambiguity can be simulated by a CFTG

- *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

□

Monadic, Terminal, and Lexicalized Productions

Definition

Production $\ell \to r$

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

Theorem

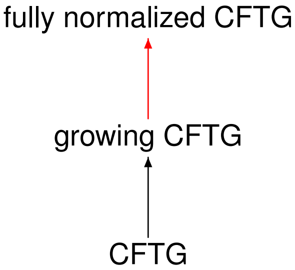
Every CFTG with finite ambiguity can be simulated by a CFTG

- *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 □

Limited Ambiguity



Overview

Lexicalization

Definition

A CFTG is lexicalized if each (non-initial) production is lexicalized

Theorem

Every CFTG with finite ambiguity can be lexicalized

Lexicalization

Definition

A CFTG is lexicalized if each (non-initial) production is lexicalized

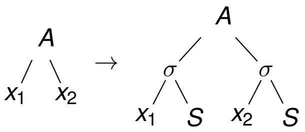
Theorem

Every CFTG with finite ambiguity can be lexicalized

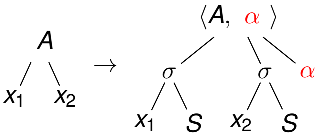
Lexicalization — Step 1

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

Input



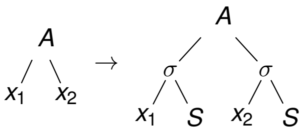
Output



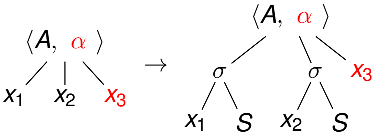
Lexicalization — Step 2

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

Input



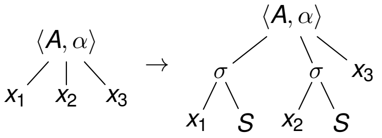
Output



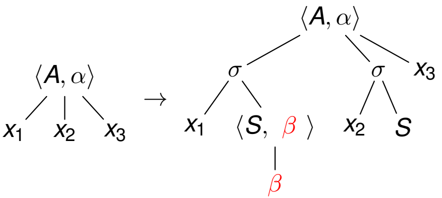
Lexicalization — Step 3

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

Input



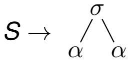
Output



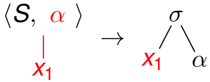
Lexicalization — Step 4

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

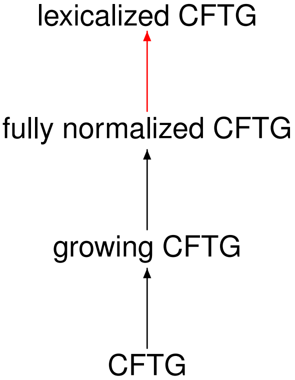
Input



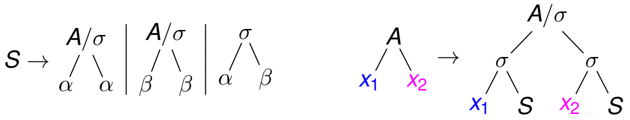
Output



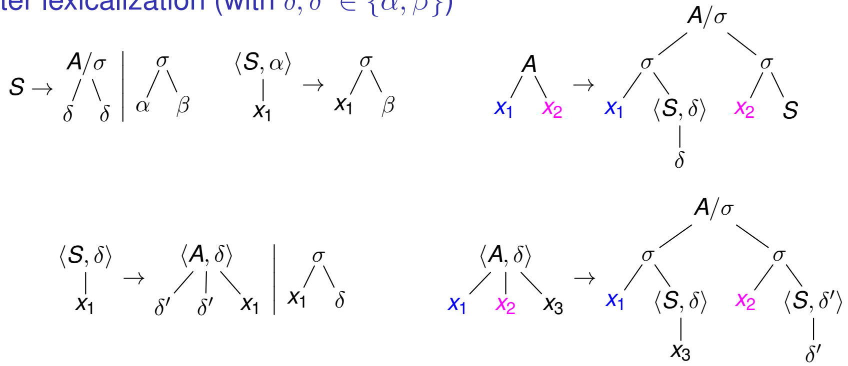
Lexicalization



Lexicalization — Example



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 $\leq k$

CFTG($k+1$)
↑
CFTG(k)
↑
...
↑
CFTG(2)
↑
CFTG(1) — includes TAG
↑
CFTG(0) — regular tree grammar

Corollary

CFTG(k) are strongly lexicalized by CFTG($k+1$)

Corollary

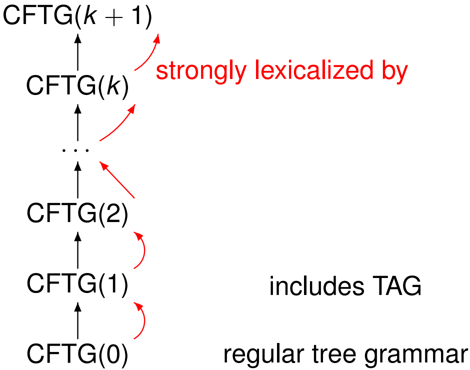
TAG are strongly lexicalized by CFTG(2)

Open problem

Rank increase necessary?

Summary

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



Corollary

CFTG(k) are strongly lexicalized by CFTG($k+1$)

Corollary

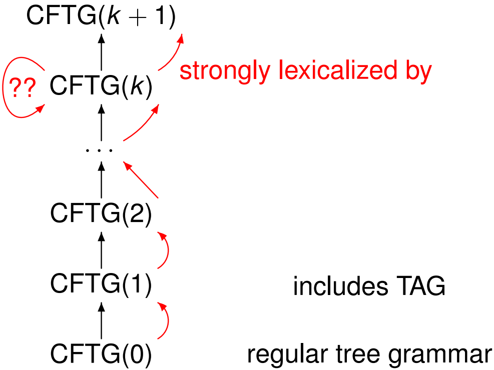
TAG are strongly lexicalized by CFTG(2)

Open problem

Rank increase necessary?

Summary

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



Corollary

CFTG(k) are strongly lexicalized by CFTG($k+1$)

Corollary

TAG are strongly lexicalized by CFTG(2)

Open problem

Rank increase necessary?

References

- ENGELFRIET, ROZENBERG, SLUTZKI: Tree transducers, L systems, and two-way machines. *J. Comput. System Sci.* 20(2), 1980
- HOPCROFT, MOTWANI, ULLMAN: *Introduction to automata theory, languages, and computation.* Addison-Wesley, 2001
- JOSHI, KOSARAJU, YAMADA: String adjunct grammars. In *SWAT* 1969
- JOSHI, LEVY, TAKAHASHI: Tree adjunct grammars. *J. Comput. System Sci.* 10(1), 1975
- KALLMEYER: *A lexicalized tree-adjoining grammar for a fragment of German focussing on syntax and semantics.* Emmy-Noether research group 2010
- KUHLMANN, SATTA: Tree-adjoining grammars are not closed under strong lexicalization. *Comput. Linguist.* 2012 (to appear)
- MÖNNICH: Adjunction as substitution: an algebraic formulation of regular, context-free and tree adjoining languages. In *FG* 1997
- ROUNDS: Context-free grammars on trees. In *STOC* 1969
- SCHABES: *Mathematical and computational aspects of lexicalized grammars.* Ph.D. thesis. University of Pennsylvania, 1990
- STAMER, OTTO: Restarting tree automata and linear context-free tree languages. In *CAI* 2007
- XTAG RESEARCH GROUP: *A Lexicalized Tree Adjoining Grammar for English.* Techn. Report IRCS-01-03. University of Pennsylvania, 2001