Extended cross-serial dependencies in Tree Adjoining Grammars

Marco Kuhlmann and **Mathias Möhl**
Programming Systems Lab
Saarland University
Saarbrücken, Germany
{kuhlmann|mmohl}@ps.uni-sb.de

Abstract

The ability to represent cross-serial dependencies is one of the central features of Tree Adjoining Grammar (TAG). The class of dependency structures representable by lexicalized TAG derivations can be captured by two graph-theoretic properties: a bound on the *gap degree* of the structures, and a constraint called *well-nestedness*. In this paper, we compare formalisms from two strands of extensions to TAG in the context of the question, how they behave with respect to these constraints. In particular, we show that multi-component TAG does not necessarily retain the well-nestedness constraint, while this constraint is inherent to Coupled Context-Free Grammar (Hotz and Pitsch, 1996).

1 Introduction

The ability to assign 'limited cross-serial dependencies' to the words in a sentence is a hallmark of mildly context-sensitive grammar formalisms (Joshi, 1985). In the case of TAG, an exact definition of this ability can be given in terms of two graph-theoretic properties of the dependency structures induced by TAG derivations: the *gap degree restriction* and the *well-nestedness* constraint (Bodirsky et al., 2005).

Gap degree and well-nestedness can be seen as the formal correspondents of what Joshi (1985) refers to as 'a limited amount of cross-serial dependencies' and 'the nesting properties as in the case of context-free grammars.' More specifically, the gap degree of a dependency structure counts the number of discontinuities in a dependency subtree, while well-nestedness constrains the positions of disjoint subtrees relative to one another. The dependency structures that correspond to the derivations in a lexicalized TAG are well-nested, and their gap degree is at most 1.

In the present paper, we compare formalisms from two strands of extensions to TAG in the context of the question, what classes of dependency structures they are able to induce.

We are particularly interested in formalisms that induce only well-nested dependency structures. This interest is motivated by two observations: First, well-nestedness is interesting as a generalization of projectivity (Marcus, 1967)—while more than 23% of the 73 088 dependency structures in the Prague Dependency Treebank of Czech (Hajič et al., 2001) are non-projective, only 0.11% are not well-nested (Kuhlmann and Nivre, 2006). Second, well-nestedness is interesting for processing. Specifically, parsers for well-nested grammar formalisms are not confronted with the 'crossing configurations' that make the universal recognition problem of Linear Context-Free Rewriting Systems NP-complete (Satta, 1992). In summary, it appears that well-nestedness can strike a successful balance between empirical coverage and computational tractability. If this is true, then a formalism that has the well-nestedness constraint hardwired is preferable over one that has not.

The results of this paper can be summarized as follows: Derivations in lexicalized multi-component TAGs (Weir, 1988; Kallmeyer, 2005), in which a single adjunction adds a set of elementary trees, either induce exactly the same dependency structures as TAG, or induce all structures of bounded gap degree, even non-well-nested ones. This depends on the decision whether one takes 'lexicalized' to mean 'one lexical anchor per tree', or 'one lexical anchor per tree set'. In contrast, multi-foot extensions of TAG (Abe, 1988; Hotz and Pitsch, 1996), where a single elementary tree may have more than one foot node, only induce well-nested dependency structures of bounded gap degree. Thus, from the dependency point of view, they constitute the structurally more conservative extension of TAG.

2 Dependency structures for TAG

We start with a presentation of the dependency view on TAG that constitutes the basis for our work, and introduce the relevant terminology. The main objective of this section is to provide intuitions; for the formal details, see Bodirsky et al. (2005).

2.1 The dependency view on TAG

Let $s = w_1 \cdots w_n$ be a sentence (a sequence of tokens). By a *dependency structure* for s, we mean a tuple $(W, \rightarrow, \prec)$, where $W = \{w_1, \ldots, w_n\}$, and

$$\rightarrow \;=\; \{\, (w_i, w_j) \in W \times W \mid w_j \text{ depends on } w_i \,\}$$
$$\prec \;=\; \{\, (w_i, w_j) \in W \times W \mid i < j \,\}$$

To interpret a grammar formalism as a specification for a set of dependency structures, we need to assign meaning to the relation 'depends' in terms of this formalism. For TAG, this can be done based on the Fundamental Hypothesis that 'every syntactic dependency is expressed locally within a single elementary tree' (Frank, 2002). More specifically, a derivation in a (strongly) lexicalized TAG can be viewed as a dependency structure as follows: The set W contains the (occurences of) lexical anchors involved in the derivation. For two anchors $w_i, w_j \in W$, $w_i \rightarrow w_j$ if the elementary tree anchored at w_j was substituted or adjoined into the tree anchored at w_i. We then have $w_i \prec w_j$ if w_i precedes w_j in the yield of the derived tree corresponding to the derivation. Notice that the relation $\rightarrow$ in such a dependency structure is almost exactly the derivation tree of the underlying TAG derivation; the only difference is that elementary trees have been replaced by their lexical anchors.

Figure 1 shows a TAG grammar together with a dependency structure induced by a derivation of this grammar. Tokens in the derived string are represented by labelled nodes; the solid arcs between the nodes represent the dependencies.

2.2 Gap degree and well-nestedness

An interesting feature of the dependency structure shown in Figure 1 is that it violates a standard constraint on dependency structures known as *projectivity* (Marcus, 1967). We introduce some terminology for non-projective dependency structures:

A set $T \subseteq W$ is *convex*, if for no two tokens $w_1, w_2 \in T$, there exists a token w from $W - T$ such that $w_1 \prec w \prec w_2$. The *cover* of T, $\mathcal{C}(T)$, is the smallest convex set that contains T. For $w \in W$, we write $\downarrow w$ for the set of tokens in the subtree rooted at w (including w itself). A *gap* in $\downarrow w$ is a largest convex set in $\mathcal{C}(\downarrow w) - \downarrow w$. The *gap degree* of w, $gd(w)$, is the number of gaps in $\downarrow w$. The gaps in $\downarrow w$ partition $\downarrow w$ into $gd(w) - 1$ largest convex blocks; we write $\downarrow_i w$ to refer to the i-th of these blocks, counted from left to right (with respect to $\prec$). The gap degree of a dependency structure is the maximum over the gap degrees of its subtrees; we write $\mathcal{D}_g$ for the set of all dependency structures with a gap degree of at most g.

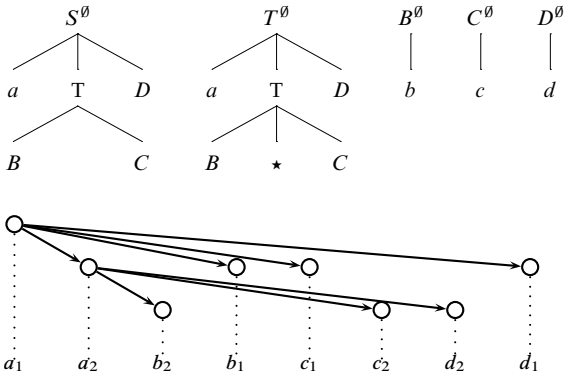


Figure 1: TAG grammar for $a^n b^n c^n d^n$, and a dependency structure induced by this grammar

The gap degree provides a quantitative measure for the non-projectivity of dependency structures. *Well-nestedness* is a qualitative property: it constrains the relative positions of disjoint subtrees. Let $w_1, w_2 \in W$ such that $\downarrow w_1$ and $\downarrow w_2$ are disjoint. Four tokens $w_1^1, w_1^2 \in \downarrow w_1$, $w_2^1, w_2^2 \in \downarrow w_2$ *interleave*, if $w_1^1 \prec w_2^1 \prec w_1^2 \prec w_2^2$. A dependency structure is *well-nested*, if it does not contain interleaving tokens. We write $\mathcal{D}_{wn}$ for the set of all well-nested dependency structures.

For illustration, consider again the dependency structure shown in Figure 1. It has gap degree 1: a_2 is the only token w for which $\downarrow w$ is not convex; the set $\{b_1, c_1\}$ forms a gap in $\downarrow a_2$. The structure is also well-nested. In contrast, the structure shown in the right half of Figure 2 is not well-nested; the tokens b, c, d, e interleave. Bodirsky et al. (2005) show that TAG induces precisely the set $\mathcal{D}_{wn} \cap \mathcal{D}_1$.

3 Multi-component extensions

Multi-component TAG (MCTAG) extends TAG with the ability to adjoin a whole set of elementary trees (*components*) simultaneously. To answer the question, whether this extension also leads to an extended class of dependency structures, we first need to decide how we want to transfer the Fundamental Hypothesis (Frank, 2002) to MCTAGs.

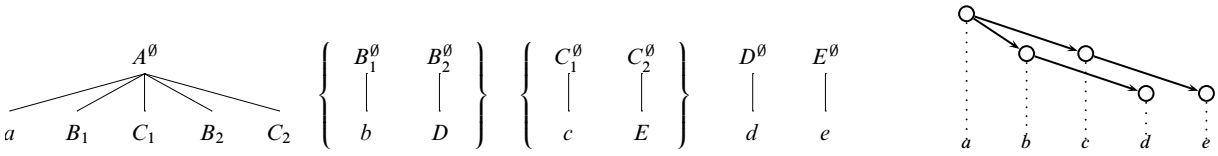


Figure 2: An MCTAG and a not well-nested dependency structure derived by it.

3.1 One anchor per component

If we commit to the view that each component of a tree set introduces a separate lexical anchor and its syntactic dependencies, the dependency structures induced by MCTAG are exactly the structures induced by TAG. In particular, each node in the derivation tree, and therefore each token in the dependency tree, corresponds to a single elementary tree. As Kallmeyer (2005) puts it, one can then consider an MCTAG as a TAG G 'where certain derivation trees in G are disallowed since they do not satisfy certain constraints.' The ability of MCTAG to perform multiple adjunctions simultaneously allows one to induce more complex *sets* of dependency structures—each individual structure is limited as in the case of standard TAG.

3.2 One anchor per tree set

If, on the other hand, we take a complete tree set as the level on which syntactic dependencies are specified, MCTAGs can induce a larger class of dependency structures. Under this perspective, tokens in the dependency structure correspond not to individual components, but to tree sets (Weir, 1988). For each token w, $\downarrow w$ then contains the lexical anchors of all the subderivations starting in the tree set corresponding to w. As there can be a gap between each two of these subderivations, the gap degree of the induced dependency structures is bounded only by the maximal number of components per tree set. At the same time, even non-well-nested structures can be induced; an example is shown in Figure 2. Here, $\downarrow b$ is distributed over the components rooted at B_1 and B_2, and $\downarrow c$ is distributed over C_1 and C_2. The elementary tree rooted at A arranges the substitution sites such that b, c, d, e interleave. Note that the MCTAG used in this example is heavily restricted: it is tree-local and does not even use adjunction. This restricted form suffices to induce non-well-nested dependency structures.

4 Multi-foot extensions

A second way to extend TAG, orthogonal to the multi-component approach, is to allow a single elementary tree to have more than one foot node. For this kind of extension, the Fundamental Hypothesis does not need to be re-interpreted. Probably the most prominent multi-foot extension of TAG is Ranked Node Rewriting Grammar (RNRG) (Abe, 1988); however, the properties that we are interested in here can be easier investigated in a notational variant of RNRG, Coupled Context-Free Grammar (Hotz and Pitsch, 1996).

Terminology Multi-foot formalisms require a means to specify which foot node gets what material in an adjunction. To do so, they use ranked symbols. A *ranked alphabet* is a pair $\Pi = (\Sigma, \rho)$, where Σ is an alphabet, and $\rho \in \Sigma \to \mathbb{N}$ is a total function that assigns every symbol $\sigma \in \Sigma$ a (positive) *rank*. Define $\Pi[r] := \{\, \sigma \in \Sigma \mid \rho(\sigma) = r \,\}$. The *components* of σ, $comp(\sigma)$, are the elements of the set $\{\, (\sigma, i) \mid 1 \leq i \leq \rho(\sigma) \,\}$. We write σ_i instead of (σ, i). Let $comp(\Pi) := \bigcup_{\sigma \in \Pi} comp(\sigma)$.

4.1 Coupled Context-Free Grammar

Coupled Context-Free Grammar (CCFG) is a generalization of context-free grammar in which non-terminals come from a ranked alphabet, and components of a non-terminal can only be substituted simultaneously. The 'TAG-ness' of CCFG is reflected in the requirement, that the RHS of productions must be words from a bracket-like language, and thus have the same hierarchical structure as elementary trees in a TAG. As an example, the second elementary tree from Figure 1 can be linearized as

$$\langle T_1 a T_1 B_1, C_1 T_2 D_1 T_2 \rangle ,$$

where each pair (T_1, T_2) of matching components corresponds to an inner node in the tree, and the boundary between the first and the second part of the tuple marks the position of the foot node. The required structure of the RHS can be formalized as follows:

Definition 1 Let Π be a ranked alphabet, and let Σ be an unranked alphabet. The *extended semi-Dyck set* over Π and Σ, $\mathrm{ESD}(\Pi, \Sigma)$, is the smallest set that satisfies the following properties:

(a) $\Sigma^* \subseteq \text{ESD}(\Pi, \Sigma)$; (b) $\Pi[1] \subseteq \text{ESD}(\Pi, \Sigma)$; (c) if $s_1, \ldots, s_k \in \text{ESD}(\Pi, \Sigma)$ and $\pi \in \Pi[k+1]$, then $\pi_1 s_1 \pi_2 \cdots \pi_k s_k \pi_{k+1} \in \text{ESD}(\Pi, \Sigma)$; (d) if $s_1, s_2 \in \text{ESD}(\Pi, \Sigma)$, then $s_1 s_2 \in \text{ESD}(\Pi, \Sigma)$.

Definition 2 Let N be a ranked alphabet of non-terminals, and let T be an (unranked) alphabet of terminals. A *ranked rewriting system* over $\text{ESD}(N, T)$ is a finite, non-empty set of productions of the form $X \to \langle \alpha_1, \ldots, \alpha_r \rangle$, where $X \in N[r]$, and $\alpha := \alpha_1 \cdots \alpha_r \in \text{ESD}(N, T)$.

We write $\rho(p)$ to refer to the rank of the non-terminal on the LHS of a production p.

RNRG and CCFG are notational variants because each RNRG elementary tree with $r-1$ foot nodes can be linearized into the RHS of a production $X \to \langle \alpha_1, \ldots, \alpha_r \rangle$ in a ranked rewriting system, as indicated by the example above.

Definition 3 A *coupled context-free grammar* is a tuple $G = (N, T, P, S)$ where: N is a ranked alphabet of *non-terminal symbols*; T is an unranked alphabet of *terminal symbols*; P is a ranked rewriting system over $\text{ESD}(N, T)$; $S \in N[1]$ is a *start symbol*.

We say that a CCFG G is an r-CCFG, if the maximal rank among all non-terminals in G is r.

Definition 4 Put $V := \mathit{comp}(N) \cup T$, and let

$$\begin{aligned} \phi \in V^* &= u_1 X_1 u_2 \cdots u_r X_r u_{r+1} \\ \psi \in V^* &= u_1 \alpha_1 u_2 \cdots u_r \alpha_r u_{r+1} \end{aligned}$$

such that $u_2, \ldots, u_r \in \text{ESD}(N, T)$, and $X \in N[r]$. We say that ψ can be derived from ϕ in one step, and write $\phi \Rightarrow_G \psi$, if G contains a production $X \to \langle \alpha_1, \ldots, \alpha_r \rangle$. The *string language* of G is the set $L(G) := \{ s \in T^* \mid S \Rightarrow_G^* s \}$.

Based on this definition, the notions of *derivation tree* and *derived tree* are defined in the usual way. In particular, the nodes of the derivation tree are labelled with productions, while the nodes of the corresponding derived tree are labelled with components from $\mathit{comp}(\Pi)$ (inner nodes) and terminal symbols (leaves). We write $(T^\sharp, T^\flat)$ to refer to a derivation in CCFG: $T^\sharp$ stands for the derivation tree, $T^\flat$ for the corresponding derived tree.

4.2 The dependency view on CCFG

A CCFG G is *strongly lexicalized*, if each production p contains exactly one terminal symbol, written as $\mathit{anchor}(p)$. Just as in the case of TAG, a strongly lexicalized CCFG G can be interpreted as a dependency grammar: Let $(T^\sharp, T^\flat)$ be a derivation in G. Since G is strongly lexicalized, there is a one-to-one mapping between the nodes of the derivation tree $T^\sharp$ (labelled with productions) and the leaves of the derived tree $T^\flat$ (labelled with terminals); we refer to this mapping by the name f_L.

Definition 5 A dependency structure D is *induced* by a derivation $(T^\sharp, T^\flat)$, written $(T^\sharp, T^\flat) \vdash D$, if (a) $\mathit{anchor}(p_1) \to \mathit{anchor}(p_2)$ in D if and only if $p_1 \to p_2$ in $T^\sharp$; (b) $\mathit{anchor}(p_1) \prec \mathit{anchor}(p_2)$ in D if and only if $f_L(p_1) \prec f_L(p_2)$ in $T^\flat$.

We write $\mathcal{D}(G)$ for the set of all dependency structures induced by derivations in G. Figure 3 shows a sample CCFG G, a derivation in G, and the dependency structure induced by this derivation.

4.3 Projections

To reason about the structural properties of the dependency languages induced by CCFGs, we need some additional definitions. In the following, we use the notation $(u : \sigma)$ to refer to a node u with label σ in some given labelled tree.

Let $D \in \mathcal{D}(G)$ be a dependency structure such that $(T^\sharp, T^\flat) \vdash D$, and let $(u : p) \in T^\sharp$ be a node. Somewhere in the course of the derivation represented by $T^\sharp$, the $\rho(p)$ components of the non-terminal on the LHS of the production p are simultaneously rewritten. Let $f_I(u)$ be the $\rho(p)$-tuple of nodes in $T^\flat$ that correspond to these components. Note that, while f_L maps nodes in the derivation tree $T^\sharp$ to leaves in the derived tree $T^\flat$, f_I takes nodes in $T^\sharp$ to *tuples* of inner nodes in $T^\flat$. Define

$$\begin{aligned} \mathit{down}(u) &= \{ v \mid u \to^* v \text{ in } T^\sharp \}, \\ \mathit{proj}(u, i) &= \{ v \mid f_I(u)_i \to^* f_L(v) \text{ in } T^\flat \}. \end{aligned}$$

The set $\mathit{down}(u)$ contains the lexical anchors in the sub-derivation starting at u. The set $\mathit{proj}(u, i)$ identifies that part of this sub-derivation that is derived from the i-th component of the non-terminal at the LHS of the production corresponding to u. For the derivation shown in Figure 3, we have

$$f_I(p_2) = \langle B_1, B_2, B_3 \rangle, \quad \mathit{proj}(p_2, 1) = \{p_2\}.$$

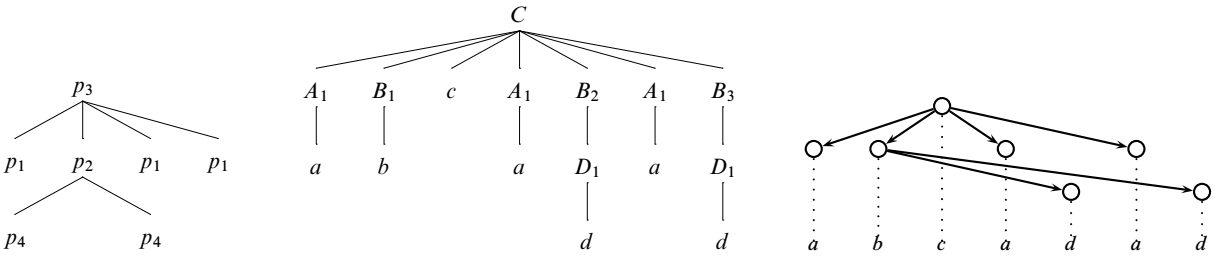
Lemma 6 For all nodes $u \in T^\sharp$,

$$\mathit{down}(u) = \biguplus_{1 \le i \le \rho(p)} \mathit{proj}(u, i).$$

4.4 Results

In this section, we prove the main technical results of this paper: that all dependency structures

Grammar G^*: $p_1\colon A \to \langle a \rangle,\ p_2\colon B \to \langle b, D_1, D_1 \rangle,\ p_3\colon C \to \langle A_1 B_1 c A_1 B_2 A_1 B_3 \rangle,\ p_4\colon D \to \langle d \rangle$



(a) Derivation tree (b) Derived tree (c) Induced dependency structure

Figure 3: A CCFG derivation and the dependency structure induced by it

induced by an r-CCFG have a gap degree that is bounded by r; that they are all well-nested; and that each well-nested structure with a gap degree bounded by r can be induced by an r-CCFG. In the following, let G be an r-CCFG, and write $\mathcal{G}_r$ for the set of all r-CCFGs.

Lemma 7 $\mathcal{D}(G) \subseteq \mathcal{D}_{r-1}$

Proof Let $(T^\sharp, T^\flat) \vdash D$, and let $(u : p) \in T^\sharp$. By definition of *proj*, for each $1 \leq i \leq \rho(p)$, the set $proj(u, i)$ forms a contiguous region of the sentence derived by $T^\sharp$. Using Lemma 6, we then see that $down(u)$ is distributed over at most $\rho(u)$ contiguous regions of that sentence. This means that the dependency subtree rooted at $anchor(p)$ has at most $\rho(p) - 1$ gaps.

Lemma 8 $\mathcal{D}(G) \subseteq \mathcal{D}_{wn}$

Proof Choose a $D \in \mathcal{D}(G)$, and assume that D is not well-nested. Then there is a governor $u \in D$ with two distinct dependents v, w such that $\downarrow v$ contains tokens v_1, v_2, and $\downarrow w$ contains tokens w_1, w_2 such that $v_1 \prec w_1 \prec v_2 \prec w_2$. For the derivation $(T^\sharp, T^\flat)$ that induces D, this means that there is a node $(u : p)$ with children $(v : p_v)$ and $(w : p_w)$ in $T^\sharp$ such that

$$\exists (v_1, v_2 \in down(v))\colon \exists (w_1, w_2 \in down(w))\colon$$
$$f_L(v_1) \prec f_L(w_1) \prec f_L(v_2) \prec f_L(w_2) \text{ in } T^\flat\,.$$

Since $down(v)$ and $down(w)$ are disjoint; v_1 and v_2 must come from distinct convex blocks in $down(v)$, and w_1 and w_2 must come from distinct convex blocks in $down(w)$. Therefore,

$$v_1 \in proj(v, i_1),\ v_2 \in proj(v, i_2),\ i_1 < i_2 \quad \text{and}$$
$$w_1 \in proj(w, j_1),\ w_2 \in proj(w, j_2),\ j_1 < j_2\,.$$

By definition, $proj(x, k)$ $(x \in \{v, w\})$ is the projection of a node $f_I(x)_k$ in $T^\flat$; the label of this node is $\text{LHS}(p_x)_k$. Assume now that the non-terminal on the LHS of p_v is V, and that the non-terminal on the LHS of p_w is W. Given that p_v and p_w are used to rewrite p, $\text{RHS}(p)$ contains the substring $V_{i_1} \cdots W_{j_1} \cdots V_{i_2} \cdots W_{j_2}$. This contradicts the fact that $\text{RHS}(p) \in \text{ESD}(N, T)$.

Lemma 9 $\mathcal{D}_{wn} \cap \mathcal{D}_{r-1} \subseteq \bigcup_{G \in \mathcal{G}_r} \mathcal{D}(G)$

Proof Let $D = (W, \to, \prec)$ be a dependency structure from $\mathcal{D}_{wn} \cap \mathcal{D}_{r-1}$. We construct an r-CCFG $G = (N, T, P, S)$ that induces D. For the ranked alphabet N of non-terminals, put

$$N = \{\, N^w \mid w \in W \,\},\ \rho(N^w) = gd(w) + 1\,.$$

The set S of start symbols is $\{N^\top\}$, where $\top$ is the root of D. For the terminal alphabet, put $T = W$. The set P consists of $|W|$ productions of the form $N^w \to \vec{\alpha}$, where $w \in W$, and $\vec{\alpha}$ is a tuple with arity $gd(w) + 1$ that contains the terminal w and non-terminal components for all children of w as follows. Consider the following family of sets:

$$\mathcal{C}_w = \{\{w\}\} \cup \{\, \downarrow_i v \mid w \to v, 1 \leq i \leq gd(v)+1 \,\}\,.$$

All sets in $\mathcal{C}_w$ are disjoint, and their union equals the set $\downarrow w$. We define a function $[\cdot]$ that interprets the elements of $\mathcal{C}_w$ as elements from $N \cup T$ as follows: $[\{w\}] := w$, and $[\downarrow_i v] := N_i^v$. Now the RHS of a rule $N^w \to \vec{\alpha}$ is fully specified by the following equivalences, where $C \in \mathcal{C}_w$:

$$[C] \text{ occurs in } \alpha_i \text{ iff } C \subseteq \downarrow_i w$$
$$[C_1] \text{ precedes } [C_2] \text{ in } \vec{\alpha} \text{ iff } C_1 \times C_2 \subseteq \prec$$

Applied to the dependency structure of Figure 3c, this constructs the given grammar G^*. Note that, due to the well-nestedness of D, the RHS of each rule forms a valid extended semi-Dyck word.

5 Summary

Starting from the fact that TAG is able to derive well-nested dependency structures with a gap degree of at most 1, we have investigated how multi-component and multi-foot extensions of TAG alter this expressivity. Our results are as follows:

- For multi-component TAG, the notion of 'induced dependency structures' depends on the assumed notion of lexicalization. Therefore, either the same structures as in TAG, or arbitrary gap-bounded dependency structures are derivable. In the former case, MCTAG has the same structural limits as standard TAG; in the latter case, even non-well-nested dependency structures are induced.

- The multi-foot extension CCFG (and its equivalent RNRG) is restricted to well-nested dependency structures, but in contrast to TAG, it can induce structures with any bounded gap degree. The rank of a grammar is an upper bound on the gap degree of the dependency structures it induces.

Since the extensions inherent to MCTAG and CCFG are orthogonal, it is possible to combine them: Multi-Component Multi-Foot TAG (MMTAG) as described by Chiang (2001) allows to simultaneously adjoin sets of trees, where each tree may have multiple foot nodes. The structural limitations of the dependency structures inducible by MCTAG and CCFG generalize to MMTAG as one would expect. As in the case of MCTAG, there are two different understandings of how a dependency structure is induced by an MMTAG. Under the 'one anchor per component' perspective, MMTAG, just like CCFG, derives well-nested structures of bounded gap-degree. Under the 'one anchor per tree set' perspective, just like MCTAG, it also derives non-well-nested gap-bounded structures.

Acknowledgements We thank Jan Schwinghammer, Guido Tack, and Stefan Thater for fruitful discussions during the preparation of this paper, and three anonymous reviewers for their detailed comments on an earlier version. The work of Marco Kuhlmann is funded by the Collaborative Research Centre 'Resource-Adaptive Cognitive Processes' of the Deutsche Forschungsgemeinschaft.

References

Naoki Abe. 1988. Feasible learnability of formal grammars and the theory of natural language acquisition. In *12th International Conference on Computational Linguistics*, pages 1–6, Budapest, Hungary.

Manuel Bodirsky, Marco Kuhlmann, and Mathias Möhl. 2005. Well-nested drawings as models of syntactic structure. In *Tenth Conference on Formal Grammar and Ninth Meeting on Mathematics of Language (FG-MoL)*, Edinburgh, UK.

David Chiang. 2001. Constraints on strong generative power. In *39th Annual Meeting and Tenth Conference of the European Chapter of the Association for Computational Linguistics*, pages 124–131, Toulouse, France.

Robert Frank. 2002. *Phrase Structure Composition and Syntactic Dependencies*. MIT Press.

Jan Hajič, Barbora Vidova Hladka, Jarmila Panevová, Eva Hajičová, Petr Sgall, and Petr Pajas. 2001. Prague Dependency Treebank 1.0. LDC, 2001T10.

Günther Hotz and Gisela Pitsch. 1996. On parsing coupled-context-free languages. *Theoretical Computer Science*, 161:205–233.

Aravind K. Joshi. 1985. Tree adjoining grammars: How much context-sensitivity is required to provide reasonable structural descriptions? In David R. Dowty, Lauri Karttunen, and Arnold M. Zwicky, editors, *Natural Language Parsing*, pages 206–250. Cambridge University Press, Cambridge, UK.

Laura Kallmeyer. 2005. A descriptive characterization of multicomponent tree adjoining grammars. In *Traitement Automatique des Langues Naturelles (TALN)*, volume 1, pages 457–462, Dourdan, France.

Marco Kuhlmann and Joakim Nivre. 2006. Mildly non-projective dependency structures. In *22nd International Conference on Computational Linguistics and 43rd Annual Meeting of the Association for Computational Linguistics (COLING-ACL), Companion Volume*, Sydney, Australia.

Solomon Marcus. 1967. *Algebraic Linguistics: Analytical Models*, volume 29 of *Mathematics in Science and Engineering*. Academic Press, New York.

Giorgio Satta. 1992. Recognition of linear context-free rewriting systems. In *30th Meeting of the Association for Computational Linguistics (ACL)*, pages 89–95, Newark, Delaware, USA.

David J. Weir. 1988. *Characterizing Mildly Context-Sensitive Grammar Formalisms*. Ph.D. thesis, University of Pennsylvania, Philadelphia, USA.