Strong Lexicalization of Tree Adjoining Grammars

Andreas Maletti[*]
IMS, Universität Stuttgart
Pfaffenwaldring 5b
70569 Stuttgart, Germany
maletti@ims.uni-stuttgart.de

Joost Engelfriet
LIACS, Leiden University
P.O. Box 9512
2300 RA Leiden, The Netherlands
engelfri@liacs.nl

Abstract

Recently, it was shown (KUHLMANN, SATTA: *Tree-adjoining grammars are not closed under strong lexicalization*. Comput. Linguist., 2012) that finitely ambiguous tree adjoining grammars cannot be transformed into a normal form (preserving the generated tree language), in which each production contains a lexical symbol. A more powerful model, the simple context-free tree grammar, admits such a normal form. It can be effectively constructed and the maximal rank of the nonterminals only increases by 1. Thus, simple context-free tree grammars strongly lexicalize tree adjoining grammars and themselves.

1 Introduction

Tree adjoining grammars [TAG] (Joshi et al., 1969; Joshi et al., 1975) are a mildly context-sensitive grammar formalism that can handle certain nonlocal dependencies (Kuhlmann and Mohl, 2006), which occur in several natural languages. A good overview on TAG, their formal properties, their linguistic motivation, and their applications is presented by Joshi and Schabes (1992) and Joshi and Schabes (1997), in which also strong lexicalization is discussed. In general, lexicalization is the process of transforming a grammar into an equivalent one (potentially expressed in another formalism) such that each production contains a lexical item (or anchor). Each production can then be viewed as lexical information on its anchor. It demonstrates a syntactical construction in which the anchor can occur. Since a lexical item is a letter of the string alphabet, each production of a lexicalized grammar produces at least one letter of the generated string. Consequently, lexicalized grammars offer significant parsing benefits (Schabes et al., 1988) as the number of applications of productions (i.e., derivation steps) is clearly bounded by the length of the input string. In addition, the lexical items in the productions guide the production selection in a derivation, which works especially well in scenarios with large alphabets.[1] The GREIBACH normal form (Hopcroft et al., 2001; Blum and Koch, 1999) offers those benefits for context-free grammars [CFG], but it changes the parse trees. Thus, we distinguish between two notions of equivalence: *Weak equivalence* (Bar-Hillel et al., 1960) only requires that the generated string languages coincide, whereas *strong equivalence* (Chomsky, 1963) requires that even the generated tree languages coincide. Correspondingly, we obtain weak and strong lexicalization based on the required equivalence.

The GREIBACH normal form shows that CFG can weakly lexicalize themselves, but they cannot strongly lexicalize themselves (Schabes, 1990). It is a prominent feature of tree adjoining grammars that they can strongly lexicalize CFG (Schabes, 1990),[2] and it was claimed and widely believed that they can strongly lexicalize themselves. Recently, Kuhlmann and Satta (2012) proved that TAG actually cannot strongly lexicalize themselves. In fact, they prove that TAG cannot even strongly lexicalize the weaker tree insertion grammars (Schabes and Waters, 1995). However, TAG can weakly lexicalize themselves (Fujiyoshi, 2005).

[*] Financially supported by the German Research Foundation (DFG) grant MA 4959 / 1-1.

[1] Chen (2001) presents a detailed account.

[2] Good algorithmic properties and the good coverage of linguistic phenomena are other prominent features.

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 506–515,
Jeju, Republic of Korea, 8-14 July 2012.

Simple (i.e., linear and nondeleting) context-free tree grammars [CFTG] (Rounds, 1969; Rounds, 1970) are a more powerful grammar formalism than TAG (Mönnich, 1997). However, the monadic variant is strongly equivalent to a slightly extended version of TAG, which is called non-strict TAG (Kepser and Rogers, 2011). A GREIBACH normal form for a superclass of CFTG (viz., second-order abstract categorial grammars) was discussed by Kanazawa and Yoshinaka (2005) and Yoshinaka (2006). In particular, they also demonstrate that monadic CFTG can strongly lexicalize regular tree grammars (Gécseg and Steinby, 1984; Gécseg and Steinby, 1997).

CFTG are weakly equivalent to the simple macro grammars of Fischer (1968), which are a notational variant of the well-nested linear context-free rewriting systems (LCFRS) of Vijay-Shanker et al. (1987) and the well-nested multiple context-free grammars (MCFG) of Seki et al. (1991).[3] Thus, CFTG are mildly context-sensitive since their generated string languages are semi-linear and can be parsed in polynomial time (Gómez-Rodríguez et al., 2010).

In this contribution, we show that CFTG can strongly lexicalize TAG and also themselves, thus answering the second question in the conclusion of Kuhlmann and Satta (2012). This is achieved by a series of normalization steps (see Section 4) and a final lexicalization step (see Section 5), in which a lexical item is guessed for each production that does not already contain one. This item is then transported in an additional argument until it is exchanged for the same item in a terminal production. The lexicalization is effective and increases the maximal rank (number of arguments) of the nonterminals by at most 1. In contrast to a transformation into GREIBACH normal form, our lexicalization does not radically change the structure of the derivations. Overall, our result shows that if we consider only lexicalization, then CFTG are a more natural generalization of CFG than TAG.

2 Notation

We write $[k]$ for the set $\{i \in \mathbb{N} \mid 1 \leq i \leq k\}$, where $\mathbb{N}$ denotes the set of nonnegative integers. We use a fixed countably infinite set $X = \{x_1, x_2, \dots\}$ of (mutually distinguishable) variables, and we let $X_k = \{x_i \mid i \in [k]\}$ be the first k variables from X for every $k \in \mathbb{N}$. As usual, an alphabet Σ is a finite set of symbols, and a ranked alphabet (Σ, rk) adds a ranking $\mathrm{rk}\colon \Sigma \to \mathbb{N}$. We let $\Sigma_k = \{\sigma \mid \mathrm{rk}(\sigma) = k\}$ be the set of k-ary symbols. Moreover, we just write Σ for the ranked alphabet (Σ, rk).[4] We build trees over the ranked alphabet Σ such that the nodes are labeled by elements of Σ and the rank of the node label determines the number of its children. In addition, elements of X can label leaves. Formally, the set $T_\Sigma(X)$ of Σ-trees indexed by X is the smallest set T such that $X \subseteq T$ and $\sigma(t_1, \dots, t_k) \in T$ for all $k \in \mathbb{N}$, $\sigma \in \Sigma_k$, and $t_1, \dots, t_k \in T$.[5]

We use positions to address the nodes of a tree. A position is a sequence of nonnegative integers indicating successively in which subtree the addressed node is. More precisely, the root is at position ε and the position ip with $i \in \mathbb{N}$ and $p \in \mathbb{N}^*$ refers to the position p in the i^{th} direct subtree. Formally, the set $\mathrm{pos}(t) \subseteq \mathbb{N}^*$ of positions of a tree $t \in T_\Sigma(X)$ is defined by $\mathrm{pos}(x) = \{\varepsilon\}$ for $x \in X$ and

$$\mathrm{pos}(\sigma(t_1, \dots, t_k)) = \{\varepsilon\} \cup \{ip \mid i \in [k], p \in \mathrm{pos}(t_i)\}$$

for all symbols $\sigma \in \Sigma_k$ and $t_1, \dots, t_k \in T_\Sigma(X)$. The positions are indicated as superscripts of the labels in the tree of Figure 1. The subtree of t at position $p \in \mathrm{pos}(t)$ is denoted by $t|_p$, and the label of t at position p by $t(p)$. Moreover, $t[u]_p$ denotes the tree obtained from t by replacing the subtree at p by the tree $u \in T_\Sigma(X)$. For every label set $S \subseteq \Sigma$, we let $\mathrm{pos}_S(t) = \{p \in \mathrm{pos}(t) \mid t(p) \in S\}$ be the S-labeled positions of t. For every $\sigma \in \Sigma$, we let $\mathrm{pos}_\sigma(t) = \mathrm{pos}_{\{\sigma\}}(t)$. The set $C_\Sigma(X_k)$ contains all trees t of $T_\Sigma(X)$, in which every $x \in X_k$ occurs exactly once and $\mathrm{pos}_{X \setminus X_k}(t) = \emptyset$. Given $u_1, \dots, u_k \in T_\Sigma(X)$, the first-order substitution $t[u_1, \dots, u_k]$ is inductively defined by

$$x_i[u_1, \dots, u_k] = \begin{cases} u_i & \text{if } i \in [k] \\ x_i & \text{otherwise} \end{cases}$$

$$t[u_1, \dots, u_k] = \sigma\big(t_1[u_1, \dots, u_k], \dots, t_k[u_1, \dots, u_k]\big)$$

for every $i \in \mathbb{N}$ and $t = \sigma(t_1, \dots, t_k)$ with $\sigma \in \Sigma_k$ and $t_1, \dots, t_k \in T_\Sigma(X)$. First-order substitution is illustrated in Figure 1.

[3] Kuhlmann (2010), Mönnich (2010), and Kanazawa (2009) discuss well-nestedness.

[4] We often decorate a symbol σ with its rank k [e.g. $\sigma^{(k)}$].

[5] We will often drop quantifications like 'for all $k \in \mathbb{N}$'.

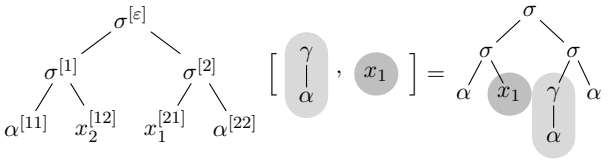

Figure 1: Tree in $C_\Sigma(X_2) \subset T_\Sigma(X)$ with indicated positions, where $\Sigma = \{\sigma, \gamma, \alpha\}$ with $\mathrm{rk}(\sigma) = 2$, $\mathrm{rk}(\gamma) = 1$, and $\mathrm{rk}(\alpha) = 0$, and an example first-order substitution.

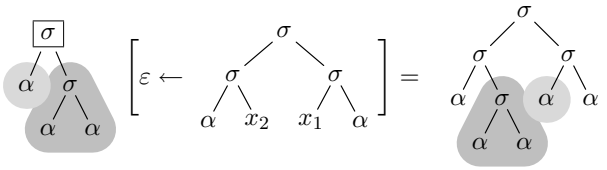

Figure 2: Example second-order substitution, in which the boxed symbol σ is replaced.

In first-order substitution we replace leaves (elements of X), whereas in second-order substitution we replace an internal node (labeled by a symbol of Σ). Let $p \in \mathrm{pos}(t)$ be such that $t(p) \in \Sigma_k$, and let $u \in C_\Sigma(X_k)$ be a tree in which the variables X_k occur exactly once. The second-order substitution $t[p \leftarrow u]$ replaces the subtree at position p by the tree u into which the children of p are (first-order) substituted. In essence, u is "folded" into t at position p. Formally, $t[p \leftarrow u] = t\big[u[t|_1, \ldots, t|_k]\big]_p$. Given $P \subseteq \mathrm{pos}_\sigma(t)$ with $\sigma \in \Sigma_k$, we let $t[P \leftarrow u]$ be $t[p_1 \leftarrow u] \cdots [p_n \leftarrow u]$, where $P = \{p_1, \ldots, p_n\}$ and $p_1 > \cdots > p_n$ in the lexicographic order. Second-order substitution is illustrated in Figure 2. Gécseg and Steinby (1997) present a detailed introduction to trees and tree languages.

3 Context-free tree grammars

In this section, we recall linear and nondeleting context-free tree grammars [CFTG] (Rounds, 1969; Rounds, 1970). The property 'linear and nondeleting' is often called 'simple'. The nonterminals of regular tree grammars only occur at the leaves and are replaced using first-order substitution. In contrast, the nonterminals of a CFTG are ranked symbols, can occur anywhere in a tree, and are replaced using second-order substitution.[6] Consequently, the nonterminals N of a CFTG form a ranked alphabet. In the left-hand sides of productions we write $A(x_1, \ldots, x_k)$ for a nonterminal $A \in N_k$ to indicate the variables that hold the direct subtrees of a particular occurrence of A.

Definition 1. A *(simple) context-free tree grammar* [CFTG] is a system (N, Σ, S, P) such that

- N is a ranked alphabet of *nonterminal symbols*,
- Σ is a ranked alphabet of *terminal symbols*,[7]
- $S \in N_0$ is the *start nonterminal* of rank 0, and
- P is a finite set of *productions* of the form $A(x_1, \ldots, x_k) \to r$, where $r \in C_{N \cup \Sigma}(X_k)$ and $A \in N_k$.

The components ℓ and r are called left- and right-hand side of the production $\ell \to r$ in P. We say that it is an A-production if $\ell = A(x_1, \ldots, x_k)$. The right-hand side is simply a tree using terminal and nonterminal symbols according to their rank. Moreover, it contains all the variables of X_k exactly once. Let us illustrate the syntax on an example CFTG. We use an abstract language for simplicity and clarity. We use lower-case Greek letters for terminal symbols and upper-case Latin letters for nonterminals.

Example 2. As a running example, we consider the CFTG $G_{\mathrm{ex}} = (\{S^{(0)}, A^{(2)}\}, \Sigma, S, P)$ where

- $\Sigma = \{\sigma^{(2)}, \alpha^{(0)}, \beta^{(0)}\}$ and
- P contains the productions (see Figure 3):[8]

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

We recall the (term) rewrite semantics (Baader and Nipkow, 1998) of the CFTG $G = (N, \Sigma, S, P)$. Since G is simple, the actual rewriting strategy is irrelevant. The sentential forms of G are simply $\mathrm{SF}(G) = T_{N \cup \Sigma}(X)$. This is slightly more general than necessary (for the semantics of G), but the presence of variables in sentential forms will be useful in the next section because it allows us to treat right-hand sides as sentential forms. In essence in a rewrite step we just select a nonterminal $A \in N$ and an A-production $\rho \in P$. Then we replace an occurrence of A in the sentential form by the right-hand side of ρ using second-order substitution.

Definition 3. Let $\xi, \zeta \in \mathrm{SF}(G)$ be sentential forms. Given an A-production $\rho = \ell \to r$ in P and an

[6] see Sections 6 and 15 of (Gécseg and Steinby, 1997)

[7] We assume that $\Sigma \cap N = \emptyset$.

[8] We separate several right-hand sides with '|'.

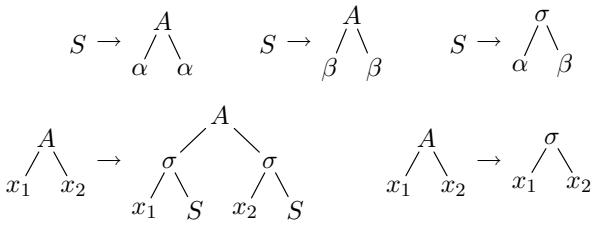

Figure 3: Productions of Example 2.

A-labeled position $p \in \mathrm{pos}_A(\xi)$ in ξ, we write $\xi \Rightarrow_G^{\rho,p} \xi[p \leftarrow r]$. If there exist $\rho \in P$ and $p \in \mathrm{pos}(\xi)$ such that $\xi \Rightarrow_G^{\rho,p} \zeta$, then $\xi \Rightarrow_G \zeta$.[9] The semantics $[\![G]\!]$ of G is $\{t \in T_\Sigma \mid S \Rightarrow_G^* t\}$, where $\Rightarrow_G^*$ is the reflexive, transitive closure of $\Rightarrow_G$.

Two CFTG G_1 and G_2 are (strongly) equivalent if $[\![G_1]\!] = [\![G_2]\!]$. In this contribution we are only concerned with strong equivalence (Chomsky, 1963). Although we recall the string corresponding to a tree later on (via its yield), we will not investigate weak equivalence (Bar-Hillel et al., 1960).

Example 4. Reconsider the CFTG G_{ex} of Example 2. A derivation to a tree of T_Σ is illustrated in Figure 4. It demonstrates that the final tree in that derivation is in the language $[\![G_{\mathrm{ex}}]\!]$ generated by G_{ex}.

Finally, let us recall the relation between CFTG and tree adjoining grammars [TAG] (Joshi et al., 1969; Joshi et al., 1975). Joshi et al. (1975) show that TAG are special footed CFTG (Kepser and Rogers, 2011), which are weakly equivalent to monadic CFTG, i.e., CFTG whose nonterminals have rank at most 1 (Mönnich, 1997; Fujiyoshi and Kasai, 2000). Kepser and Rogers (2011) show the strong equivalence of those CFTG to non-strict TAG, which are slightly more powerful than traditional TAG. In general, TAG are a natural formalism to describe the syntax of natural language.[10]

4 Normal forms

In this section, we first recall an existing normal form for CFTG. Then we introduce the property of finite ambiguity in the spirit of (Schabes, 1990; Joshi and Schabes, 1992; Kuhlmann and Satta, 2012), which allows us to normalize our CFTG even further. A major tool is a simple production elimination scheme, which we present in detail. From now on, let $G = (N, \Sigma, S, P)$ be the considered CFTG.

The CFTG G is *start-separated* if $\mathrm{pos}_S(r) = \emptyset$ for every production $\ell \to r \in P$. In other words, the start nonterminal S is not allowed in the right-hand sides of the productions. It is clear that each CFTG can be transformed into an equivalent start-separated CFTG. In such a CFTG we call each production of the form $S \to r$ *initial*. From now on, we assume, without loss of generality, that G is start-separated.

Example 5. Let $G_{\mathrm{ex}} = (N, \Sigma, S, P)$ be the CFTG of Example 2. An equivalent start-separated CFTG is $G'_{\mathrm{ex}} = (\{S'^{(0)}\} \cup N, \Sigma, S', P \cup \{S' \to S\})$.

We start with the growing normal form of Stamer and Otto (2007) and Stamer (2009). It requires that the right-hand side of each non-initial production contains at least two terminal or nonterminal symbols. In particular, it eliminates projection productions $A(x_1) \to x_1$ and unit productions, in which the right-hand side has the same shape as the left-hand side (potentially with a different root symbol and a different order of the variables).

Definition 6. A production $\ell \to r$ is *growing* if $|\mathrm{pos}_{N\cup\Sigma}(r)| \geq 2$. The CFTG G is growing if all of its non-initial productions are growing.

The next theorem is Proposition 2 of (Stamer and Otto, 2007). Stamer (2009) provides a full proof.

Theorem 7. For every start-separated CFTG there exists an equivalent start-separated, growing CFTG.

Example 8. Let us transform the CFTG G'_{ex} of Example 5 into growing normal form. We obtain the CFTG $G''_{\mathrm{ex}} = (\{S'^{(0)}, S^{(0)}, A^{(2)}\}, \Sigma, S', P'')$ where P'' contains $S' \to S$ and for each $\delta \in \{\alpha, \beta\}$

$$S \to A(\delta,\delta) \mid \sigma(\delta,\delta) \mid \sigma(\alpha,\beta) \quad (1)$$
$$A(x_1,x_2) \to A(\sigma(x_1,S),\sigma(x_2,S)) \quad (2)$$
$$A(x_1,x_2) \to \sigma(\sigma(x_1,S),\sigma(x_2,S)) \ .$$

From now on, we assume that G is growing. Next, we recall the notion of finite ambiguity from (Schabes, 1990; Joshi and Schabes, 1992; Kuhlmann and Satta, 2012).[11] We distinguish a subset $\Delta \subseteq \Sigma_0$ of *lexical* symbols, which are the symbols that are preserved by the yield mapping. The yield of a tree is

[9] For all $k \in \mathbb{N}$ and $\xi \Rightarrow_G \zeta$ we note that $\xi \in C_{N\cup\Sigma}(X_k)$ if and only if $\zeta \in C_{N\cup\Sigma}(X_k)$.

[10] XTAG Research Group (2001) wrote a TAG for English.

[11] It should not be confused with the notion of 'finite ambiguity' of (Goldstine et al., 1992; Klimann et al., 2004).

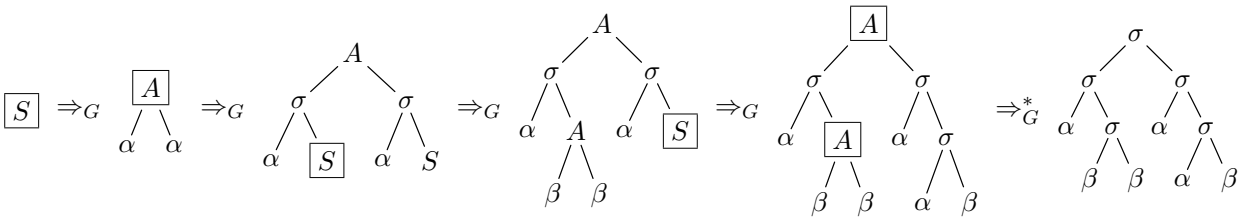

Figure 4: Derivation using the CFTG G_{ex} of Example 2. The selected positions are boxed.

a string of lexical symbols. All other symbols are simply dropped (in a pre-order traversal). Formally, $\mathrm{yd}_\Delta : T_\Sigma \to \Delta^*$ is such that for all $t = \sigma(t_1, \ldots, t_k)$ with $\sigma \in \Sigma_k$ and $t_1, \ldots, t_k \in T_\Sigma$

$$\mathrm{yd}_\Delta(t) = \begin{cases} \sigma\, \mathrm{yd}_\Delta(t_1) \cdots \mathrm{yd}_\Delta(t_k) & \text{if } \sigma \in \Delta \\ \mathrm{yd}_\Delta(t_1) \cdots \mathrm{yd}_\Delta(t_k) & \text{otherwise.} \end{cases}$$

Definition 9. The tree language $L \subseteq T_\Sigma$ has *finite Δ-ambiguity* if $\{t \in L \mid \mathrm{yd}_\Delta(t) = w\}$ is finite for every $w \in \Delta^*$.

Roughly speaking, we can say that the set L has finite Δ-ambiguity if each $w \in \Delta^*$ has finitely many parses in L (where t is a parse of w if $\mathrm{yd}_\Delta(t) = w$). Our example CFTG G_{ex} is such that $[\![G_{\text{ex}}]\!]$ has finite $\{\alpha, \beta\}$-ambiguity (because $\Sigma_1 = \emptyset$).

In this contribution, we want to (strongly) lexicalize CFTG, which means that for each CFTG G such that $[\![G]\!]$ has finite Δ-ambiguity, we want to construct an equivalent CFTG such that each non-initial production contains at least one lexical symbol. This is typically called strong lexicalization (Schabes, 1990; Joshi and Schabes, 1992; Kuhlmann and Satta, 2012) because we require strong equivalence.[12] Let us formalize our lexicalization property.

Definition 10. The production $\ell \to r$ is *Δ-lexicalized* if $\mathrm{pos}_\Delta(r) \neq \emptyset$. The CFTG G is Δ-lexicalized if all its non-initial productions are Δ-lexicalized.

Note that the CFTG G''_{ex} of Example 8 is not yet $\{\alpha, \beta\}$-lexicalized. We will lexicalize it in the next section. To do this in general, we need some auxiliary normal forms. First, we define our simple production elimination scheme, which we will use in the following. Roughly speaking, a non-initial A-production such that A does not occur in its right-hand side can be eliminated from G by applying it in all possible ways to occurrences in right-hand sides of the remaining productions.

Definition 11. Let $\rho = A(x_1, \ldots, x_k) \to r$ in P be a non-initial production such that $\mathrm{pos}_A(r) = \emptyset$. For every other production $\rho' = \ell' \to r'$ in P and $J \subseteq \mathrm{pos}_A(r')$, let $\rho'_J = \ell' \to r'[J \leftarrow r]$. The CFTG $\mathrm{Elim}(G, \rho) = (N, \Sigma, S, P')$ is such that

$$P' = \bigcup_{\rho' = \ell' \to r' \in P \setminus \{\rho\}} \{\rho'_J \mid J \subseteq \mathrm{pos}_A(r')\} \ .$$

In particular, $\rho'_\emptyset = \rho'$ for every production ρ', so every production besides the eliminated production ρ is preserved. We obtained the CFTG G''_{ex} of Example 8 as $\mathrm{Elim}(G'_{\text{ex}}, A(x_1, x_2) \to \sigma(x_1, x_2))$ from G'_{ex} of Example 5.

Lemma 12. The CFTG G and $G'_\rho = \mathrm{Elim}(G, \rho)$ are equivalent for every non-initial A-production $\rho = \ell \to r$ in P such that $\mathrm{pos}_A(r) = \emptyset$.

Proof. Clearly, every single derivation step of G'_ρ can be simulated by a derivation of G using potentially several steps. Conversely, a derivation of G can be simulated directly by G'_ρ except for derivation steps $\Rightarrow_G^{\rho,p}$ using the eliminated production ρ. Since $S \neq A$, we know that the nonterminal at position p was generated by another production ρ'. In the given derivation of G we examine which nonterminals in the right-hand side of the instance of ρ' were replaced using ρ. Let J be the set of positions corresponding to those nonterminals (thus $p \in J$). Then instead of applying ρ' and potentially several times ρ, we equivalently apply ρ'_J of G'_ρ. □

In the next normalization step we use our production elimination scheme. The goal is to make sure that non-initial monic productions (i.e., productions of which the right-hand side contains at most one nonterminal) contain at least one lexical symbol. We define the relevant property and then present

[12] The corresponding notion for weak equivalence is called weak lexicalization (Joshi and Schabes, 1992).

the construction. A sentential form $\xi \in \mathrm{SF}(G)$ is *monic* if $|\mathrm{pos}_N(\xi)| \leq 1$. The set of all monic sentential forms is denoted by $\mathrm{SF}_{\leq 1}(G)$. A production $\ell \to r$ is monic if r is monic. The next construction is similar to the simultaneous removal of epsilon-productions $A \to \varepsilon$ and unit productions $A \to B$ for context-free grammars (Hopcroft et al., 2001). Instead of computing the closure under those productions, we compute a closure under non-Δ-lexicalized productions.

Theorem 13. If $[\![G]\!]$ has finite Δ-ambiguity, then there exists an equivalent CFTG such that all its non-initial monic productions are Δ-lexicalized.

Proof. Without loss of generality, we assume that G is start-separated and growing by Theorem 7. Moreover, we assume that each nonterminal is useful. For every $A \in N$ with $A \neq S$, we compute all monic sentential forms without a lexical symbol that are reachable from $A(x_1, \ldots, x_k)$, where $k = \mathrm{rk}(A)$. Formally, let

$$\Xi_A = \{\xi \in \mathrm{SF}_{\leq 1}(G) \mid A(x_1, \ldots, x_k) \Rightarrow_{G'}^{+} \xi\} \ ,$$

where $\Rightarrow_{G'}^{+}$ is the transitive closure of $\Rightarrow_{G'}$ and the CFTG $G' = (N, \Sigma, S, P')$ is such that P' contains exactly the non-Δ-lexicalized productions of P. The set Ξ_A is finite since only finitely many non-Δ-lexicalized productions can be used due to the finite Δ-ambiguity of $[\![G]\!]$. Moreover, no sentential form in Ξ_A contains A for the same reason and the fact that G is growing. We construct the CFTG $G_1 = (N, \Sigma, S, P \cup P_1)$ such that

$$P_1 = \{A(x_1, \ldots, x_k) \to \xi \mid A \in N_k, \xi \in \Xi_A\} \ .$$

Clearly, G and G_1 are equivalent. Next, we eliminate all productions of P_1 from G_1 using Lemma 12 to obtain an equivalent CFTG G_2 with the productions P_2. In the final step, we drop all non-Δ-lexicalized monic productions of P_2 to obtain the CFTG $\overline{G}$, in which all monic productions are Δ-lexicalized. It is easy to see that $\overline{G}$ is growing, start-separated, and equivalent to G_2. □

The CFTG G''_{ex} only has $\{\alpha, \beta\}$-lexicalized non-initial monic productions, so we use a new example.

Example 14. Let $(\{S^{(0)}, A^{(1)}, B^{(1)}\}, \Sigma, S, P)$ be the CFTG such that $\Sigma = \{\sigma^{(2)}, \alpha^{(0)}, \beta^{(0)}\}$ and

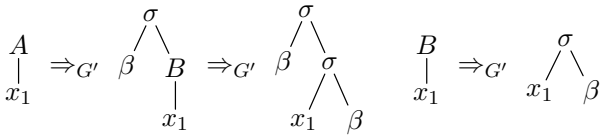

Figure 5: The relevant derivations using only productions that are not Δ-lexicalized (see Example 14).

P contains the productions

$$\begin{aligned} A(x_1) &\to \sigma(\beta, B(x_1)) \qquad B(x_1) \to \sigma(x_1, \beta) \\ B(x_1) &\to \sigma(\alpha, A(x_1)) \qquad S \to A(\alpha) \ . \end{aligned} \tag{3}$$

This CFTG G_{ex2} is start-separated and growing. Moreover, all its productions are monic, and $[\![G_{\text{ex2}}]\!]$ is finitely Δ-ambiguous for the set $\Delta = \{\alpha\}$ of lexical symbols. Then the productions (3) are non-initial and not Δ-lexicalized. So we can run the construction in the proof of Theorem 13. The relevant derivations using only non-Δ-lexicalized productions are shown in Figure 5. We observe that $|\Xi_A| = 2$ and $|\Xi_B| = 1$, so we obtain the CFTG $(\{S^{(0)}, B^{(1)}\}, \Sigma, S, P')$, where P' contains[13]

$$\begin{aligned} S &\to \sigma(\beta, B(\alpha)) \mid \sigma(\beta, \sigma(\alpha, \beta)) \\ B(x_1) &\to \sigma(\alpha, \sigma(\beta, B(x_1))) \\ B(x_1) &\to \sigma(\alpha, \sigma(\beta, \sigma(x_1, \beta))) \ . \end{aligned} \tag{4}$$

We now do one more normalization step before we present our lexicalization. We call a production $\ell \to r$ *terminal* if $r \in T_\Sigma(X)$; i.e., it does not contain nonterminal symbols. Next, we show that for each CFTG G such that $[\![G]\!]$ has finite Δ-ambiguity we can require that each non-initial terminal production contains at least two occurrences of Δ-symbols.

Theorem 15. If $[\![G]\!]$ has finite Δ-ambiguity, then there exists an equivalent CFTG (N, Σ, S, P') such that $|\mathrm{pos}_\Delta(r)| \geq 2$ for all its non-initial terminal productions $\ell \to r \in P'$.

Proof. Without loss of generality, we assume that G is start-separated and growing by Theorem 7. Moreover, we assume that each nonterminal is useful and that each of its non-initial monic productions is Δ-lexicalized by Theorem 13. We obtain the desired CFTG by simply eliminating each non-initial terminal production $\ell \to r \in P$ such that $|\mathrm{pos}_\Delta(r)| = 1$. By Lemma 12 the obtained CFTG

[13] The nonterminal A became useless, so we just removed it.

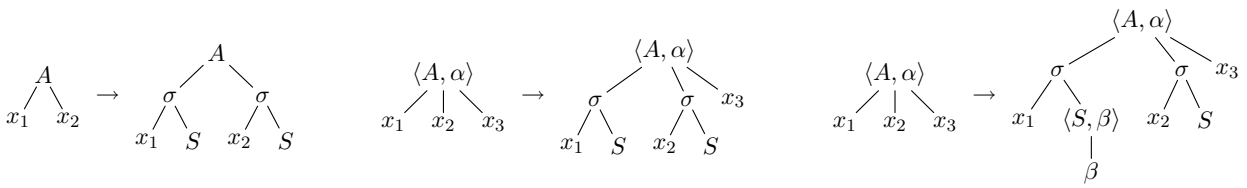

Figure 6: Production $\rho = \ell \to r$ of (2) [left], a corresponding production ρ_α of P' [middle] with right-hand side $r_{\alpha,2}$, and a corresponding production of P''' [right] with right-hand side $(\overline{r_{\alpha,2}})_\beta$ (see Theorem 17).

is equivalent to G. The elimination process terminates because a new terminal production can only be constructed from a monic production and a terminal production or several terminal productions, but those combinations already contain two occurrences of Δ-symbols since non-initial monic productions are already Δ-lexicalized. □

Example 16. Reconsider the CFTG obtained in Example 14. Recall that $\Delta = \{\alpha\}$. Production (4) is the only non-initial terminal production that violates the requirement of Theorem 15. We eliminate it and obtain the CFTG with the productions

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

5 Lexicalization

In this section, we present the main lexicalization step, which lexicalizes non-monic productions. We assume that $[\![G]\!]$ has finite Δ-ambiguity and is normalized according to the results of Section 4: no useless nonterminals, start-separated, growing (see Theorem 7), non-initial monic productions are Δ-lexicalized (see Theorem 13), and non-initial terminal productions contain at least two occurrences of Δ-symbols (see Theorem 15).

The basic idea of the construction is that we guess a lexical symbol for each non-Δ-lexicalized production. The guessed symbol is put into a new parameter of a nonterminal. It will be kept in the parameter until we reach a terminal production, where we exchange the same lexical symbol by the parameter. This is the reason why we made sure that we have two occurrences of lexical symbols in the terminal productions. After we exchanged one for a parameter, the resulting terminal production is still Δ-lexicalized. Lexical items that are guessed for distinct (occurrences of) productions are transported to distinct (occurrences of) terminal productions [cf. Section 3 of (Potthoff and Thomas, 1993) and page 346 of (Hoogeboom and ten Pas, 1997)].

Theorem 17. For every CFTG G such that $[\![G]\!]$ has finite Δ-ambiguity there exists an equivalent Δ-lexicalized CFTG.

Proof. We can assume that $G = (N, \Sigma, S, P)$ has the properties mentioned before the theorem without loss of generality. We let $N' = N \times \Delta$ be a new set of nonterminals such that $\mathrm{rk}(\langle A, \delta\rangle) = \mathrm{rk}(A) + 1$ for every $A \in N$ and $\delta \in \Delta$. Intuitively, $\langle A, \delta\rangle$ represents the nonterminal A, which has the lexical symbol δ in its last (new) parameter. This parameter is handed to the (lexicographically) first nonterminal in the right-hand side until it is resolved in a terminal production. Formally, for each right-hand side $r \in T_{N \cup N' \cup \Sigma}(X)$ such that $\mathrm{pos}_N(r) \neq \emptyset$ (i.e., it contains an original nonterminal), each $k \in \mathbb{N}$, and each $\delta \in \Delta$, let $r_{\delta,k}$ and $\overline{r}_\delta$ be such that

$$\begin{aligned} r_{\delta,k} &= r[\langle B, \delta\rangle(r_1, \ldots, r_n, x_{k+1})]_p \\ \overline{r}_\delta &= r[\langle B, \delta\rangle(r_1, \ldots, r_n, \delta)]_p \ , \end{aligned}$$

where p is the lexicographically smallest element of $\mathrm{pos}_N(r)$ and $r|_p = B(r_1, \ldots, r_n)$ with $B \in N$ and $r_1, \ldots, r_n \in T_{N \cup N' \cup \Sigma}(X)$. For each nonterminal A-production $\rho = \ell \to r$ in P let

$$\rho_\delta = \langle A, \delta\rangle(x_1, \ldots, x_{k+1}) \to r_{\delta,k} \ ,$$

where $k = \mathrm{rk}(A)$. This construction is illustrated in Figure 6. Roughly speaking, we select the lexicographically smallest occurrence of a nonterminal in the right-hand side and pass the lexical symbol δ in the extra parameter to it. The extra parameter is used in terminal productions, so let $\rho = \ell \to r$ in P

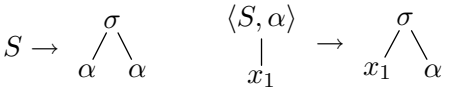

Figure 7: Original terminal production ρ from (1) [left] and the production $\overline{\rho}$ (see Theorem 17).

be a terminal A-production. Then we define

$$\overline{\rho} = \langle A, r(p)\rangle(x_1, \ldots, x_{k+1}) \to r[x_{k+1}]_p \ ,$$

where p is the lexicographically smallest element of $\mathrm{pos}_\Delta(r)$ and $k = \mathrm{rk}(A)$. This construction is illustrated in Figure 7. With these productions we obtain the CFTG $G' = (N \cup N', \Sigma, S, \overline{P})$, where $\overline{P} = P \cup P' \cup P''$ and

$$P' = \bigcup_{\substack{\rho=\ell\to r\in P\\ \ell\neq S, \mathrm{pos}_N(r)\neq\emptyset}} \{\rho_\delta \mid \delta \in \Delta\} \qquad P'' = \bigcup_{\substack{\rho=\ell\to r\in P\\ \ell\neq S, \mathrm{pos}_N(r)=\emptyset}} \{\overline{\rho}\} \ .$$

It is easy to prove that those new productions manage the desired transport of the extra parameter if it holds the value indicated in the nonterminal.

Finally, we replace each non-initial non-Δ-lexicalized production in G' by new productions that guess a lexical symbol and add it to the new parameter of the (lexicographically) first nonterminal of N in the right-hand side. Formally, we let

$$\overline{P}_{\text{nil}} = \{\ell \to r \in \overline{P} \mid \ell \neq S, \mathrm{pos}_\Delta(r) = \emptyset\}$$
$$P''' = \{\ell \to \overline{r}_\delta \mid \ell \to r \in \overline{P}_{\text{nil}}, \delta \in \Delta\} \ ,$$

of which P''' is added to the productions. Note that each production $\ell \to r \in \overline{P}_{\text{nil}}$ contains at least one occurrence of a nonterminal of N (because all monic productions of G are Δ-lexicalized). Now all non-initial non-Δ-lexicalized productions from $\overline{P}$ can be removed, so we obtain the CFTG G'', which is given by $(N \cup N', \Sigma, S, R)$ with $R = (\overline{P} \cup P''') \setminus \overline{P}_{\text{nil}}$. It can be verified that G'' is Δ-lexicalized and equivalent to G (using the provided argumentation). □

Instead of taking the lexicographically smallest element of $\mathrm{pos}_N(r)$ or $\mathrm{pos}_\Delta(r)$ in the previous proof, we can take any fixed element of that set. In the definition of P' we can change $\mathrm{pos}_N(r) \neq \emptyset$ to $|\mathrm{pos}_\Delta(r)| \leq 1$, and simultaneously in the definition of P'' change $\mathrm{pos}_N(r) = \emptyset$ to $|\mathrm{pos}_\Delta(r)| \geq 2$. With the latter changes the guessed lexical item is only transported until it is resolved in a production with at least two lexical items.

Example 18. For the last time, we consider the CFTG G''_{ex} of Example 8. We already illustrated the parts of the construction of Theorem 17 in Figures 6 and 7. The obtained $\{\alpha, \beta\}$-lexicalized CFTG has the following 25 productions for all $\delta, \delta' \in \{\alpha, \beta\}$:

$$\begin{aligned}
S' &\to S \\
S &\to A(\delta,\delta) \mid \sigma(\delta,\delta) \mid \sigma(\alpha,\beta) \\
S_\delta(x_1) &\to A_\delta(\delta',\delta',x_1) \mid \sigma(x_1,\delta) \\
S_\alpha(x_1) &\to \sigma(x_1,\beta) \\
A(x_1,x_2) &\to A_\delta(\sigma(x_1,S),\sigma(x_2,S),\delta) \qquad (5)\\
A_\delta(x_1,x_2,x_3) &\to A_\delta(\sigma(x_1,S_{\delta'}(\delta')),\sigma(x_2,S),x_3) \\
A(x_1,x_2) &\to \sigma(\sigma(x_1,S_\delta(\delta)),\sigma(x_2,S)) \\
A_\delta(x_1,x_2,x_3) &\to \sigma(\sigma(x_1,S_\delta(x_3)),\sigma(x_2,S_{\delta'}(\delta'))) \ ,
\end{aligned}$$

where $A_\delta = \langle A, \delta\rangle$ and $S_\delta = \langle S, \delta\rangle$.

If we change the lexicalization construction as indicated before this example, then all the productions $S_\delta(x_1) \to A_\delta(\delta', \delta', x_1)$ are replaced by the productions $S_\delta(x_1) \to A(x_1, \delta)$. Moreover, the productions (5) can be replaced by the productions $A(x_1, x_2) \to A(\sigma(x_1, S_\delta(\delta)), \sigma(x_2, S))$, and then the nonterminals A_δ and their productions can be removed, which leaves only 15 productions.

Conclusion

For $k \in \mathbb{N}$, let CFTG(k) be the set of those CFTG whose nonterminals have rank at most k. Since the normal form constructions preserve the nonterminal rank, the proof of Theorem 17 shows that CFTG(k) are strongly lexicalized by CFTG$(k{+}1)$. Kepser and Rogers (2011) show that non-strict TAG are strongly equivalent to CFTG(1). Hence, non-strict TAG are strongly lexicalized by CFTG(2).

It follows from Section 6 of Engelfriet et al. (1980) that the classes CFTG(k) with $k \in \mathbb{N}$ induce an infinite hierarchy of string languages, but it remains an open problem whether the rank increase in our lexicalization construction is necessary.

Gómez-Rodríguez et al. (2010) show that well-nested LCFRS of maximal fan-out k can be parsed in time $O(n^{2k+2})$, where n is the length of the input string $w \in \Delta^*$. From this result we conclude that CFTG(k) can be parsed in time $O(n^{2k+4})$, in the sense that we can produce a parse tree t that is generated by the CFTG with $\mathrm{yd}_\Delta(t) = w$. It is not clear yet whether lexicalized CFTG(k) can be parsed more efficiently in practice.

References

Franz Baader and Tobias Nipkow. 1998. *Term Rewriting and All That*. Cambridge University Press.

Yehoshua Bar-Hillel, Haim Gaifman, and Eli Shamir. 1960. On categorial and phrase-structure grammars. *Bulletin of the Research Council of Israel*, 9F(1):1–16.

Norbert Blum and Robert Koch. 1999. Greibach normal form transformation revisited. *Inform. and Comput.*, 150(1):112–118.

John Chen. 2001. *Towards Efficient Statistical Parsing using Lexicalized Grammatical Information*. Ph.D. thesis, University of Delaware, Newark, USA.

Noam Chomsky. 1963. Formal properties of grammar. In R. Duncan Luce, Robert R. Bush, and Eugene Galanter, editors, *Handbook of Mathematical Psychology*, volume 2, pages 323–418. John Wiley and Sons, Inc.

Joost Engelfriet, Grzegorz Rozenberg, and Giora Slutzki. 1980. Tree transducers, L systems, and two-way machines. *J. Comput. System Sci.*, 20(2):150–202.

Michael J. Fischer. 1968. Grammars with macro-like productions. In *Proc. 9th Ann. Symp. Switching and Automata Theory*, pages 131–142. IEEE Computer Society.

Akio Fujiyoshi. 2005. Epsilon-free grammars and lexicalized grammars that generate the class of the mildly context-sensitive languages. In *Proc. 7th Int. Workshop Tree Adjoining Grammar and Related Formalisms*, pages 16–23.

Akio Fujiyoshi and Takumi Kasai. 2000. Spinal-formed context-free tree grammars. *Theory Comput. Syst.*, 33(1):59–83.

Ferenc Gécseg and Magnus Steinby. 1984. *Tree Automata*. Akadémiai Kiadó, Budapest.

Ferenc Gécseg and Magnus Steinby. 1997. Tree languages. In Grzegorz Rozenberg and Arto Salomaa, editors, *Handbook of Formal Languages*, volume 3, chapter 1, pages 1–68. Springer.

Jonathan Goldstine, Hing Leung, and Detlef Wotschke. 1992. On the relation between ambiguity and nondeterminism in finite automata. *Inform. and Comput.*, 100(2):261–270.

Carlos Gómez-Rodríguez, Marco Kuhlmann, and Giorgio Satta. 2010. Efficient parsing of well-nested linear context-free rewriting systems. In *Proc. Ann. Conf. North American Chapter of the ACL*, pages 276–284. Association for Computational Linguistics.

Hendrik Jan Hoogeboom and Paulien ten Pas. 1997. Monadic second-order definable text languages. *Theory Comput. Syst.*, 30(4):335–354.

John E. Hopcroft, Rajeev Motwani, and Jeffrey D. Ullman. 2001. *Introduction to automata theory, languages, and computation*. Addison-Wesley series in computer science. Addison Wesley, 2nd edition.

Aravind K. Joshi, S. Rao Kosaraju, and H. Yamada. 1969. String adjunct grammars. In *Proc. 10th Ann. Symp. Switching and Automata Theory*, pages 245–262. IEEE Computer Society.

Aravind K. Joshi, Leon S. Levy, and Masako Takahashi. 1975. Tree adjunct grammars. *J. Comput. System Sci.*, 10(1):136–163.

Aravind K. Joshi and Yves Schabes. 1992. Tree-adjoining grammars and lexicalized grammars. In Maurice Nivat and Andreas Podelski, editors, *Tree Automata and Languages*. North-Holland.

Aravind K. Joshi and Yves Schabes. 1997. Tree-adjoining grammars. In Grzegorz Rozenberg and Arto Salomaa, editors, *Beyond Words*, volume 3 of *Handbook of Formal Languages*, pages 69–123. Springer.

Makoto Kanazawa. 2009. The convergence of well-nested mildly context-sensitive grammar formalisms. Invited talk at the 14th Int. Conf. Formal Grammar. slides available at: `research.nii.ac.jp/~kanazawa`.

Makoto Kanazawa and Ryo Yoshinaka. 2005. Lexicalization of second-order ACGs. Technical Report NII-2005-012E, National Institute of Informatics, Tokyo, Japan.

Stephan Kepser and James Rogers. 2011. The equivalence of tree adjoining grammars and monadic linear context-free tree grammars. *J. Log. Lang. Inf.*, 20(3):361–384.

Ines Klimann, Sylvain Lombardy, Jean Mairesse, and Christophe Prieur. 2004. Deciding unambiguity and sequentiality from a finitely ambiguous max-plus automaton. *Theoret. Comput. Sci.*, 327(3):349–373.

Marco Kuhlmann. 2010. *Dependency Structures and Lexicalized Grammars: An Algebraic Approach*, volume 6270 of LNAI. Springer.

Marco Kuhlmann and Mathias Mohl. 2006. Extended cross-serial dependencies in tree adjoining grammars. In *Proc. 8th Int. Workshop Tree Adjoining Grammars and Related Formalisms*, pages 121–126. ACL.

Marco Kuhlmann and Giorgio Satta. 2012. Tree-adjoining grammars are not closed under strong lexicalization. *Comput. Linguist.* available at: `dx.doi.org/10.1162/COLI_a_00090`.

Uwe Mönnich. 1997. Adjunction as substitution: An algebraic formulation of regular, context-free and tree adjoining languages. In *Proc. 3rd Int. Conf. Formal Grammar*, pages 169–178. Université de Provence, France. available at: `arxiv.org/abs/cmp-lg/9707012v1`.

Uwe Mönnich. 2010. Well-nested tree languages and attributed tree transducers. In *Proc. 10th Int. Conf. Tree Adjoining Grammars and Related Formalisms*. Yale University. available at: `www2.research.att.com/~srini/TAG+10/papers/uwe.pdf`.

Andreas Potthoff and Wolfgang Thomas. 1993. Regular tree languages without unary symbols are star-free. In *Proc. 9th Int. Symp. Fundamentals of Computation Theory*, volume 710 of LNCS, pages 396–405. Springer.

William C. Rounds. 1969. Context-free grammars on trees. In *Proc. 1st ACM Symp. Theory of Comput.*, pages 143–148. ACM.

William C. Rounds. 1970. Tree-oriented proofs of some theorems on context-free and indexed languages. In *Proc. 2nd ACM Symp. Theory of Comput.*, pages 109–116. ACM.

Yves Schabes. 1990. *Mathematical and Computational Aspects of Lexicalized Grammars*. Ph.D. thesis, University of Pennsylvania, Philadelphia, USA.

Yves Schabes, Anne Abeillé, and Aravind K. Joshi. 1988. Parsing strategies with 'lexicalized' grammars: Application to tree adjoining grammars. In *Proc. 12th Int. Conf. Computational Linguistics*, pages 578–583. John von Neumann Society for Computing Sciences, Budapest.

Yves Schabes and Richard C. Waters. 1995. Tree insertion grammar: A cubic-time parsable formalism that lexicalizes context-free grammars without changing the trees produced. *Comput. Linguist.*, 21(4):479–513.

Hiroyuki Seki, Takashi Matsumura, Mamoru Fujii, and Tadao Kasami. 1991. On multiple context-free grammars. *Theoret. Comput. Sci.*, 88(2):191–229.

Heiko Stamer. 2009. *Restarting Tree Automata: Formal Properties and Possible Variations*. Ph.D. thesis, University of Kassel, Germany.

Heiko Stamer and Friedrich Otto. 2007. Restarting tree automata and linear context-free tree languages. In *Proc. 2nd Int. Conf. Algebraic Informatics*, volume 4728 of LNCS, pages 275–289. Springer.

K. Vijay-Shanker, David J. Weir, and Aravind K. Joshi. 1987. Characterizing structural descriptions produced by various grammatical formalisms. In *Proc. 25th Ann. Meeting of the Association for Computational Linguistics*, pages 104–111. Association for Computational Linguistics.

XTAG Research Group. 2001. A lexicalized tree adjoining grammar for English. Technical Report IRCS-01-03, University of Pennsylvania, Philadelphia, USA.

Ryo Yoshinaka. 2006. *Extensions and Restrictions of Abstract Categorial Grammars*. Ph.D. thesis, University of Tokyo.