Tree-Local Multicomponent Tree-Adjoining Grammars with Shared Nodes

Laura Kallmeyer*
TALaNa/Lattice, Université Paris 7

This article addresses the problem that the expressive power of tree-adjoining grammars (TAGs) is too limited to deal with certain syntactic phenomena, in particular, with scrambling in free-word-order languages. The TAG variants proposed so far in order to account for scrambling are not entirely satisfying. Therefore, the article introduces an alternative extension of TAG that is based on the notion of node sharing, so-called (restricted) tree-local multicomponent TAG with shared nodes (RSN-MCTAG). The analysis of some German scrambling data is sketched in order to show that this TAG extension can deal with scrambling. Then it is shown that for RSN-MCTAGs of a specific type, equivalent simple range concatenation grammars can be constructed. As a consequence, these RSN-MCTAGs are mildly context-sensitive and in particular polynomially parsable. These specific RSN-MCTAGs probably can deal not with all scrambling phenomena, but with an arbitrarily large subset.

1. Introduction: LTAG and Scrambling

1.1 Lexicalized Tree-Adjoining Grammars

Tree-adjoining grammar (TAG) is a tree-rewriting formalism originally defined by Joshi, Levy, and Takahashi (1975). A TAG (see Joshi and Schabes 1997 for an introduction) consists of a finite set of trees (elementary trees). The nodes of these trees are labeled with nonterminals and terminals (terminals label only leaf nodes). Starting from the elementary trees, larger trees are derived using composition operations of substitution (replacing a leaf with a new tree) and adjunction (replacing an internal node with a new tree). In the case of an adjunction, the tree being adjoined has exactly one leaf node that is marked as the foot node (marked with an asterisk). Such a tree is called an **auxiliary tree**. When such a tree is adjoined to a node μ, in the resulting tree, the subtree with root node μ from the old tree is put below the foot node of the new auxiliary tree. Elementary trees that are not auxiliary trees are called **initial trees**. Each derivation starts with an initial tree. In the final derived tree, all leaves must have terminal labels.[1]

* UFR de Linguistique, Case 7003, 2 Place Jussiec, 75005 Paris. E-mail: Laura.Kallmeyer@linguist.jussieu.fr.

1 Additionally, TAG allows for each internal node to specify the set of auxiliary trees that can be adjoined using so-called adjunction constraints and, furthermore, to specify whether adjunction at that node is obligatory. This is an important feature of TAG, since it influences the generative capacity of the formalism: $\{a^n b^n c^n d^n \mid n \geq 0\}$, for example, is a language that can be generated by a TAG with adjunction constraints but not by a TAG without adjunction constraints (Joshi 1985). For this article, however, adjunction constraints do not play any important role.

Submission received: 1st September 2003; Revised submission received: 4th May 2004; Accepted for publication: 17th June 2004

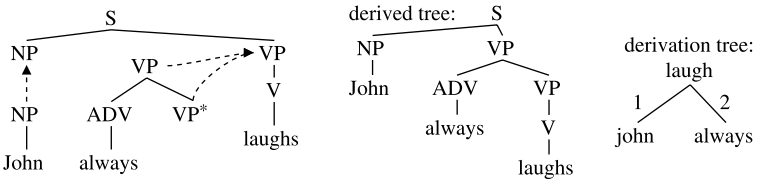


Figure 1
TAG derivation for *John always laughs*.

Figure 1 shows a sample TAG derivation. Here, the three elementary trees for *laughs*, *John,* and *always* are combined: Starting from the elementary tree for *laughs*, the tree for *John* is substituted for the noun phrase (NP) leaf and the tree for *always* is adjoined at the verb phrase (VP) node.

TAG derivations are represented by derivation trees that record the history of how the elementary trees are put together. A derivation tree is the result of carrying out substitutions and adjunctions. Each edge in the derivation tree stands for an adjunction or a substitution. The edges are labeled with Gorn addresses of the nodes where the substitutions and adjunctions have taken place: The root has the address ϵ, and the jth child of the node with address p has address pj. In Figure 1, for example, the derivation tree indicates that the elementary tree for *John* is substituted for the node at address 1 and *always* is adjoined at node address 2.

What we have sketched so far are the mathematical aspects of the TAG formalism. For natural languages, TAGs with specific properties are used. These properties are not part of the formalism itself, but they are additional linguistic principles that are respected when a TAG is constructed for a natural language. First, a TAG for natural languages is **lexicalized** (Schabes 1990), which means that each elementary tree has a lexical anchor (usually unique, but in some cases, there is more than one anchor). Second, the elementary trees of a lexicalized TAG (LTAG) represent extended projections of lexical items (the anchors) and encapsulate all syntactic arguments of the lexical anchor; that is, they contain slots (nonterminal leaves) for all arguments. Furthermore, elementary trees are minimal in the sense that only the arguments of the anchor are encapsulated; all recursion is factored away. This amounts to the **condition on elementary tree minimality** (CETM) from Frank (1992) (see also Frank [2002] for further discussions of the linguistic principles underlying TAG).[2] The tree for *laughs* in Figure 1, for example, contains only a nonterminal leaf for the subject NP (a substitution node), and there is no slot for a VP adjunct. The adverb *always* is added by adjunction at an internal node.

Because of these principles, in linguistic applications, combining two elementary trees by substitution or adjunction corresponds to the application of a predicate to an argument. The derivation tree then reflects the predicate-argument structure of the sentence. This is why most approaches to semantics in TAG use the derivation tree as an interface between syntax and semantics (see, e.g., Candito and Kahane 1998; Joshi and Vijay-Shanker 1999; Kallmeyer and Joshi 2003). In this article, we are not particularly concerned with semantics, but one of the goals of the article is to obtain analyses with derivation trees representing the correct predicate-argument dependencies.

2 This minimality is actually the reason that the substitution operation is needed; formally TAGs without substitution and TAGs as introduced above have the same weak and strong generative capacity.

An extension of TAG that has been shown to be useful for several linguistic applications is multicomponent TAG (MCTAG) (Joshi 1987; Weir 1988). Instead of single elementary trees, an MCTAG has sets of elementary trees. In each derivation step, one of these sets is chosen, and all trees from the set are added simultaneously. Depending on the nodes to which the different trees from the set attach, different kinds of MCTAGs are distinguished: If all nodes are required to be part of the same elementary tree, the MCTAG is called **tree-local**; if all nodes are required to be part of the same tree set, the grammar is **set-local**; and otherwise the grammar is **nonlocal**.

1.2 Scrambling in TAG

Roughly, **scrambling** can be described as the permutation of elements (arguments and adjuncts) of a sentence (we use the term *scrambling* in a purely descriptive sense without implying any theory involving actual movement). A special case of scrambling is so-called **long-distance scrambling**, in which arguments or adjuncts of an embedded infinitive are "moved" out of the embedded VP. This occurs, for instance, in languages such as German, Hindi, Japanese, and Korean. As an example of long-distance scrambling in German, consider example (1):

(1) …*dass* [*es*]$_1$ *der Mechaniker* [t_1 *zu reparieren*] *verspricht*
…that it the mechanic to repair promises
'…that the mechanic promises to repair it'

In example (1), the accusative NP *es* is an argument of the embedded infinitive *zu reparieren*, but it precedes *der Mechaniker*, the subject of the main verb *verspricht*, and it is not part of the embedded VP.

It has been argued (see Rambow 1994a) that in German, there is no bound on the number of scrambled elements and no bound on the depth of scrambling (i.e., in terms of movement, the number of VP borders crossed by the moved element).

TAGs are not powerful enough to describe scrambling in German in an adequate way (Becker, Joshi, and Rambow 1991). By this we mean that a TAG analysis of scrambling respecting the CETM and therefore giving the correct predicate-argument structure (i.e., an analysis with each argument attaching to the verb it depends on) is not possible.

Let us consider the TAG analysis of example (1) in order to see why scrambling poses a problem for TAG. If we leave aside the complementizer *dass*, standard TAG elementary trees for *verspricht* and *reparieren* in the style of the XTAG grammar (XTAG Research Group 1998) might look as shown in Figure 2. In the derivation, the

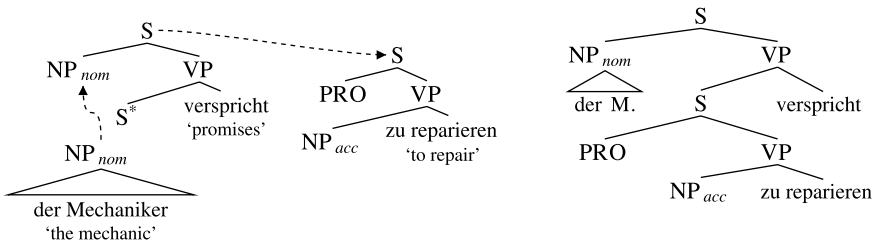


Figure 2
Standard TAG combination of *der Mechaniker*, *zu reparieren*, and *verspricht* in example (1).

verspricht-tree adjoins to the root node of the *reparieren*-tree, and the nominative NP *der Mechaniker* is substituted for the subject node in the *verspricht*-tree. This leads to the tree on the right in Figure 2.

When *es* is added, there is a problem: It should be added to *reparieren*, since it is one of its arguments. But at the same time, it should precede *der Mechaniker*; that is, it must be adjoined either to the root or to the NP_{nom} node. The root node belongs to *verspricht*, and the NP_{nom} node belongs to *der Mechaniker*. Consequently, an adjunction to one of them would not give the desired predicate-argument structure. If one wanted to analyze only example (1), one could add a tree to the grammar for *reparieren* with a scrambled NP that allows adjunction of *verspricht* between the NP and the verb. But as soon as there are several scrambled elements that are arguments of different verbs, this no longer works.

This example has given an idea of why scrambling is problematic for TAG. However, adopting specific elementary trees, it is possible to deal with a part of the difficult scrambling data: It has been shown (see Joshi, Becker, and Rambow 2000) that TAG can describe scrambling up to depth two (two crossed VP borders). But this is not sufficient. Even though examples of scrambling of depth greater than two are rare, they can occur. An example is example (2), taken from Kulick (2000):

(2) …*dass* [*den Kühlschrank*]$_1$ *niemand* [[[t_1 *zu reparieren*] *zu versuchen*]
…that the refrigerator nobody to repair to try

zu versprechen] *bereit ist*
to promise willing is
'…that nobody is willing to promise to try to repair the refrigerator'

Consequently, TAG is not powerful enough to account for scrambling.[3]

Becker, Rambow, and Niv (1992) argue that even linear context-free rewriting systems (LCFRSs) (Weir 1988) are not powerful enough to describe scrambling. (LCFRSs are weakly equivalent to set-local MCTAGs and therefore more powerful than TAGs.) Although we think that the language Becker, Rambow, and Niv define as a kind of test language for scrambling is not exactly what one needs (see section 2.3), we still suspect that they are right in claiming that LCFRSs cannot describe scrambling.

1.3 TAG Variants Proposed for Scrambling

The problem with long-distance scrambling and TAG is that the trees representing the syntax of scrambled German subordinate clauses do not have the simple nested structure that ordinary TAG generates. The CETM requires that (positions for) all of the arguments of the lexical anchor of an elementary tree be included in that tree. But in a scrambled tree, the arguments of several verbs are interleaved freely. All TAG extensions that have been proposed to accommodate this interleaving involve factoring the elementary structures into multiple components and inserting these components at multiple positions in the course of the derivation.

One of the first proposals made was an analysis of German scrambling data using nonlocal MCTAG with additional dominance constraints (Becker, Joshi, and Rambow 1991). However, the formal properties of nonlocal MCTAG are not well understood, and

3 See also Gerdes (2002) for a discussion of the limitation of TAG with respect to scrambling in German.

it is assumed that the formalism is not polynomially parsable. Therefore this approach is no longer pursued, but it has influenced the different subsequent proposals.

An alternative formalism for scrambling is V-TAG (Rambow 1994a, 1994b; Rambow and Lee 1994), a formalism that has nicer formal properties than nonlocal MCTAG. V-TAG also uses multicomponent sets (**vectors**) for scrambled elements; in this it is a variant of MCTAG. Additionally, there are dominance links among the trees of the same vector. In contrast to MCTAG, the trees of a vector in V-TAG are not required to be added simultaneously. The lexicalized V-TAGs that are of interest for natural languages are polynomially parsable. Rambow (1994a) proposes detailed analyses of a large range of different word order phenomena in German using V-TAG and thereby shows the linguistic usefulness of V-TAG.

Even though V-TAG does not pose the problems of nonlocal MCTAG in terms of parsing complexity, it is still a nonlocal formalism in the sense that, as long as the dominance links are respected, arbitrary nodes can be chosen to attach the single components of a vector. Therefore, in order to formulate certain locality restrictions (e.g., for *wh*-movement and also for scrambling), one needs an additional means of putting constraints on what can interleave with the different trees of a vector, or in other words, constraints on how far a dominance link can be stretched. V-TAG allows us to put **integrity constraints** on certain nodes that disallow the occurrence of these nodes between two trees linked by a dominance link. This has the effect of making these nodes act as barriers. With integrity constraints, constructions involving long-distance movements can be correctly analyzed. But the explicit marking of barriers is somewhat against the original appealing TAG idea that such constraints result from imposition of the CETM, according to which the position of the moved element and the verb it depends on must be in the same elementary structure, and from the further combination possibilities of this structure. In other words, in local formalisms with an extended domain of locality such as TAG or tree-local and set-local MCTAG, such constraints result from the form of the elementary structures and from the locality of the derivation operation. That is, they follow from general properties of the grammar, and they need not be stated explicitly. This is one of the aspects that make TAG so attractive from a linguistic point of view, and it gets lost in nonlocal TAG variants.

D-tree substitution grammars (DSGs) (Rambow, Vijay-Shanker, and Weir 2001) are another TAG variant one could use for scrambling. DSGs are a description-based formalism; that is, the objects a DSG deals with are tree descriptions. A problem with DSG is that the expressive power of the formalism is probably too limited to deal with all natural language phenomena: According to Rambow, Vijay-Shanker, and Weir (2001) it "does not appear to be possible for DSG to generate the copy language" (page 101). This means that the formalism is probably not able to describe cross-serial dependencies in Swiss German. Furthermore, DSG is nonlocal and therefore, as in the case of V-TAG, additional constraints (**path constraints**) have to be placed on material interleaving with the different parts of an elementary structure.

Another TAG variant using tree descriptions is local tree description grammar (TDG) (Kallmeyer 2001). Local TDG can be used for scrambling in a way similar to DSG or V-TAG. The languages generated by local TDGs are semilinear. However, the formalism allows one to generate tree descriptions with underspecified dominance relations, and the process of resolving the remaining dominance links is nonlocal. Therefore one may have the same problem as in the case of DSG and V-TAG. Furthermore, so far it has not been shown that the formalism is polynomially parsable, and it is not clear whether such parsing is possible without any additional constraint or limitation on the underspecified tree descriptions.

A further TAG variant proposed in order to deal with scrambling is segmented tree-adjoining grammar (SegTAG) (Kulick 2000). SegTAG uses an operation on trees called **segmented adjunction** that consists partly of a standard TAG adjunction and partly of a kind of tree merging or tree unification. In this operation, two different things get mixed up, the more or less resource-sensitive adjoining operation of standard TAG, in which subtrees cannot be identified,[4] and the completely different unification operation. Perhaps using tree descriptions instead of trees, a more coherent definition of SegTAG can be achieved. But we will not pursue this here.

The formal properties of SegTAG are not clear. Kulick (2000) suggests that SegTAGs are probably in the class of LCFRSs, but there is no actual proof of this. However, if SegTAG is in LCFRS, the generative power of the formalism is probably too limited to deal with scrambling in a general way. But it seems that the limit imposed by the grammar on the complexity of the scrambling data is fixed but arbitrarily high. (With increasing complexity, the elementary trees, however, get larger and larger.) This means that one can probably define a SegTAG that can analyze scrambling up to some complexity level n for any $n \in \mathbb{N}$. (A definition of what a complexity level is, is not given; it is perhaps the depth of scrambling.) In this sense, a general treatment of scrambling might be possible. We follow a similar approach in this article by proposing a mildly context-sensitive formalism that can deal with scrambling up to some fixed complexity limit n that can be chosen arbitrarily high.

All these TAG variants are interesting with respect to scrambling, and they give a great deal of insight into what kind of structures are needed for scrambling. But as explained above, none of them is entirely satisfying. The most convincing one is V-TAG, since this formalism can deal with scrambling, lexicalized V-TAG is polynomially parsable, and the set of languages V-TAG generates contains the set of all tree-adjoining languages (TALs) (in particular, the copy language). Furthermore, a large range of word order phenomena has been treated with V-TAG, and thereby the usefulness of V-TAG for linguistic applications has been shown. But as already mentioned, V-TAG has the inconvenience of being a nonlocal formalism. For the reasons explained above, it is desireable to find a local TAG extension for scrambling (as opposed to the nonlocality of derivations in V-TAG, DSG, and nonlocal MCTAG) such that locality constraints for movements follow only from the form of the elementary structures and from the local character of derivations. This article proposes a local TAG variant that can deal with scrambling (at least with an arbitrarily large set of scrambling phenomena), that is polynomially parsable, and that properly extends TAG in the sense that the set of all TALs is a proper subset of the languages it generates.

In section 2, tree-local MCTAG with shared nodes (SN-MCTAG) and in particular restricted SN-MCTAG (RSN-MCTAG) are introduced, formalisms that extend TAG in the sense mentioned above. Section 3 shows linguistic applications of RSN-MCTAG, in particular, an analysis of scrambling. In section 4, a relation between RSN-MCTAG and range concatenation grammar (RCG) (Boullier 1999, 2000) is established. This relation allows us to show that certain subclasses of RSN-MCTAG are mildly context-sensitive and therefore in particular polynomially parsable. These subclasses do not cover all cases of long-distance scrambling but, in contrast to TAG, they cover an arbitrarily large

4 More precisely, only the root of the new elementary tree and eventually (i.e., in the case of an adjunction) the foot node get identified with the node the new tree attaches to. But there is no unification of whole subtrees. Consequently, every edge occurring in the derived tree comes from exactly one edge in an elementary tree, and every edge from the elementary trees used in the derivation occurs exactly once in the derived tree. In this sense the operation is resource-sensitive.

set, providing scrambling analyses that respect the CETM. This means that the limit they impose on the complexity of the scrambling data one can analyze is variable. Based on empirical studies, it can be chosen sufficiently great such that the grammar covers all scrambling cases that one assumes to occur.

2. The Formalism

An informal introduction of (restricted) tree-local MCTAG with shared nodes can also be found in Kallmeyer and Yoon (2004).

2.1 Motivation: The Idea of Shared Nodes

Let us consider again example (1) in order to illustrate the general idea of shared nodes. In standard TAG, nodes to which new elementary trees are adjoined or substituted disappear; that is, they are replaced by the new elementary tree. For example, after having performed the derivation steps shown in Figure 2, the root node of the *reparieren* tree does not exist any longer. It is replaced by the *verspricht* tree, and its daughters have become daughters of the foot node of the *verspricht* tree. That is, the root node of the derived tree is considered to belong only to the *verspricht* tree. Therefore, an adjunction at that node is an adjunction at the *verspricht* tree.

However, this standard TAG view is not completely justified: In the derived tree, the root node and the lower S node might as well be considered to belong to *reparieren*, since they are results of identifying the root node of *reparieren* with the root and the foot node of *verspricht*.[5] Therefore, we propose that the two nodes in question belong to both *verspricht* and *reparieren*. In other words, these nodes are shared by the two elementary trees. Consequently, they can be used to add new elementary trees to *verspricht* and (in contrast to standard TAG) also to *reparieren*.

In the following, we use an MCTAG, and we assume tree-locality; that is, the nodes to which the trees of such a set are added must all belong to the same elementary tree. Standard tree-local MCTAGs are weakly and even strongly equivalent to TAGs, but they allow us to generate a richer set of derivation structures. In combination with shared nodes, tree-local multicomponent derivation extends the weak generative power of the grammar (see Figure 4 for a sample tree-local MCTAG with shared nodes that generates a language that is not a tree-adjoining language).[6]

Let us go back to example (1). Assume the tree set in Figure 3 for the scrambled NP *es*. If the idea of shared nodes is adopted, this tree set can be added to *reparieren* using the root of the derived tree for adjunction of the first tree and the NP_{acc} substitution node for substitution of the second tree. The operation is tree-local, since both nodes are part of the *reparieren* tree.

5 Actually, in a feature-structure based TAG (FTAG) (Vijay-Shanker and Joshi 1988), the top feature structure of the root of the derived tree is the unification of the top of the root of *verspricht* and the top of the root of *reparieren*. The bottom feature structure of the lower S node is the unification of the bottom of the foot of *verspricht* and the bottom of the root of *reparieren*. In this sense, the root of the *reparieren* tree gets split into two parts. The upper part merges with the root node of the *verspricht* tree, and the lower part merges with the foot node of the *verspricht* tree.

6 In a way, the idea of node sharing is already present in description-based definitions of TAG-related formalisms (see Vijay-Shanker 1992; Rogers 1994; Kallmeyer 2001). This is why these formalisms are monotonic with respect to the node properties described in the tree descriptions. However, the possibility of exploiting this in order to obtain multiple adjunctions combined with multicomponent tree descriptions has not been pursued so far.

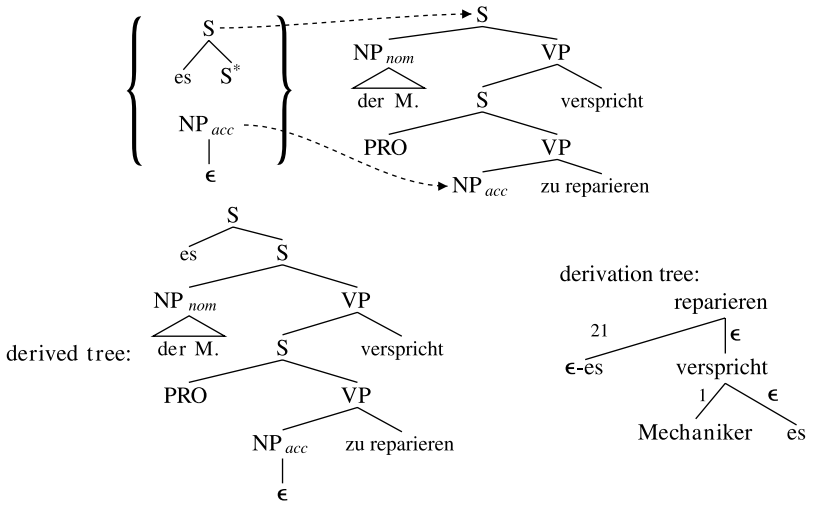


Figure 3
Derivation of (1) *dass es der Mechaniker zu reparieren verspricht* ('that the mechanic promises to repair it') using shared nodes.

The notion of shared nodes means in particular that a node can be used for more than one adjunction. (E.g., in Figure 3, two trees were adjoined at the root of the *reparieren* tree.) A similar idea has led to the definition of **extended derivation** in Schabes and Shieber (1994). For certain auxiliary trees, Schabes and Shieber allow more than one adjunction at the same node. However, the definition of the derived tree in Schabes and Shieber (1994) is such that if first β_1 and then β_2 are adjoined at some node μ (i.e., in the derivation tree there are edges from some γ to β_1 and β_2, both with the position p of the node μ in γ), then first the whole tree derived from β_1 is added to position p, and afterwards the whole tree derived from β_2 is added to position p. In other words, before β_2 is adjoined, all the trees to be added by adjunction or substitution to β_1 must be added. This is different in the case of shared nodes: After β_1 and then β_2 have been adjoined, the root node of β_2 in the derived tree is shared by β_1 and β_2 and consequently can be used for adjunctions at β_1.[7] In other words, trees to be adjoined at the roots of β_1 and β_2 can be adjoined in any order. This is important for obtaining all the possible permutations of scrambled elements.

2.2 Formal Definition of Tree-Local MCTAG with Shared Nodes

As already mentioned, the idea of tree-local MCTAG with shared nodes is the following: In the case of a substitution of an elementary tree α into an elementary tree γ, in the resulting tree, the root node of the subtree α is considered to be part of α and of γ. Similarly, when an elementary tree β is adjoined at a node that is part of the elementary trees $\gamma_1, \ldots, \gamma_n$, then in the resulting tree, the root and foot node of β are both considered to be part of $\gamma_1, \ldots, \gamma_n$ and β. Consequently, if an elementary tree γ' is added to an elementary tree γ, and if there is then a sequence of adjunctions at root

7 In this case, one obtains crossed dotted edges in the SN-derivation structure defined later (see Figure 14 for an example).

or foot nodes starting from γ', then each of these adjunctions can be considered an adjunction at γ, since it takes place at a node shared by γ, γ', and all the subsequently adjoined trees.

Therefore, one way to define SN-MCTAG refers to the standard TAG derivation tree in the following way. Define the grammar as an MCTAG and then allow only derivation trees that satisfy the following tree-locality condition: For each instance $\{\gamma_1, \ldots, \gamma_k\}$ of an elementary tree set in the derivation tree, there is a γ such that each of the γ_i is either a daughter of γ or is linked to one of the daughters of γ by a chain of adjunctions at root or foot nodes.

As an example, consider the derivation tree for (1) in Figure 3. It shows that the trees used in the derivation are the *reparieren* tree, the *verspricht* tree, the *Mechaniker* tree, and the two trees *es* and ϵ-*es* from the tree set in Figure 3. ϵ-*es* is substituted into *reparieren* at position 21, and *verspricht* is adjoined to *reparieren* at position ϵ. Then, *Mechaniker* is substituted into *verspricht* at position 1, and *es* is adjoined to *verspricht* at position ϵ. The derivation is tree-local in the node-sharing sense, since for the tree set $\{\epsilon$-*es*, *es*$\}$, ϵ-*es* is a daughter of *reparieren* in the derivation tree and *es* is linked to *reparieren* by a first adjunction of *verspricht* to *reparieren* and a further adjunction of *es* to the root of *verspricht*.

In the following, we adopt this way of viewing derivations in SN-MCTAG as specific multicomponent TAG derivations; that is, we define SN-MCTAG as a variant of MCTAG. This avoids formalizing a notion of shared nodes, even though this was the starting motivation for the formalism.

We assume a definition of TAG as a tuple $G = \langle I, A, N, T \rangle$ with I being the set of initial trees, A the set of auxiliary trees, and N and T the nonterminal and terminal node labels, respectively (see, for example, Vijay-Shanker [1987] for a formal definition of TAG). Now we formally introduce multicomponent tree-adjoining grammars (Joshi 1987; Weir 1988):

Definition 1
A **multicomponent tree-adjoining grammar** is a tuple $G = \langle I, A, N, T, \mathcal{A} \rangle$ such that

- $G_{TAG} := \langle I, A, N, T \rangle$ is a TAG;
- $\mathcal{A} \subseteq P(I \cup A)$ is a set of subsets of $I \cup A$, the set of elementary tree sets.[8]

$\gamma \Rightarrow \gamma'$ is a **multicomponent derivation step** in G iff there is an instance $\{\gamma_1, \ldots, \gamma_n\}$ of an elementary tree set in $\mathcal{A}$ and there are pairwise different node addresses $p_1, \ldots, p_n$ such that $\gamma' = \gamma[p_1, \gamma_1] \ldots [p_n, \gamma_n]$, where $\gamma[p_1, \gamma_1] \ldots [p_n, \gamma_n]$ is the result of adding the γ_i $(1 \leq i \leq n)$ at node positions p_i in γ.[9]

As in TAG, a derivation starts from an initial tree, and in the end, in the final derived tree, there must not be any obligatory adjunction constraint, and all leaves must be labeled by a terminal or by the empty word.

In each MCTAG derivation step, an elementary tree set is chosen, and the trees from this set are added to the already derived tree. Since they are added to pairwise different

8 $P(X)$ is the set of subsets of some set X.

9 As usual (see Vijay-Shanker 1987; Weir 1988), $\gamma[p, \gamma']$ is defined as follows: If γ' is (derived from) an initial tree and the node at position p in γ is a substitution node, then $\gamma[p, \gamma']$ is the tree one obtains by substitution of γ' into γ at node position p. If γ' is (derived from) an auxiliary tree and the node at position p in γ is an internal node, then $\gamma[p, \gamma']$ is the tree one obtains by adjunction of γ' to γ at node position p. Otherwise $\gamma[p, \gamma']$ is undefined.

nodes, one can just as well add them one after the other; that is, each multicomponent derivation in an MCTAG $G = \langle I, A, N, T, \mathcal{A} \rangle$ corresponds to a derivation in the TAG $G_{TAG} := \langle I, A, N, T \rangle$. Let us define the **TAG derivation tree** of such a multicomponent derivation as the corresponding derivation tree in G_{TAG}. We can then define tree-local, set-local, and nonlocal MCTAG and also the different variants of SN-MCTAG this article deals with by putting different constraints on this derivation tree.[10] Note that for each operation $\gamma[p, \gamma_i]$, the node address p in the derived tree γ points at a node that is at some address p' in some elementary tree γ' that was already added (γ' and p' are unique). In the TAG derivation tree, there will be in this case an edge from γ' to γ_i with position p'.

A TAG derivation tree can be considered a tuple of nodes and edges. As usual in finite trees, the edges are directed from the mother node to the daughter. Linear precedence is not needed in a derivation tree, since it does not influence the result of the derivation. So a derivation tree is a tuple $\langle \mathcal{N}, \mathcal{E} \rangle$, with $\mathcal{N}$ being a finite set of instances of elementary trees and with $\mathcal{E} \subset \mathcal{N} \times \mathcal{N} \times \mathbb{N}^*$, where $\mathbb{N}^*$ is the set of Gorn addresses. We define the **parent** relation as the relation between mothers and daughters in a derivation tree, the **dominance** relation as the reflexive transitive closure of the parent relation, and the **node-sharing** relation as the relation between nodes that either are mother and daughter or are linked first by a substitution/adjunction and then a chain of adjunctions at root or foot nodes:

Definition 2
Let $D = \langle \mathcal{N}, \mathcal{E} \rangle$ be a derivation tree in a TAG.

- $\mathcal{P}_D := \{\langle n_1, n_2 \rangle \mid n_1, n_2 \in \mathcal{N}$, and there is a $p \in \mathbb{N}^*$ such that $\langle n_1, n_2, p \rangle \in \mathcal{E}\}$ is the **parent** relation in D.
- $\mathcal{D}_D := \{\langle n_1, n_2 \rangle \mid n_1, n_2 \in \mathcal{N}$, and either $n_1 = n_2$, or there is a n_3 such that $\langle n_1, n_3 \rangle \in \mathcal{P}_D$ and $\langle n_3, n_2 \rangle \in \mathcal{D}_D\}$ is the **dominance** relation in D.
- $\mathcal{SN}_D := \{\langle n_1, n_2 \rangle \mid$ either $\langle n_1, n_2 \rangle \in \mathcal{P}_D$ or there are $t_1, \ldots, t_k \in \mathcal{N}$, such that $\langle n_1, t_1 \rangle \in \mathcal{P}_D$, $n_2 = t_k$ and for all $j, 1 \leq j \leq k-1$: $\langle t_i, t_{i+1}, p' \rangle \in \mathcal{E}$ with either $p' = \epsilon$ or t_i being an auxiliary tree with foot node address $p'\}$ is the **node-sharing** relation in D.

A node-sharing relation $\langle \gamma_1, \gamma_2 \rangle$ that corresponds to an actual parent relation is called a **primary** node-sharing relation, and γ_2 is called a **primary SN-daughter** of γ_1, whereas any other node-sharing relation $\langle \gamma_1, \gamma_2 \rangle$ is called **secondary** and in this case γ_2 is called a **secondary SN-daughter** of γ_1.

The TAG derivation trees for MCTAG derivations have certain properties resulting from the requirement that the elements of instances of elementary tree sets must be added simultaneously to the already derived tree: First, if an elementary tree set is used, then all trees from this set must occur in the derivation tree. Secondly, one tree from an elementary tree set cannot be substituted or adjoined into another tree from the same set, and, thirdly, two tree sets cannot be interleaved. For nonlocal MCTAG, these are all constraints the TAG derivation tree needs to satisfy.

10 This TAG derivation tree is not the MCTAG derivation tree defined in Weir (1988). The nodes of Weir's MCTAG derivation trees are labeled by sequences of elementary trees (i.e., by elementary tree sets), and each edge stands for simultaneous adjunctions/substitutions of all elements of such a set.

Lemma 1
Let $G = \langle I, A, N, T, \mathcal{A}\rangle$ be an MCTAG, $G_{TAG} := \langle I, A, N, T\rangle$. Let $D = \langle \mathcal{N}, \mathcal{E}\rangle$ be a derivation tree in G_{TAG} with the corresponding derived tree t being in $L(G_{TAG})$.

D is a possible TAG derivation tree in G with $t \in L(G)$ iff D is such that

- (MC1) The root of D is an instance of an initial tree $\alpha \in I$ and all other nodes are instances of trees from tree sets in $\mathcal{A}$ such that for all instances Γ of elementary tree sets from $\mathcal{A}$ and for all $\gamma_1, \gamma_2 \in \Gamma$, if $\gamma_1 \in \mathcal{N}$, then $\gamma_2 \in \mathcal{N}$.
- (MC2) For all instances Γ of elementary tree sets from $\mathcal{A}$ and for all $\gamma_1, \gamma_2 \in \Gamma, \gamma_1 \neq \gamma_2$: $\langle \gamma_1, \gamma_2\rangle \notin \mathcal{D}_D$.
- (MC3) For all pairwise different instances $\Gamma_1, \Gamma_2, \ldots, \Gamma_n, n \geq 2$, of elementary tree sets from $\mathcal{A}$, there are no $\gamma_1^{(i)}, \gamma_2^{(i)} \in \Gamma_i, 1 \geq i \geq n$, such that $\langle \gamma_1^{(1)}, \gamma_2^{(n)}\rangle \in \mathcal{D}_D$ and $\langle \gamma_1^{(i)}, \gamma_2^{(i-1)}\rangle \in \mathcal{D}_D$ for $2 \geq i \geq n$.

The proof of this lemma is given in the appendix. The lemma gives us a way to characterize nonlocal MCTAG via the properties of the TAG derivation trees the grammar licenses. With this characterization we get rid of the original simultaneity requirement: The corresponding properties are now captured in the three constraints (MC1)–(MC3). But since these constraints need to hold only for the TAG derivation trees that correspond to derived trees in the tree language, subderivation trees need not satisfy them. In other words, γ_1 and γ_2 from the same elementary tree set can be added at different moments of the derivation as long as the final complete TAG derivation tree satisfies (MC1)–(MC3).

We now define tree-local, set-local, SN-tree-local, and SN-set-local TAG derivation trees by imposing further conditions. Basically, the difference between the first two and their SN variants is that in the first two, the definition refers to the parent relation, whereas in the second two, it refers to the node-sharing relation.

Definition 3
Let $G = \langle I, A, N, T, \mathcal{A}\rangle$ be an MCTAG. Let $D = \langle \mathcal{N}, \mathcal{E}\rangle$ be a TAG derivation tree for some $t \in L(\langle I, A, N, T\rangle)$.

- D is a **multicomponent** derivation tree iff it satisfies (MC1)–(MC3).
- D is **tree-local** iff for all instances $\{\gamma_1, \ldots, \gamma_n\}$ of elementary tree sets with $\gamma_1, \ldots, \gamma_n \in \mathcal{N}$, there is one γ such that $\langle \gamma, \gamma_1\rangle, \ldots, \langle \gamma, \gamma_n\rangle \in \mathcal{P}_D$.
- D is **set-local** iff for all instances $\{\gamma_1, \ldots, \gamma_n\}$ of elementary tree sets with $\gamma_1, \ldots, \gamma_n \in \mathcal{N}$, there is an instance Γ of an elementary tree set such that for all $1 \leq i \leq n$, there is a $t_i \in \Gamma$ with $\langle t_i, \gamma_i\rangle \in \mathcal{P}_D$.
- D is **SN-tree-local** iff for all instances $\{\gamma_1, \ldots, \gamma_n\}$ of elementary tree sets with $\gamma_1, \ldots, \gamma_n \in \mathcal{N}$, there is one γ such that $\langle \gamma, \gamma_1\rangle, \ldots, \langle \gamma, \gamma_n\rangle \in \mathcal{SN}_D$.
- D is **SN-set-local** iff for all instances $\{\gamma_1, \ldots, \gamma_n\}$ of elementary tree sets with $\gamma_1, \ldots, \gamma_n \in \mathcal{N}$, there is an instance Γ of an elementary tree set such that for all $1 \leq i \leq n$, there is a $t_i \in \Gamma$ with $\langle t_i, \gamma_i\rangle \in \mathcal{SN}_D$.

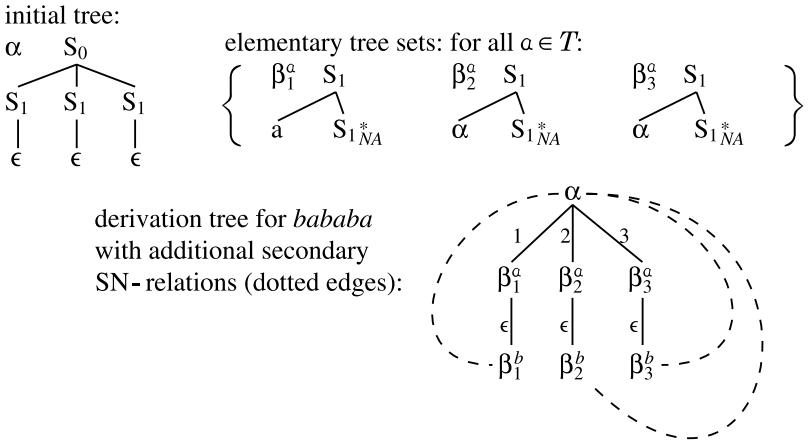


Figure 4
SN-MCTAG for $\{w^3 \mid w \in T^*\}$.

The formalism we are proposing for scrambling is MCTAG with SN-tree-local TAG derivation trees. We call these grammars **tree-local MCTAGs with shared nodes**:

Definition 4
Let G be an MCTAG. G is a **tree-local MCTAG with shared nodes** iff the set of trees generated by G, $L_T(G)$, is defined as the set of those trees that can be derived with an SN-tree-local multicomponent TAG derivation tree in G.

As usual, the string language $L_S(G)$ is then defined as the set of strings yielded by the trees in $L_T(G)$.

All tree-adjoining languages can be generated by SN-MCTAGs, since a TAG corresponds to an MCTAG with unary multicomponent sets. For such an MCTAG, each TAG derivation tree is trivially SN-tree-local. In other words, in this case the tree sets are the same, whether the grammar is considered a TAG, a tree-local MCTAG, or an SN-MCTAG.[11] In particular, all TAG analyses proposed so far can be maintained, since each TAG is trivially also an instance of SN-MCTAG.

SN-MCTAG is a proper extension of TAG (and of tree-local MCTAG) in the sense that there are languages that can be generated by an SN-MCTAG but not by a TAG. As an example, consider Figure 4, which shows an SN-MCTAG for $\{www \mid w \in T^*\}$.[12] Similar to the grammar in Figure 4, for all copy languages $\{w^n \mid w \in T^*\}$ for some $n \in \mathbb{N}$, an SN-MCTAG can be found. Other languages that can be generated by SN-MCTAG and that are not TALs are the counting languages $\{a_1^n \dots a_k^n \mid n \geq 1\}$ for any $k > 4$ (for $k \leq 4$, these languages are tree-adjoining languages).

There are two crucial differences between V-TAG and SN-MCTAG: First, in V-TAG, the adjunctions of auxiliary trees from the same set need not be simultaneous. In this respect, V-TAG differs not only from SN-MCTAG, but from any of the different

11 However, viewing a TAG as an SN-MCTAG allows us to obtain a richer set of SN-derivation structures, as introduced in the next section. This is exploited in Kallmeyer (2002) for semantics.

12 The subscript *NA* in the figure stands for *null adjunction*; that is, it disallows adjunctions at the node in question.

MCTAGs mentioned above. Secondly, V-TAG is nonlocal in the sense of nonlocal MC-TAG, whereas SN-MCTAG is local, even though the locality is not based on the parent relation in the derivation tree, as is the case in standard local MCTAG, but on the SN-dominance relation in the derivation tree. As a consequence of the locality, we do not need dominance links (i.e., dominance constraints that have to be satisfied by the derived tree) in SN-MCTAG, in contrast to other TAG variants for scrambling. The locality condition put on the derivation sufficiently constrains the possibilities for attaching the trees from elementary tree sets: Different trees from a tree set attach to different nodes of the same elementary tree. Consequently, the dominance relations among these different nodes determine the dominance relations among the different trees from the tree set. Therefore extra dominance links are not necessary. This is different for nonlocal TAG variants such as V-TAG or DSG, in which one can in principle attach the different components of an elementary structure at arbitrary nodes in the derived tree.

2.3 SN-MCTAG and Scrambling: Formal Considerations

Figure 5 shows an SN-MCTAG generating a language that cannot even be generated by linear context-free rewriting systems (see Becker, Rambow, and Niv [1992] for a proof), and therefore not by set-local MCTAG. This example, however, concerns neither weak nor strong generative capacity, but something that Becker, Rambow, and Niv (1992) call **derivational capacity**: the derivation of $n^k v^k$ must be such that the $\pi(i)$th n and the ith v come from the same elementary tree set in the grammar.

The grammar in Figure 5 works in the following way: Each derivation starts with α. Then a first instance of the tree set (yielding n_1 and v_1) is added to the N and V nodes in α. For each further instance of the tree set (yielding n_i and v_i), β_v is adjoined to the root node of the β_v tree of v_{i-1}. Therefore all β_v adjunctions except the first are occurring at root nodes, and consequently all β_v are (primary or secondary) SN-daughters of α. The β_n tree of n_i can be adjoined to any of the root or foot nodes of the β_n that have already been added, since in this way all adjunctions of β_n except the first one occur at root or foot nodes, and therefore all these β_n are SN-daughters of α. This allows us to place n_i at any position in the string already containing $\{n_1, \ldots, n_{i-1}\}$, and thereby any permutation of the ns can be obtained. Since all nodes in the derivation tree are SN-daughters of α, the derivation is SN-tree-local. Note that in the grammar in Figure 5, there is no *NA* constraint on the foot node of the first auxiliary tree in the tree set. This is crucial for allowing all permutations of the $n_1, \ldots, n_k$. In this respect, the elementary trees differ from what is usually done in TAG.

Becker, Rambow, and Niv (1992) argue that a formalism that cannot generate the language in Figure 5 is not able to analyze scrambling in an adequate way. We think,

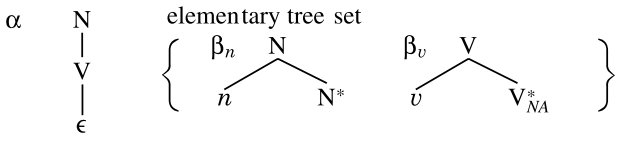


Figure 5
SN-MCTAG for $\{n_{\pi(k)} \ldots n_{\pi(1)} v_k \ldots v_1 \mid k \geq 0, n_i = n, v_i = v$, and n_i and v_i are in the same elementary tree set and they were added in the ith derivation step for all $i, 1 \leq i \leq k$, and π is a permutation of $(1, \ldots, k)\}$.

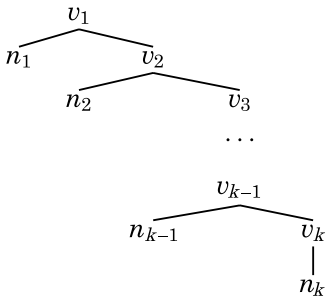


Figure 6
Predicate argument structure for *SCR*.

however, that this language is not exactly what one needs for scrambling. The assumption underlying the language in Figure 5 is that n_i is an argument of v_i. But in this case, instead of adding n_i and v_i at the same time, n_i should be added to v_i. If one makes the additional assumption that argument NPs are added by substitution, then one can require that the argument NPs have already been substituted (this is what Joshi, Becker, and Rambow [2000] call the weak co-occurrence constraint), that is, that the tree for v_i contain n_i. In this case, the language in Figure 5 is an appropriate test language for scrambling. But we do not want to make this assumption.

Furthermore, there are more predicate-argument dependencies: v_i is also an argument of v_{i-1} for $i \leq 2$. This is what Joshi, Becker, and Rambow (2000) call the strong co-occurrence constraint. In other words, the dependency tree should be as in Figure 6.

Additionally to the permutation of the $n_1, \ldots, n_k$, also the v_i can be moved leftward, as long as they do not permute among themselves. Consequently, for scrambling data (without extraposition), one rather wants to generate the following language: $SCR := \{w = \pi(n_1 \ldots n_k v_1 \ldots v_k) \mid k \geq 1, n_i = n, v_i = v, \text{for all } 1 \leq i \leq k, \text{and } \pi \text{ is a permutation of } n_1 \ldots n_k v_1 \ldots v_k \text{ such that } n_i \text{ precedes } v_i \text{ in } w \text{ for all } 1 \leq i \leq k \text{ and } v_i \text{ precedes } v_{i-1} \text{ in } w \text{ for all } 1 < i \leq k\}$ with the derivation structure in Figure 6. An SN-MCTAG generating this language is shown in Figure 7.

The SN-MCTAG in Figure 7 yields the following derivations: Either start with α_2, in which case an instance of $\{\beta_n, \alpha_n\}$ must be added and nv is obtained with n depending on v, or start with α_1 for v_1, in which case, for all v except the leftmost one, the set $\{\beta_{v1}, \alpha_{v_1}\}$ is added, for the leftmost v, a set $\{\beta_{v2}, \alpha_{v2}\}$ is added, and for all the ns, sets $\{\beta_n, \alpha_n\}$ are added. These sets can be added in any order; the auxiliary tree is always adjoined to the root node of the already derived tree that is shared by all auxiliary trees that have been used so far and by the first α_1. The initial tree is primarily substituted

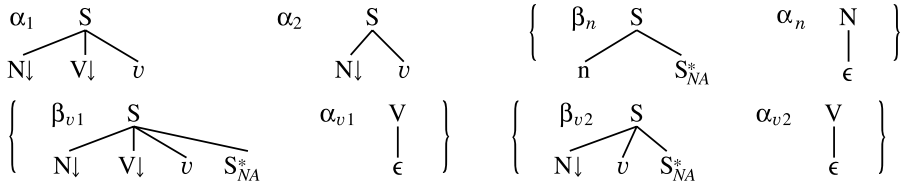


Figure 7
SN-MCTAG for *SCR*.

into the argument slot it fills. So the only condition for adding such a tree set is that the verb it depends on has already been added, since the tree of this verb provides the substitution node for the initial tree. Therefore, since the lexical material is always left of the foot node, one obtains that v_i precedes v_{i-1} for all $1 < i \leq k$ and n_i precedes v_i for all $1 \leq i \leq k$.

Note that in Figure 7, for a scrambled n_i, the substitution node is filled with an empty node, while the n is adjoined higher at a node that is not yet available in the elementary structure of v_i. So the combination of n_i and v_i cannot be precompiled here.

2.4 Restricted SN-MCTAG

When the formal properties of SN-MCTAG are examined, it becomes clear that the formalism is hard to compare to other local TAG-related formalisms, since in the derivation tree, arbitrarily many trees can be secondary SN-daughters of a single elementary tree, such that these secondary links are considered to be adjunctions to that tree. This means that these secondary links are relevant for the SN-tree-locality of the derivation. An example is the grammar in Figure 5, in which in each derivation step, the relevant node-sharing relations are the links between α and the two auxiliary trees of the new set. This means that for a word of length k, there are k SN-daughters of α that are relevant for the SN-tree-locality of the derivation. The grammar in Figure 5 indicates that this property of SN-MCTAG is at least partly responsible for the fact that SN-MCTAG allows us to generate languages that are not even mildly context-sensitive (i.e., that are not in the class of languages that can be generated by LCFRS). However, it would be desirable to stay inside the class of mildly context-sensitive languages. Therefore, in the following, we define a restricted version, RSN-MCTAG, that limits the number of relevant secondary SN-daughters of an elementary tree. The restriction is obtained as follows: We require that in each derivation step, among the SN-relations between the old γ and the new set Γ, there be at least one primary SN-relation. The number of primary SN-daughters of a specific elementary tree is limited, since the primary SN-daughters correspond to substitutions/adjunctions at pairwise different nodes and the number of nodes in an elementary tree is limited. Consequently, the number of relevant secondary SN-daughters for a node is limited as well.

An example of a derivation satisfying the new constraint is that in Figure 3, in which *es* is a secondary SN-daughter of *reparieren*, while the second element of the tree set, ϵ-*es*, is a primary SN-daughter of *reparieren*.

Definition 5

- Let $G = \langle I, A, N, T, \mathcal{A}\rangle$ be an MCTAG. Let $D = \langle \mathcal{N}, \mathcal{E}\rangle$ be the TAG derivation tree of a tree $t \in L_T(\langle I, A, N, T\rangle)$. D is **RSN-tree-local** iff for all instances $\{\gamma_1, \ldots, \gamma_n\}$ of an elementary tree set with $\gamma_1, \ldots, \gamma_n \in \mathcal{N}$, there is one γ such that

  1. $\langle \gamma, \gamma_1\rangle, \ldots, \langle \gamma, \gamma_n\rangle \in \mathcal{SN}_D$;
  2. there is one $i, 1 \leq i \leq n$, with $\langle \gamma, \gamma_i\rangle \in \mathcal{P}_D$.

- An MCTAG G is called a **restricted SN-MCTAG** iff the set of trees generated by G, $L_T(G)$, is defined as the set of those trees that can be derived with an RSN-tree-local multicomponent TAG derivation tree in G.

The first condition of the definition says that the grammar is SN-tree-local, and the second condition ensures that at least one of the relevant SN-daughters of γ is a primary SN-daughter, that is, an actual daughter of γ.

As for SN-MCTAG, all tree-adjoining languages can also be generated by RSN-MCTAGs. The sample grammars in Figures 4 and 5 are not RSN-MCTAGs. We suspect that there is no RSN-MCTAG that generates the language in Figure 5. But the grammar in Figure 7 for the language *SCR* is an RSN-MCTAG.

It can be shown that for the TAG derivation trees of an RSN-MCTAG, the following holds: For each instance of an elementary tree set Γ, the γ to which all elements of Γ are linked by node-sharing relations with at least one primary link is unique (which is not necessarily the case for general SN-MCTAG). This is formulated in the following lemma:

Lemma 2
Let $G = \langle I, A, N, T, \mathcal{A} \rangle$ be an RSN-MCTAG. Let $D = \langle \mathcal{N}, \mathcal{E} \rangle$ be a TAG derivation tree in G.

Then for all instances $\{\gamma_1, \ldots, \gamma_n\}$ of elementary tree sets with $\gamma_1, \ldots, \gamma_n \in \mathcal{N}$, there is exactly one γ such that $\langle \gamma, \gamma_1 \rangle, \ldots, \langle \gamma, \gamma_n \rangle \in \mathcal{SN}_D$, and there is one $i, 1 \leq i \leq n$, with $\langle \gamma, \gamma_i \rangle \in \mathcal{P}_D$.

For such an elementary tree set $\{\gamma_1, \ldots, \gamma_n\}$, with γ being the unique elementary tree as described in the lemma, all $\langle \gamma, \gamma_i \rangle \in \mathcal{SN}_D \setminus \mathcal{P}_D$, $1 \leq i \leq n$, are called **secondary adjunction links** in D. The proof of the lemma is given in the appendix.

Now we introduce the SN-derivation structure of a TAG derivation tree D in an RSN-MCTAG. It consists of D enriched with additional links for the secondary adjunctions. These links are equipped with the positions of the first substitutions/adjunctions on the chain that corresponds to the secondary adjunctions.

Definition 6
Let $G = \langle I, A, N, T, \mathcal{A} \rangle$ be an RSN-MCTAG. Let $D = \langle \mathcal{N}, \mathcal{E} \rangle$ be a TAG derivation tree in G. The **SN-derivation structure** of D, D_{SN}, is then $D_{SN} := \langle \mathcal{N}, \mathcal{E}' \rangle$, with $\mathcal{E} \subseteq \mathcal{E}'$.

- For all secondary adjunction links $\langle \gamma_1, \gamma_2 \rangle$ in D with γ' and p such that $\langle \gamma_1, \gamma', p \rangle \in \mathcal{E}$ and $\langle \gamma', \gamma_2 \rangle \in \mathcal{D}_D$: $\langle \gamma_1, \gamma_2, p \rangle \in \mathcal{E}'$.
- These are all elements of $\mathcal{E}'$.

All $e \in \mathcal{E}$ are called **primary edges** in D_{SN}, and all $e \in \mathcal{E}' \setminus \mathcal{E}$ are called **secondary edges** in D_{SN}.

With the notion of the SN-derivation structure, we can formulate the limitation on the maximal number of secondary adjunctions to an elementary tree that we mentioned at the beginning of this section:

Lemma 3
Let $G = \langle I, A, N, T, \mathcal{A} \rangle$ be an RSN-MCTAG. Then there is a constant c such that for all TAG derivation trees D in G with SN-derivation structure $D_{SN} := \langle \mathcal{N}, \mathcal{E} \rangle$, the following holds:

There is no $n \in \mathcal{N}$ such that there exist $m \geq c + 1$ pairwise different $n_1, \ldots, n_m$ such that for all i, $1 \leq i \leq m$, there is a p such that

- either $\langle n, n_i, p \rangle$ is a secondary edge in D_{SN};
- or $\langle n, n_i, p \rangle$ is a primary edge in D_{SN}, and there are no n' and p' such that $\langle n', n_i, p' \rangle$ is a secondary edge in D_{SN}.

That this lemma holds is nearly immediate: Each secondary adjunction must be associated with a primary adjunction or substitution into the same tree instance. There are at most k primary adjunctions or substitutions into any tree instance if k is the maximal number of nodes per elementary tree. Consequently there are at most $k \times n$ secondary adjunctions per node if $n + 1$ is the maximal number of trees per elementary tree set.

In linguistic applications, the SN-derivation structure is intended to reflect the predicate-argument dependencies of a sentence in the following way: For each tree in the SN-derivation structure, if this tree is secondarily adjoined to some other tree γ, then it depends on γ. Otherwise it depends on its mother node in the TAG derivation tree. In this way, the grammar for *SCR* in Figure 7 yields the desired dependency structure.

3. Linguistic Applications

3.1 Scrambling with RSN-MCTAG

In this section, we present a small German grammar that allows us to analyze some cases of scrambling. The aim is not an exhaustive treatment of the phenomenon, but just to show that in principle, an analysis of scrambling in German is possible using RSN-MCTAG. The data to which we restrict ourselves are word order variations of example (3) without extraposition, that is, under the assumption that the order of the verbs is *zu reparieren zu versuchen verspricht*:

(3) …*dass er* *dem Kunden* *das Fahrrad zu reparieren*
…that he$_{nom}$ the customer$_{dat}$ the bike$_{acc}$ to repair

zu versuchen verspricht
to try promises
'…that he promises the customer to try to repair the bike'

The elementary trees and tree sets for example (3) are shown in Figure 8. In contrast to standard TAG practices, which are often guided by technical considerations, we represent all arguments of a verb (including an embedded VP) by substitution nodes. For those parts that might be scrambled, there is a single elementary tree (for the case without scrambling) and a tree set used for scrambling. The tree set contains an auxiliary tree that can be primarily or secondarily adjoined to some root node and a tree with the empty word that is intended to fill the argument position. In order to avoid spurious ambiguities, we assume that whenever a derivation using the single elementary tree is possible, this is chosen.

A scrambled element always adjoins to a VP node, and the scrambled element is to the left of the foot node. Therefore it precedes everything that is below or on the

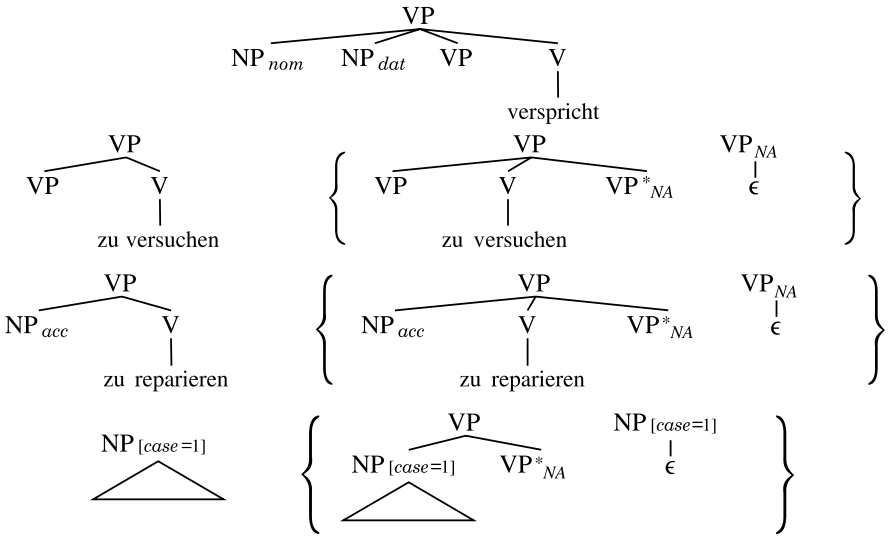


Figure 8
Elementary trees for scrambling.

right of the VP node to which it adjoins. Consequently, given the form of the verbal elementary trees in Figure 8, in which the verb is always below or to the right of all VP nodes allowing adjunction, the order $x\ v$ for an x being a nominal or a verbal argument of v is always respected.

For an element (a lexical item), the tree set for scrambling is used whenever one of the following three cases holds:

- The element is scrambled.
- Scrambling of depth more than one out of the element takes place.
- The element intervenes between some element A (on its right) and some element B (on its left) scrambled out of A, and the element itself does not belong to A.

In other words, the fact that the set for scrambling is used for some element does not necessarily mean that this element is scrambled. It just means that one of the three cases above holds, that is, that some scrambling around this element takes place.

One could actually do without the single trees and always use the tree sets. In this case, even if no scrambling took place, all argument slots would be filled by empty words, and all lexical material would be adjoined to the root node of the derived tree. At first glance, this seems rather odd. But if one does not consider the substitution nodes argument slots but rather some kind of subcategorization features marking which arguments need to be added, an analysis using only the tree sets makes sense. However, for this article, we keep the single trees.

For example (3), a derivation without secondary adjunctions and using only the single trees is possible. Let us consider the following word orders as examples of how secondary adjunction is used for scrambling:

(4) …*dass* er_1 [[*das Fahrrad zu reparieren*] *zu versuchen*]$_2$ t_1 *dem Kunden* t_2
…that he the bike to repair to try the customer

verspricht
promises

(5) …*dass er* [*das Fahrrad zu reparieren*]$_1$ *dem Kunden* [t_1 *zu versuchen*]
…that he the bike to repair the customer to try

verspricht
promises

(6) …*dass* [*das Fahrrad*]$_1$ er_2 [[t_1 *zu reparieren*] *zu versuchen*]$_3$ t_2 *dem Kunden* t_3
…that the bike he to repair to try the customer

verspricht
promises

In example (4), the *versuchen*-VP and *er* are scrambled.[13] Consequently, for *versuchen* and *er*, the sets with two trees are used, whereas for all the other elements, the single trees can be used. In example (5), the *reparieren*-VP is scrambled out of the *versuchen*-VP, with *dem Kunden* intervening between the two. Therefore, the tree sets are used for *reparieren* and *dem Kunden*. For *versuchen*, the single tree can be used, since the scrambling out of *versuchen* is of depth one. In example (6), we have the same scrambling as in example (4), and additionally, *das Fahrrad* is scrambled out of the *reparieren*-VP and the *versuchen*-VP (depth two). Consequently, in this case one needs tree sets for *Fahrrad*, *er*, *versuchen*, and *reparieren*.

Let us consider the analysis of example (4): Starting with *verspricht*, the single tree for *dem Kunden* and the tree set for *versuchen* (with adjunction of the auxiliary tree at the root) are added. This leads to the first tree in Figure 9. The VP nodes in boldface type in the figure are shared by *versuchen* and *verspricht*; that is, they can be used for further adjunction at the *verspricht* tree. (Of course, only the root node can be used for adjunction, since the other nodes have *NA* constraints.) It does not matter in which order *er* and *zu reparieren* are added. For *er*, the tree set is used. The auxiliary tree is secondarily adjoined to the root node, and the initial tree is substituted for the NP_{nom} node in the *verspricht* tree. This leads to the second tree in Figure 9. For *reparieren* and *das Fahrrad*, the single trees are added below the VP substitution node in the *versuchen* tree. The corresponding SN-derivation structure (see Figure 9) contains the desired predicate-argument dependencies. The TAG derivation tree is RSN-tree-local.

Next, let us consider example (5). Here, the single trees for *er* and *versuchen* are added to *verspricht*. This leads to the first tree in Figure 10. The VP node in boldface type in the figure belongs to *verspricht* and *versuchen*. It is next used for secondary adjunction of *dem Kunden* to the *verspricht* tree. The initial tree is substituted at the NP_{dat} slot. This leads to the second tree. Here, the bold VP node belongs to *verspricht*, *versuchen*, and *Kunde*. It is next used for secondary adjunction of the auxiliary tree of *reparieren* to *versuchen*, while the initial tree is substituted for the VP leaf in the *versuchen* tree. This

13 Actually, *er* here is not really scrambled, but since in our formalism, scrambled elements attach at the left of a VP, any other element even more to the left is treated as if it is scrambled (even if it depends on the matrix verb).

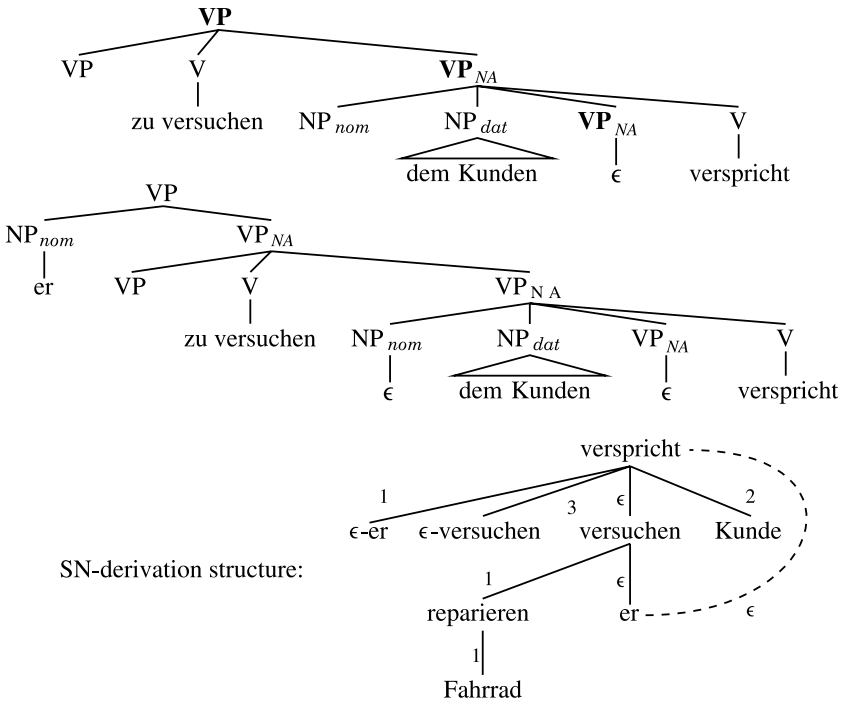


Figure 9
Analysis of example (4).

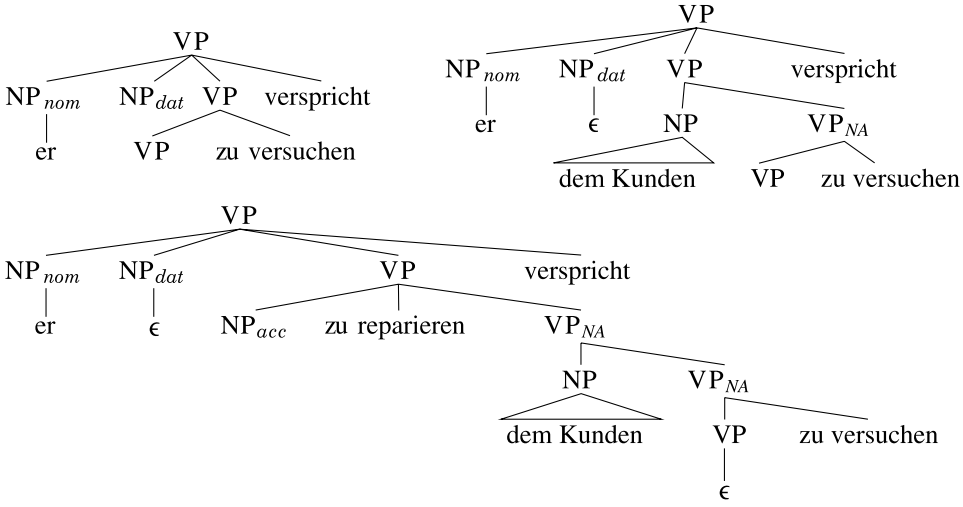


Figure 10
Analysis of example (5).

leads to the third tree. After that, one needs only to add the single tree for *das Fahrrad* to *reparieren*. Note that this is a derivation in which the foot node of the elementary tree containing the lexical material does not dominate the tree with the empty word.

Now let us consider the derivation of example (6). Here, only for *dem Kunden*, the single tree is added by substitution. In all other cases, the tree set is used with (primary

or secondary) adjunction at the root node of the already derived tree. This root node consequently belongs to all verbs that have already occurred in the derivation and can therefore be used to add arguments to any of them.

We leave it to the reader to verify that all word orders can be generated. This kind of analysis also works for more than two embeddings.

Since all scrambled elements attach to a VP node in the elementary tree of the verb they depend on, they cannot attach to the VP of a higher finite verb that embeds the sentence in which the scrambling occurs. In this way, a barrier effect is obtained without establishing any explicit barrier, as is done in V-TAG. Instead, this locality of scrambling is a consequence of the form of the elementary trees and of the locality of the derivations.

Concerning adjunct scrambling, each adjunct has a single auxiliary tree as in standard TAG and additionally a set of two auxiliary trees, a lower auxiliary tree with an empty word and a higher auxiliary tree with the adjunct. This is shown in Figure 11. The internal VP node of the higher tree in the tree set serves as an adjunction site for the lower parts of other adjuncts. Similarly, the elementary trees of verbs need an extra VP node in order to adjoin adverbs.

For more analyses of scrambling, including scrambling in combination with extraposition and topicalization, and also for an extension of the analysis presented here to Korean data, see Kallmeyer and Yoon (2004).

3.2 Raising Verbs and Subject-Auxiliary Inversion

Other phenomena often mentioned in the TAG literature (see, e.g., Rambow, Vijay-Shanker, and Weir 1995; Kulick 2000; Dras, Chiang, and Schuler 2004) as being problematic for TAG and tree-local MCTAG are sentences with raising verbs and subject-auxiliary inversion, as in examples (7) and (8):

(7) Does Gabriel seem to be likely to eat gnocchi?

(8) What does John seem to be certain to like?

The standard TAG analyses of examples (7) and (8) (see Figure 12 for the analysis of example (8)) start with the *eat* and *like* tree, respectively, adjoin an auxiliary tree for *likely* and *certain*, respectively, and then add the trees for *does* and *seem*, respectively. If we assume that these trees are in the same elementary tree set, then this last derivation step is nonlocal, since the *does* tree adjoins to *eat* and *like*, respectively, while the *seem* tree adjoins to *likely* and *certain*, respectively. Though different from scrambling, this problem seems to be of a similar nature, and formalisms that have been proposed for scrambling have also been used to treat these examples (see Kulick 2000).

RSN-MCTAG allows us to analyze examples (7) and (8) in a way that puts *does* and *seem* into a single elementary tree set: After having adjoined *to be likely* and *to be certain*,

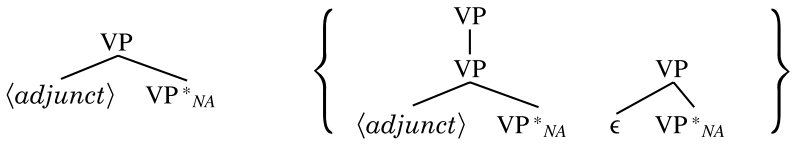


Figure 11
Trees for adjuncts.

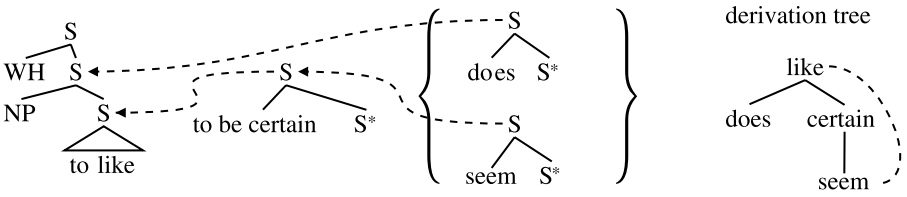


Figure 12
Derivation for (8).

respectively, the root nodes of the adjoined trees are considered still to be part of the elementary trees of *eat* and *like*, respectively. These elementary trees can then be used to add the elementary tree set for *does* and *seem*: Both auxiliary trees are adjoined to these trees. Figure 12 shows the corresponding SN-derivation structure.

4. RSN-MCTAG and Range Concatenation Grammar

In the following, we show that for each RSN-MCTAG of a certain type (i.e., with an additional restriction), a weakly equivalent simple range concatenation grammar (Boullier 1999, 2000) can be constructed. It has been shown that RCGs generate exactly the class of all polynomially parsable languages (Bertsch and Nederhof 2001; appendix A). Furthermore, as shown in Boullier (1998b), simple RCGs in particular are even weakly equivalent to linear context-free rewriting systems (Weir 1988). As a consequence, one obtains that the languages generated by simple RSN-MCTAGs are mildly context-sensitive. This last property was introduced in Joshi (1985). It includes formalisms that are polynomially parsable, are semilinear, and allow only a limited number of crossing dependencies. (We do not give formal definitions of mild context-sensitivity and of LCFRS, since we do not need these definitions in this article.)

Concerning RSN-MCTAGs in general, that is, without any further restriction, we are almost sure that they are not mildly context-sensitive. Perhaps they can even generate languages that are not in the class of languages generated by RCGs.

4.1 Range Concatenation Grammars

This section defines range concatenation grammars.[14]

Definition 7
A **range concatenation grammar** is a tuple $G = \langle N, T, V, S, P \rangle$ such that

- N is a finite set of predicates, each with a fixed arity;
- T and V are disjoint finite sets of terminals and of variables;
- $S \in N$ is the start predicate, a predicate of arity 1;
- P is a finite set *clauses* of the form $A_0(x_{01}, \ldots, x_{0a_0}) \rightarrow \epsilon$, or $A_0(x_{01}, \ldots, x_{0a_0}) \rightarrow A_1(x_{11}, \ldots, x_{1a_1}) \ldots A_n(x_{n1}, \ldots, x_{na_n})$, with $n \geq 1$ and $A_i \in N, x_{ij} \in (T \cup V)^*$ and a_i being the arity of A_i.

14 Since throughout the article, we use only positive RCGs; whenever we say "RCG," we actually mean "positive RCG."

When applying a clause with respect to a string $w = t_1 \cdots t_n$, the arguments of the predicates in the clause are instantiated with substrings of w, more precisely, with the corresponding ranges. A range $\langle i, j \rangle$ with $0 \leq i < j \leq n$ corresponds to the substring between positions i and j, that is, to the substring $t_{i+1} \cdots t_j$. If $i = j$, then $\langle i, j \rangle$ corresponds to the empty string ϵ. If $i > j$, then $\langle i, j \rangle$ is undefined.

Definition 8
For a given clause, an **instantiation** with respect to a string $w = t_1 \ldots t_n$ consists of a function $f : \{t' \mid t' \text{ is an occurrence of some } t \in T \text{ in the clause}\} \cup V \rightarrow \{\langle i, j \rangle \mid i \leq j, i, j \in \mathbb{N}\}$ such that

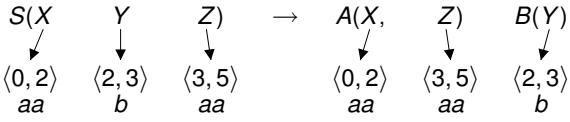
- for all occurrences t' of a $t \in T$ in the clause: $f(t') := \langle i, i+1 \rangle$ for some $i, 0 \leq i < n$, such that $t_i = t$;
- for all $v \in V$: $f(v) = \langle j, k \rangle$ for some $0 \leq j \leq k \leq n$;
- if consecutive variables and occurrences of terminals in an argument in the clause are mapped to $\langle i_1, j_1 \rangle, \ldots, \langle i_k, j_k \rangle$ for some k, then $j_m = i_{m+1}$ for $1 \leq m < k$. By definition, we then state that f maps the whole argument to $\langle i_1, j_k \rangle$.

The derivation relation is defined as follows. For a predicate A of arity k, a clause $A(\ldots) \rightarrow \ldots$, and ranges $\langle i_1, j_1 \rangle, \ldots, \langle i_k, j_k \rangle$ with respect to a given w: If there is an instantiation of this clause with left-hand side $A(\langle i_1, j_1 \rangle, \ldots, \langle i_k, j_k \rangle)$, then $A(\langle i_1, j_1 \rangle, \ldots, \langle i_i, j_k \rangle)$ can be replaced with the right-hand side of this instantiation.

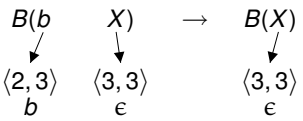
The language of an RCG G is the set of strings that can be reduced to the empty word, that is, $\{w \mid S(\langle 0, |w| \rangle) \stackrel{*}{\Rightarrow} \epsilon \text{ with respect to } w\}$.[15] An RCG with maximal predicate arity n is called an RCG of **arity *n***.

For illustration, let us consider a sample RCG: The RCG with $N = \{S, A, B\}$, $T = \{a, b\}$, $V = \{X, Y, Z\}$, start predicate S, and clauses $S(X\,Y\,Z) \rightarrow A(X, Z)\,B(Y)$, $A(a\,X, a\,Y) \rightarrow A(X, Y)$, $B(b\,X) \rightarrow B(X)$, $A(\epsilon, \epsilon) \rightarrow \epsilon, B(\epsilon) \rightarrow \epsilon$ has the string language $\{a^n b^k a^n \mid k, n \in \mathbb{N}\}$. Consider the reduction of $w = aabaa$:

We start from $S(\langle 0, 5 \rangle)$. First we can apply the following clause instantiation:

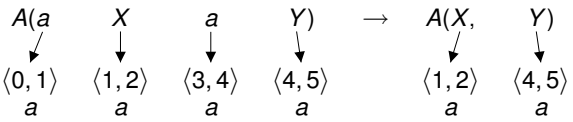


With this instantiation, $S(\langle 0, 5 \rangle) \Rightarrow A(\langle 0, 2 \rangle, \langle 3, 5 \rangle) B(\langle 2, 3 \rangle)$. Then

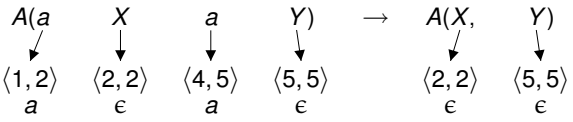


15 $|w|$ is the length of the word w; that is, the range $\langle 0, |w| \rangle$ with respect to w corresponds to the whole word w.

and $B(\epsilon) \rightarrow \epsilon$ lead to $A(\langle 0,2\rangle, \langle 3,5\rangle)B(\langle 2,3\rangle) \Rightarrow A(\langle 0,2\rangle, \langle 3,5\rangle)B(\langle 3,3\rangle) \Rightarrow A(\langle 0,2\rangle, \langle 3,5\rangle)$. Next,



leads to $A(\langle 0,2\rangle, \langle 3,5\rangle) \Rightarrow A(\langle 1,2\rangle, \langle 4,5\rangle)$. Then



and $A(\epsilon, \epsilon) \rightarrow \epsilon$ lead to $A(\langle 1,2\rangle, \langle 4,5\rangle) \Rightarrow A(\langle 2,2\rangle, \langle 5,5\rangle) \Rightarrow \epsilon$.

An RCG is said to be **noncombinatorial** if each of the arguments in the right-hand sides of the clauses are single variables. It is said to be **linear** if no variable appears more than once in the left-hand sides of the clauses and no variable appears more than once in the right-hand side of the clauses. It is said to be **nonerasing** if for each clause, each variable occurring in the left-hand side occurs also in the right-hand side and vice versa. It is said to be **simple** if it is noncombinatorial, linear, and nonerasing.

Simple RCGs and LCFRSs are equivalent (Boullier 1998b).

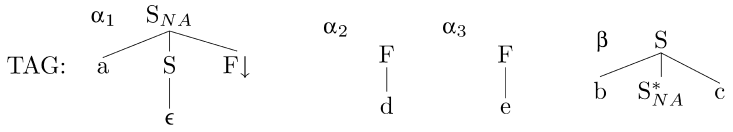
4.2 Relation between RSN-MCTAG and Simple RCG

The goal of this section is to construct an equivalent simple RCG for a given RSN-MCTAG. In order to be able to perform this construction, in the following we further constrain the formalism of RSN-MCTAG by defining RSN-MCTAG of a specific arity n. For this version of RSN-MCTAG, the construction of an equivalent simple RCG is possible.

First, let us sketch the general idea of the transformation from TAG to RCG (see Boullier 1998a). The RCG contains predicates $\langle\alpha\rangle(X)$ and $\langle\beta\rangle(L,R)$ for initial and auxiliary trees, respectively. X covers the yield of α and all trees added to α, and L and R cover those parts of the yield of β (including all trees added to β) that are to the left and the right of the foot node of β. The clauses in the RCG reduce the argument(s) of these predicates by identifying those parts that come from the elementary tree α/β itself and those parts that come from one of the elementary trees added by substitution or adjunction. A sample TAG with an equivalent RCG is shown in Figure 13.

For the construction of an equivalent RCG from a given RSN-MCTAG, we follow the same ideas while considering a secondary adjunction of β at some γ as adjunction at γ and not as adjunction at the elementary tree that is the mother node of β in the TAG derivation tree. There are two main differences between RSN-MCTAG and TAG that influence the construction of an equivalent RCG.

First, more than one tree can be added to a node. Therefore we allow predicates of the form $\langle\alpha\beta_1 \ldots \beta_k\rangle$ and $\langle\beta_0\beta_1 \ldots \beta_k\rangle$. The first means that at the node in question, first α was added by substitution, and then $\beta_1 \ldots \beta_k$ (in this order) were secondarily adjoined. The second means that at the node in question (an internal node), first β_0 was primarily adjoined, and then $\beta_1 \ldots \beta_k$ were secondarily adjoined. Since the number of secondary adjunctions at a node is limited by some constant depending on the grammar



Equivalent RCG:
$S(X) \rightarrow \langle\alpha_1\rangle(X) \mid \langle\alpha_2\rangle(X) \mid \langle\alpha_3\rangle(X)$,
$\langle\alpha_1\rangle(aF) \rightarrow \langle\alpha_2\rangle(F) \mid \langle\alpha_3\rangle(F)$ (no adjunction at internal S node)
$\langle\alpha_1\rangle(aB_1B_2F) \rightarrow \langle\beta\rangle(B_1,B_2)\langle\alpha_2\rangle(F) \mid \langle\beta\rangle(B_1,B_2)\langle\alpha_3\rangle(F)$ (adjunction at internal S node)
$\langle\beta\rangle(B_1b,cB_2) \rightarrow \langle\beta\rangle(B_1,B_2)$ (adjunction at root node)
$\langle\alpha_2\rangle(d) \rightarrow \epsilon \qquad \langle\alpha_3\rangle(e) \rightarrow \epsilon \qquad \langle\beta\rangle(b,c) \rightarrow \epsilon$ (no adjunction)

Figure 13
A sample TAG and an equivalent RCG.

(see Lemma 3), k is limited as well, and therefore this extension with respect to TAG adds only a finite number of predicates.

Second, the contribution of an elementary tree α/β including the trees added to it can be separated into arbitrarily many parts. Since each of the arguments of the predicates in the RCG has to cover a true substring of the input string, one needs predicates of arbitrary arities, namely, $\langle\alpha\ldots\rangle(L_n,\ldots,L_1,X,R_1,\ldots,R_n)$ and $\langle\beta\ldots\rangle(L_n,\ldots,L_1,L_0,R_0,R_1,\ldots,R_n)$, for the case where n auxiliary trees were added at the root of α/β that were actually secondarily adjoined at some higher tree such that these n trees separate the contribution of α/β into $2n+1$ / $2n+2$ parts, respectively. This extension is problematic, since it leads to an RCG with predicates of arbitrary arity: a **dynamic RCG** (Boullier 2001), a variant of RCG that is not polynomially parsable and that we therefore want to avoid. For this reason, we need an additional constraint on the RSN-MCTAGs we employ.

An example in which the contribution of an elementary tree is separated into three different parts is example (9), analyzed with the RSN-MCTAG in section 3.1 (see Figure 14). In the derived tree, the VP *das Fahrrad zu reparieren zu versuchen* (the broken triangles), which is the contribution of *versuchen*, is separated into three parts,

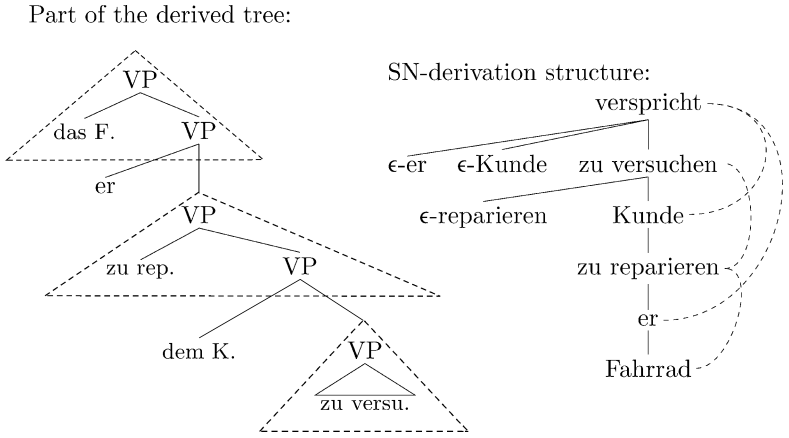


Figure 14
Analysis of example (9).

since *reparieren* secondarily adjoins at *versuchen* and *das Fahrrad* secondarily adjoins at *reparieren*.

(9) ...dass [das Fahrrad]$_1$ er [t_1 zu reparieren]$_2$ dem Kunden [t_2 zu versuchen]
...that the bike he to repair the customer to try

verspricht
promises

The crucial point in example (9) is that in the SN-derivation structure (see Figure 14), there are two crossings of secondary edges inside one group of secondary links. This means that the contribution of *versuchen* is interrupted twice by arguments of *verspricht* (by *Kunde* and *er*). In order to avoid predicates of arbitrary arity, we therefore limit the number of crossings of secondary links. We define the arity of an RSN-MCTAG depending on the maximal number of crossings that are allowed.

First, we define special subgraphs of the SN-derivation structure, **secondary groups**. These are subgraphs consisting of a chain of one primary substitution/adjunction and subsequent adjunctions at root or foot nodes such that there are secondary adjunctions along the whole chain. For example, the nodes *verspricht*, *zu versuchen*, *Kunde*, *zu reparieren*, *er*, and *Fahrrad* in the SN-derivation structure in Figure 14 form such a group. For an SN-derivation structure of a certain arity, the number of crossings of secondary edges inside a single secondary group is then limited: For an SN-derivation structure of arity n, the number of crossings of secondary edges per secondary group is limited to $\frac{n}{2} - 1$. In other words, if i is the maximal number of crossings, then $2(i+1)$ is the arity of the grammar. Of course, the arity is chosen such that an equivalent RCG of the same arity can be constructed. TAG, for example, is a grammar with 0 crossings, that is, an arity $2(0+1) = 2$ if the grammar is viewed as an SN-MCTAG, and the corresponding RCG is actually of arity 2.

Definition 9
Let $D_{SN} = \langle \mathcal{N}, \mathcal{E} \rangle$ be a SN-derivation structure.

1. $\langle \mathcal{N}', \mathcal{E}' \rangle$ is a secondary group in D_{SN} iff
   - $\mathcal{N}' = \{n_0, n_1, \ldots, n_k\} \subseteq \mathcal{N}$ for some $k > 1$ such that there are primary edges $\langle n_i, n_{i+1}, p_i \rangle$ for $0 \leq i < k$ with $p_i \in \mathbb{N}$;
   - $\mathcal{E}' \subseteq \mathcal{E}$ such that for all $n, n' \in \mathcal{N}, p \in \mathbb{N}$ with $\langle n, n', p \rangle \in \mathcal{E}$: if $n, n' \in \mathcal{N}'$, then $\langle n, n', p \rangle \in \mathcal{E}'$;
   - for all $i, 0 < i < k$, there are i_1, i_2 with $i_1 \leq i \leq i_2$, $i_1 \neq i_2$, such that $\langle n_{i_1}, n_{i_2}, p \rangle \in \mathcal{E}'$ is a secondary edge in D for some $p \in \mathbb{N}$.
2. D_{SN} is of arity n iff for each secondary group $\langle \mathcal{N}', \mathcal{E}' \rangle$ in D_{SN} with primary edges $\langle n_0, n_1, p_0 \rangle, \langle n_1, n_2, p_1 \rangle, \ldots, \langle n_{k-1}, n_k, p_{k-1} \rangle$ as above, there are at most $i \leq \frac{n}{2} - 1$ pairwise different sets of the form $\{j_0, j_1, j_2, j_3\}$ such that $j_0 < j_1 < j_2 < j_3$ and there are secondary edges $\langle n_{j_0}, n_{j_2}, p_1 \rangle$ and $\langle n_{j_1}, n_{j_3}, p_2 \rangle$ for some $p_1, p_2 \in \mathbb{N}$.

Definition 10
Let G be an MCTAG, $n \geq 1$. G is a **restricted tree-local MCTAG with shared nodes of arity *n*** iff the set of trees generated by G, $L_T(G)$, is defined as the set of those trees that can be derived in G with an RSN-tree-local multicomponent TAG derivation tree such that the corresponding SN-derivation structure is of arity n.

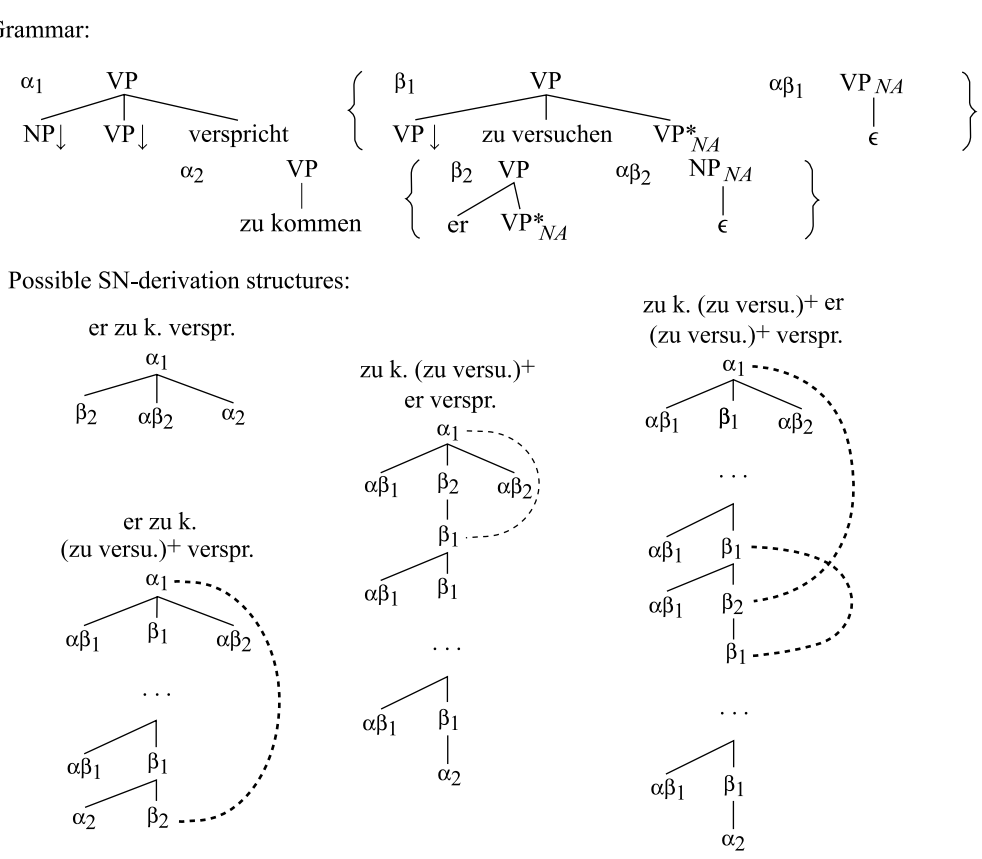


Figure 15
Sample RSN-MCTAG of arity four.

Consider a simple example of a construction of an equivalent RCG for a given RSN-MCTAG. We choose an RSN-MCTAG of arity four, and we see that the arity of the corresponding RCG is four as well. The RSN-MCTAG is shown in Figure 15. Whether this grammar is considered to be a general RSN-MCTAG or an RSN-MCTAG of arity four does not matter in this case, since even in the general case, all possible SN-derivation structures are of arity four. However, in the case of other RSN-MCTAGs, the restriction to a certain arity might exclude certain TAG derivation trees and thereby decrease the language generated by the grammar.

The language generated by the RSN-MCTAG in Figure 15 is {*er zu kommen (zu versuchen)* verspricht*, *zu kommen (zu versuchen)+ er (zu versuchen)* verspricht*}. The SN-derivation structures corresponding to the different strings are shown in Figure 15. The last one contains one crossing of secondary links; that is, the RSN-MCTAG is of arity four.

Now let us look at the corresponding RCG. Since the arity of the RSN-MCTAG is four, the predicates of the corresponding RCG are of arity three (for initial trees) and four (for auxiliary trees).

The contribution of α_1 is never separated into parts, therefore the first and the third arguments of the predicate $\langle\alpha_1\rangle$ are always ϵ. Looking at the SN-derivation structures in Figure 15, we have three different possibilities for $\langle\alpha_1\rangle$:

$$\langle\alpha_1\rangle(\epsilon, \textit{L N V verspricht } R, \epsilon) \rightarrow \langle\beta_2\rangle(\epsilon, L, R, \epsilon)\langle\alpha_{\beta_2}\rangle(\epsilon, N, \epsilon)\langle\alpha_2\rangle(\epsilon, V, \epsilon) \mid \langle\beta_1\beta_2\rangle(\epsilon, L, R, \epsilon)\langle\alpha_{\beta_2}\rangle(\epsilon, N, \epsilon)\langle\alpha_{\beta_1}\rangle(\epsilon, V, \epsilon) \mid \langle\beta_2\beta_1\rangle(\epsilon, L, R, \epsilon)\langle\alpha_{\beta_2}\rangle(\epsilon, N, \epsilon)\langle\alpha_{\beta_1}\rangle(\epsilon, V, \epsilon)$$

The interesting part of the grammar is the clauses for $\langle\beta_1\beta_2\rangle$, where two trees were added to the same node and further adjunctions at the root of β_1 are possible. The point is that the part covered by β_1 and the trees added to it can be separated into different substrings. This leads to

$$\langle\beta_1\beta_2\rangle(\epsilon, L_1\ L_2\ L_3, R_3\ R_2\ R_1, \epsilon) \rightarrow \langle\beta_1\rangle(L_1, L_3, R_3, R_1)\langle\beta_2\rangle(\epsilon, L_2, R_2, \epsilon)$$
$$\langle\beta_1\beta_2\rangle(L_1, L_2\ L_3, R_3\ R_2, R_1) \rightarrow \langle\beta_1\rangle(L_1, L_3, R_3, R_1)\langle\beta_2\rangle(\epsilon, L_2, R_2, \epsilon)$$
$$\langle\beta_1\beta_2\rangle(L_1\ L_2, L_3, R_3, R_2\ R_1) \rightarrow \langle\beta_1\rangle(L_1, L_3, R_3, R_1)\langle\beta_2\rangle(\epsilon, L_2, R_2, \epsilon)$$

Concerning $\langle\beta_2\beta_1\rangle$, the contribution of β_2 cannot be separated into different parts, since nothing can be adjoined to β_2. Consequently

$$\langle\beta_2\beta_1\rangle(\epsilon, L_1\ L_2, R_2\ R_1, \epsilon) \rightarrow \langle\beta_2\rangle(\epsilon, L_2, R_2, \epsilon)\langle\beta_1\rangle(\epsilon, L_1, R_1, \epsilon)$$
$$\langle\beta_2\beta_1\rangle(L_1, L_2, R_2, R_1) \rightarrow \langle\beta_2\rangle(\epsilon, L_2, R_2, \epsilon)\langle\beta_1\rangle(\epsilon, L_1, R_1, \epsilon)$$

Concerning $\langle\beta_1\rangle(L_1, L_3, R_3, R_1)$, either (L_1, R_1) cover something adjoined to β_1 (this can only be β_1), or (L_1, R_1) cover something adjoined to something ... adjoined to β_1. In this case, (L_3, R_3) cover β_1 and β_1 adjoined to its root, respectively. This leads to

$$\langle\beta_1\rangle(L_1, L_2\ \textit{V zu versuchen}, R_2, R_1) \rightarrow \langle\beta_1\rangle(L_1, L_2, R_2, R_1)\langle\alpha_{\beta_1}\rangle(\epsilon, V, \epsilon)$$

If the inner parts are empty, the outer parts are moved:

$$\langle\beta_1\rangle(L, \epsilon, \epsilon, R) \rightarrow \langle\beta_1\rangle(\epsilon, L, R, \epsilon)$$

Furthermore, there is a clause for $\langle\beta_1\rangle$ without adjunction at the root:

$$\langle\beta_1\rangle(\epsilon, \textit{V zu versuchen}, \epsilon, \epsilon) \rightarrow \langle\alpha_2\rangle(\epsilon, V, \epsilon)$$

For elementary trees where nothing needs to be added, simple ϵ-clauses are introduced:

$$\langle\alpha_2\rangle(\epsilon, \textit{ zu kommen}, \epsilon) \rightarrow \epsilon \quad \langle\alpha_{\beta_1}\rangle(\epsilon, \epsilon, \epsilon) \rightarrow \epsilon$$
$$\langle\beta_2\rangle(\epsilon, \textit{er}, \epsilon, \epsilon) \rightarrow \epsilon \quad \langle\alpha_{\beta_2}\rangle(\epsilon, \epsilon, \epsilon) \rightarrow \epsilon$$

In order to see how the RCG simulates the RSN-MCTAG, let us consider the derivation for *zu kommen zu versuchen er zu versuchen verspricht*:

$$\langle\alpha_1\rangle(\epsilon, \textit{zu kommen zu versuchen er zu versuchen verspricht}, \epsilon)$$
$$\Rightarrow \langle\beta_1\beta_2\rangle(\epsilon, \textit{zu kommen zu versuchen er zu versuchen}, \epsilon, \epsilon)\langle\alpha_{\beta_2}\rangle(\epsilon, \epsilon, \epsilon)\langle\alpha_{\beta_1}\rangle(\epsilon, \epsilon, \epsilon)$$
$$\stackrel{*}{\Rightarrow} \langle\beta_1\rangle(\textit{zu kommen zu versuchen},\ \textit{zu versuchen}, \epsilon, \epsilon)\langle\beta_2\rangle(\epsilon, \textit{er}, \epsilon, \epsilon)$$
$$\stackrel{*}{\Rightarrow} \langle\beta_1\rangle(\textit{zu kommen zu versuchen}, \epsilon, \epsilon, \epsilon)\langle\alpha_{\beta_1}\rangle(\epsilon, \epsilon, \epsilon)$$
$$\stackrel{*}{\Rightarrow} \langle\beta_1\rangle(\epsilon, \textit{zu kommen zu versuchen}, \epsilon, \epsilon)$$
$$\Rightarrow \langle\alpha_2\rangle(\epsilon, \textit{zu kommen}, \epsilon) \Rightarrow \epsilon$$

This example should give an idea of how an equivalent RCG for a given RSN-MCTAG of arity n can be constructed.

As already mentioned, in an RSN-MCTAG, the number of substitutions and (primary or secondary) adjunctions that can occur at each node is limited (see Lemma 3). Therefore, the number of predicates needed in the corresponding RCG is limited as well. Furthermore, in an RSN-MCTAG of arity n, the contribution of an elementary tree is separated into at most n parts. This still needs to be shown:

Lemma 4
Let G be an RSN-MCTAG of arity n. Then for all w in the string language of G and for all elementary trees γ used to derive w in G, the contribution of γ, that is, the yield of γ and everything added to γ, is separated into at most n parts.

The proof is given in the appendix.

Theorem 1
For each RSN-MCTAG G of arity n, a simple RCG G' of arity n can be constructed such that $L(G) = L(G')$.

The construction algorithm and a sketch of the proof are presented in the appendix. As a consequence of this theorem, the following corollary holds:

Corollary
For a given n, the string languages generated by RSN-MCTAGs of arity n are mildly context-sensitive, and they are in particular polynomially parsable.

Since we have shown that for RSN-MCTAG with a fixed arity, one obtains grammars that are LCFRSs, we know that we can even construct a weakly equivalent set-local MCTAG. This set-local MCTAG, however, does not present an alternative to the RSN-MCTAG with fixed arity: It is very large, containing a large number of elementary trees per tree set (the number depends on the arity of the grammar) and, furthermore, a large number of trees without lexical material and a large number of internal nodes that are needed only to provide adjunction sites.

An example is the set-local MCTAG in Figure 16. It is weakly equivalent to the RSN-MCTAG of arity four in Figure 15, and it even gives the correct dependency structure. The *verspricht* tree contains several VP nodes that are needed in order to provide adjunction sites for the different parts of *er* and *versuchen*. The *versuchen* tree set needs an extra auxiliary tree that provides an additional VP node for adjunction and has to be separated from the tree containing *versuchen*, since the contribution of *versuchen* might be separated into different parts. Of course this little grammar is still simple, since there are almost no possibilities of adjoining different trees at the same node or of separating the contribution of one lexical item into different parts.

As we have seen in Lemma 4, the linguistic signification of restricting the arity of the grammar to some n is that the lexical material containing a verb, all its arguments (including arguments and adjuncts of these arguments, etc.), and all its adjuncts cannot be separated into more than n discontinuous substrings in the whole sentence. For example, an RSN-MCTAG of arity two with elementary tree sets similar to those proposed above for scrambling would not be able to analyze example (9). However, RSN-MCTAGs of arity n for some sufficiently large fixed n can perhaps even describe

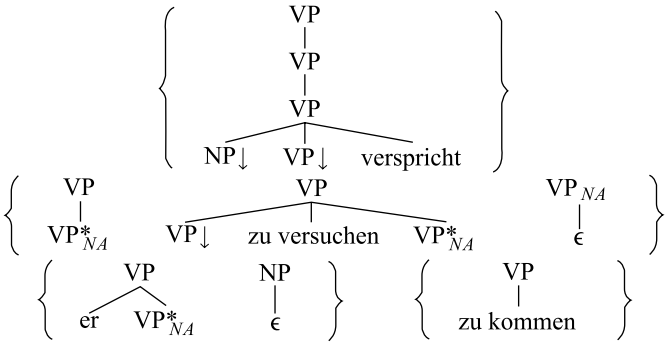


Figure 16
Equivalent set-local MCTAG for the RSN-MCTAG from Figure 15.

all cases of scrambling: See again the analysis of example (9) in Figure 14. Here, the contribution of *versuchen* and its arguments is split only by other elements secondarily adjoined to *verspricht*. If only a limited number of such secondary adjunctions were possible (this is the case), and if none of these other secondarily adjoined elements allowed for further secondary adjunctions at its root or foot node (this still needs to be investigated), then the number of crossings might be limited. We leave this issue for further research.

Even if RSN-MCTAG with a fixed arity could not analyze all scrambling data, based on empirical studies, n could be chosen sufficiently great such that the grammar would cover all scrambling cases that one assumes to occur.[16] The important point is that the complexity limit given by the fixed n is variable; that is, an arbitrary n can be chosen. This is different from TAG, for example, in which the limit is fixed (assuming, of course, that we desire only analyses respecting the CETM). In this sense one can say that RSN-MCTAG can analyze scrambling in general.

5. Conclusion and Future Work

This article addresses the problem of scrambling in tree-adjoining grammar, a formalism known not to be powerful enough to treat scrambling phenomena. In order to keep the advantages of TAG while being able to analyze scrambling, a local TAG variant is proposed that is based on the notion of node sharing, so-called (restricted) tree-local multicomponent TAG with node sharing. RSN-MCTAG is a true extension of TAG in the sense that the formalism can generate all tree-adjoining languages. The analysis of some German scrambling data is sketched in order to show that this TAG extension can treat scrambling.

Then, RSN-MCTAGs of specific arities are defined, and it is shown that for each RSN-MCTAG of a fixed arity n, an equivalent simple RCG of arity n can be constructed. Simple RCGs are mildly context-sensitive and in particular polynomially parsable and therefore, this also holds for RSN-MCTAGs of a fixed arity. RSN-MCTAGs of arity n perhaps cannot analyze all scrambling phenomena but, if the n is appropriately chosen, it can analyze an arbitrarily large set.

16 Joshi, Becker, and Rambow (2000) even argue that there might be a competence limit regarding the complexity of scrambling data. However, we do not discuss this issue here.

The scrambling data analyzed in section 3 present just a small part of the possible scrambling configurations. As already noted, this article does not present an exhaustive treatment of the phenomenon. Even though the examples we looked at indicate that RSN-MCTAGs are able to deal with scrambling, an exhaustive analysis of a larger amount of data still needs to be done, in particular, of scrambling in combination with other "movements" that cause word order variations, such as topicalization or extraposition. A first proposal in this direction can be found in Kallmeyer and Yoon (2004), but this proposal does not cover all phenomena one needs to take into account. So this is still an important issue for further research.

A formal issue one would like to see investigated more in detail is the relations between the different types of MCTAG. We have shown that the languages of RSN-MCTAG with fixed arity are in the class of set-local MCTALs. Furthermore, general SN-MCTAG can generate languages that cannot be generated by set-local MCTAG. However, this leaves open many interesting questions concerning the relations between set-local MCTAG and non-local MCTAG and the different formalisms defined in this article, namely, RSN-MCTAG of fixed arity and RSN-MCTAG and MCTAG with SN-tree-local and SN-set-local derivations. We plan to address these questions in the future.

Appendix: Proofs

Proof of Lemma 1

Let $G = \langle I, A, N, T, \mathcal{A}\rangle$ be an MCTAG, $G_{TAG} := \langle I, A, N, T\rangle$. Let $D = \langle \mathcal{N}, \mathcal{E}\rangle$ be a derivation tree in G_{TAG} with corresponding derived tree $t \in L(G_{TAG})$.

1. First show $\Rightarrow$ of the iff: Let D be a TAG derivation tree with $t \in L(G)$. It is immediate that the root of D is an instance of an initial tree and that all other nodes are elements of instances of elementary tree sets.

Assume that

- either there is an instance Γ of elementary tree sets from $\mathcal{A}$ such that there are $\gamma_1, \gamma_2 \in \Gamma$, with $\gamma_1 \in \mathcal{N}$ and $\gamma_2 \notin \mathcal{N}$, and γ_1 is not the root of D. $\Rightarrow$ it is not possible that Γ has been used in one of the multicomponent derivation steps in the course of the derivation of t. Contradiction.
- or there is an instance Γ of an elementary tree set such that there are $\gamma_1, \gamma_2 \in \Gamma$, $\gamma_1 \neq \gamma_2$, with $\langle \gamma_1, \gamma_2\rangle \in \mathcal{D}_D \Rightarrow \gamma_2$ has been added to a tree derived from γ_1. Contradiction to condition that all elements of Γ must have been added simultaneously.
- or there are pairwise different instances $\Gamma_1, \Gamma_2, \ldots, \Gamma_n$ of elementary tree sets from $\mathcal{A}$ such that there are $\gamma_1^{(i)}, \gamma_2^{(i)} \in \Gamma_i$, $1 \geq i \geq n$, with $\langle \gamma_1^{(1)}, \gamma_2^{(n)}\rangle \in \mathcal{D}_D$ and $\langle \gamma_1^{(i)}, \gamma_2^{(i-1)}\rangle \in \mathcal{D}_D$ for $2 \geq i \geq n$. $\Rightarrow \gamma_1^{(i)}$ was added before $\gamma_2^{(i-1)}$ for $2 \leq i \leq n$, and since all elements from Γ_i must be added simutaneously for $1 \leq i \leq n$, Γ_n was added before Γ_1. $\Rightarrow \langle \gamma_1^{(1)}, \gamma_2^{(n)}\rangle \notin \mathcal{D}_{\mathcal{D}}$. Contradiction.

Consequently, D satisfies (MC1)–(MC3).

2. Then show $\Leftarrow$ of the iff: Let D be a derivation tree in G_{TAG} satisfying (MC1)–(MC3).

There are different orderings of the derivation steps in D possible: Let the node positions on the derived tree be pairs $\langle \gamma, p \rangle$, with γ being an instance of an elementary tree and p being a position in γ. Every top-down order read off D (no matter whether [partly] depth first or not and whether left to right or right to left) is a possible derivation order in G_{TAG} for the derivation tree D, since in order to perform the derivation step $\ldots[\langle \gamma_1, p \rangle, \gamma_2]$ corresponding to an edge $\langle \gamma_1, \gamma_2, p \rangle$ in D, one needs only to ensure that γ_1 (i.e., the mother node of γ_2) has already been added.

Because of (MC1), the root of D is an initial tree, and the set of all other nodes in D can be partitioned into pairwise different instances of elementary tree sets.

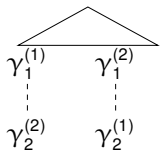
To show: There is a top-down traversal of D such that the traversal starts with an initial tree and then there is always one instance Γ of an elementary set whose members are visited next in any order (i.e., simultaneously).

The top-down traversal has to start with the root node (i.e., an initial tree α). Assume that at some point of the traversal, the choice of a new instance of an elementary set to be visited next is not possible. $\Rightarrow$ for each set Γ that has not been visited yet, there is at least one $\gamma \in \Gamma$ whose mother node has not been visited yet (otherwise Γ could be visited next).

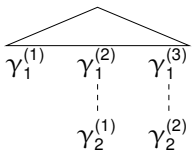
Pick an unvisited Γ_1 with at least one $\gamma_1^{(1)} \in \Gamma_1$ whose mother node has been visited. Assume $\gamma_2^{(1)} \in \Gamma_1$ with mother not yet visited. Suppose $\gamma_1^{(2)}$ to be the highest unvisited node dominating $\gamma_2^{(1)}$. Since $\gamma_1^{(2)} \neq \gamma_2^{(1)}$ and $\langle \gamma_1^{(2)}, \gamma_2^{(1)} \rangle \in \mathcal{D}_{\mathcal{D}}$, (with (MC2)) $\gamma_1^{(2)} \in \Gamma_2 \neq \Gamma_1$.

Then there is a $\gamma_2^{(2)} \in \Gamma_2$ with unvisited mother such that

(a) either $\langle \gamma_1^{(1)}, \gamma_2^{(2)} \rangle \in \mathcal{D}_{\mathcal{D}}$. Contradiction to (MC3) with $n = 2$.



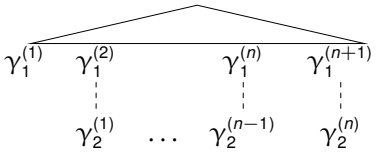
(b) or $\langle \gamma_1^{(1)}, \gamma_2^{(2)} \rangle \notin \mathcal{D}_{\mathcal{D}}$. Because of (MC2), $\langle \gamma_1^{(2)}, \gamma_2^{(2)} \rangle \notin \mathcal{D}_{\mathcal{D}}$. Let $\gamma_1^{(3)} \in \Gamma_3$ be the highest unvisited node dominating $\gamma_2^{(2)}$. Because of (MC2), $\Gamma_3 \neq \Gamma_2$, and because of (MC3), $\Gamma_3 \neq \Gamma_1$.



In the (b) case, there is a $\gamma_2^{(3)} \in \Gamma_3$ with unvisited mother node. Because of (MC2) and (MC3), $\langle \gamma_1^{(1)}, \gamma_2^{(3)} \rangle \notin \mathcal{D}_{\mathcal{D}}$, $\langle \gamma_1^{(2)}, \gamma_2^{(3)} \rangle \notin \mathcal{D}_{\mathcal{D}}$, and $\langle \gamma_1^{(3)}, \gamma_2^{(3)} \rangle \notin \mathcal{D}_{\mathcal{D}}$. Then there is a highest unvisited node $\gamma_1^{(4)} \in \Gamma_4$ dominating $\gamma_2^{(3)}$ with $\Gamma_4 \neq \Gamma_3, \Gamma_4 \neq \Gamma_2$, and $\Gamma_4 \neq \Gamma_1$. And there is a $\gamma_2^{(4)} \in \Gamma_4$ with unvisited mother node.

In general, for each of the Γ_n, $1 \leq n$, with $\gamma_1^{(n)}, \gamma_2^{(n)}$ as above, the situation is as follows: $\Gamma_n \neq \Gamma_i$ for $1 \leq i < n$ (otherwise contradiction to (MC2) or (MC3)) and $\langle \gamma_1^{(i)}, \gamma_2^{(n)} \rangle \notin \mathcal{D}_{\mathcal{D}}$ for $i \leq n$ (otherwise contradiction to (MC2) for $i = n$ or to (MC3)

for $i \neq n$). Consequently, there is always a new Γ_{n+1}, with a new $\gamma_1^{(n+1)}$ being the highest unvisited node dominating $\gamma_2^{(n)}$ and $\gamma_2^{(n+1)}$ being a node with unvisited mother.



Contradiction to the finiteness of the number of nodes in D. $\Rightarrow$ there is a top-down traversal of D that corresponds to a multicomponent derivation in G in the sense that it allows us to visit the instances of elementary tree sets one after the other. ■

Proof of Lemma 2

Only the uniqueness needs to be shown.

Let $G = \langle I, A, N, T, \mathcal{A} \rangle$ be an RSN-MCTAG. Let $D = \langle \mathcal{N}, \mathcal{E} \rangle$ be a TAG derivation tree in G.

Assume that there is an instance $\{\gamma_1, \ldots, \gamma_n\}$ of an elementary tree set such that there are γ, γ' with $\gamma \neq \gamma'$ and $\langle \gamma, \gamma_1 \rangle, \ldots, \langle \gamma, \gamma_n \rangle, \langle \gamma', \gamma_1 \rangle, \ldots, \langle \gamma', \gamma_n \rangle \in \mathcal{SN}_D$ and there are $i, j, 1 \leq i, j \leq n$ with $\langle \gamma, \gamma_i \rangle, \langle \gamma', \gamma_j \rangle \in \mathcal{P}_D$.

$\Rightarrow$ since $\langle \gamma', \gamma_j \rangle \in \mathcal{P}_D$ and $\langle \gamma, \gamma_j \rangle \in \mathcal{D}_D$ with $\gamma \neq \gamma_j$, there is a γ'' with $\langle \gamma, \gamma'' \rangle \in \mathcal{P}_D$ and $\langle \gamma'', \gamma' \rangle \in \mathcal{D}_D$. Furthermore, $\langle \gamma_i, \gamma'' \rangle \notin \mathcal{D}_D$ (otherwise $\langle \gamma_i, \gamma_j \rangle \in \mathcal{D}_D$ which would contradict (MC2)) and $\langle \gamma'', \gamma_i \rangle \notin \mathcal{D}_D$ (otherwise, since $\gamma_i \neq \gamma''$, $\langle \gamma, \gamma_i \rangle \notin \mathcal{P}_D$). Consequently $\langle \gamma', \gamma_i \rangle \notin \mathcal{D}_D$. Contradiction to assumption. ■

Proof of Lemma 4

Let G be an RSN-MCTAG of arity n, $w \in L(G)$, such that the elementary tree γ was used to derive w. Assume that the contribution of γ is separated into $m > n$ parts.

Then the SN-derivation structure for this derivation is as shown in Figure 17.

Consequently, there are at least $\lceil \frac{m}{2} - 1 \rceil$ crossings, and the arity of G is $(\lceil \frac{m}{2} - 1 \rceil + 1) \cdot 2$. $\Rightarrow$ if m is even, G is of arity m, and if m is odd, G is of arity $m + 1$. This is a contradiction to the assumption that the arity of G is $n < m$. ■

Proof of Theorem 1

For reasons of space, we do not give the whole proof of the theorem but restrict ourselves to the construction algorithm and a rough outline of the rest of the proof.

Construction algorithm. Let G be an RSN-MCTAG of arity n.

Construction of a weakly equivalent RCG G':

The terminals and nonterminals will be implicitly defined by the clauses of the grammar.

Predicates: Let k_1 be the maximal number of nodes in an elementary tree in G and k_2 be the maximal number of trees in an elementary tree set. $k := k_1(k_2 - 1)$.

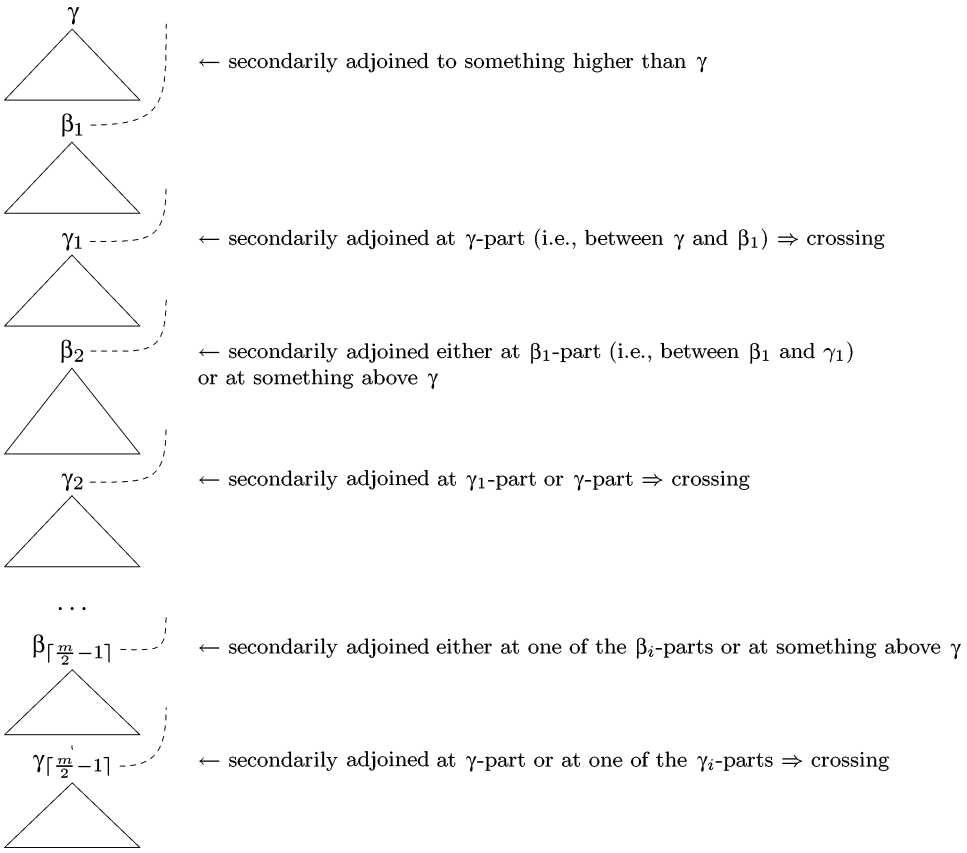


Figure 17
SN-derivation structure for proof of Lemma 4.

- There is a unary predicate S.
- Each $\langle\alpha\rangle$ and each $\langle\alpha\beta_1\cdots\beta_l\rangle$, with $l < k$, α an initial tree, and $\beta_1, \ldots, \beta_l$ auxiliary trees is an $n-1$-ary predicate.
- Each $\langle\beta\rangle$ and each $\langle\beta\beta_1\cdots\beta_l\rangle$ with $l < k$ and $\beta, \beta_1, \ldots, \beta_l$ auxiliary trees is an n-ary predicate.

Define the decoration string σ_γ of an elementary tree γ as in Boullier (1999), except that a root node μ has n variables, $L_{\mu 1} \ldots L_{\mu \frac{n}{2}}$ on the left and $R_{\mu \frac{n}{2}} \ldots R_{\mu 1}$ on the right. Every other internal node μ has two variables L_μ and R_μ, and each substitution node has one variable X. In a top-down, left-to-right traversal, the left variables are collected during the top-down traversal, the terminals and variables of substitution nodes are collected while visiting the leaves, and the right variables are collected during bottom-up traversal.

Construction of the clauses:

In the following, $P(\epsilon, \ldots, \epsilon, x, \epsilon, \ldots, \epsilon)$ signifies that x is the $\frac{n}{2}$th argument, and $P(\epsilon, \ldots, \epsilon, x_1, x_2, \epsilon, \ldots, \epsilon)$ signifies that x_1 is the $\frac{n}{2}$th and x_2 the $(\frac{n}{2}+1)$th argument.

(1) **Predicate** S
For each initial α, there is a clause

$$S(X) \rightarrow \langle\alpha\rangle(\epsilon, \ldots, \epsilon, X, \epsilon, \ldots, \epsilon)$$

(2) **Predicates** $\langle\gamma\rangle$
For each elementary γ: $lhs := \sigma_\gamma$, $rhs := \epsilon$.

For each combination of substitutions and adjunctions at γ, with substitutions at all substitution nodes and adjunctions at all internal nodes, with obligatory adjunction that respects the conditions for restricted tree-local multicomponent derivation:

For all nodes μ in γ:

- If μ is an internal node that is not the root, and no adjunction takes place at μ, then delete L_μ and R_μ in lhs.
- If μ is the root node and no adjunction takes place at μ, then delete $L_{\mu_1} \cdots L_{\mu_{\frac{n}{2}}}$ and $R_{\mu_{\frac{n}{2}}} \cdots R_{\mu_1}$ in lhs.
- If only the initial tree α is substituted at μ, $rhs := \langle\alpha\rangle(\epsilon, \ldots, \epsilon, S_\mu, \epsilon, \ldots, \epsilon)rhs$.
- If α is substituted at μ and then $\beta_1, \ldots, \beta_m$ (in that order) are secondarily adjoined at μ, then $rhs := \langle\alpha\beta_1 \cdots \beta_m\rangle(\epsilon, \ldots, \epsilon, S_\mu, \epsilon, \ldots, \epsilon)rhs$.
- If μ is an internal node that is not the root and $\beta_1, \ldots, \beta_m$ are adjoined (in that order) at μ, then $rhs := \langle\beta_1 \cdots \beta_m\rangle(\epsilon, \ldots, \epsilon, L_\mu, R_\mu, \epsilon, \ldots, \epsilon)rhs$.

If there is no adjunction at the root of γ, there is a clause

$$\langle\gamma\rangle(\epsilon, \ldots, \epsilon, lhs, \epsilon, \ldots, \epsilon) \rightarrow rhs$$

If $\beta_1, \ldots, \beta_m$ are adjoined in this order at the root of γ, and if $L_1, \ldots, L_{\frac{n}{2}}, R_{\frac{n}{2}}, \ldots, R_1$ are the parts of the root in lhs such that $lhs = L_1 \cdots L_{\frac{n}{2}} lhs' R_{\frac{n}{2}} \cdots R_1$, then there is a clause

$$\langle\gamma\rangle(L_1, \ldots, L_{\frac{n}{2}-1}, L_{\frac{n}{2}} lhs' R_{\frac{n}{2}}, R_{\frac{n}{2}-1}, \ldots, R_1) \rightarrow \langle\beta_1 \cdots \beta_m\rangle(L_1, \ldots, L_{\frac{n}{2}}, R_{\frac{n}{2}}, \ldots, R_1)rhs$$

Further, for each initial γ and for all i, $1 \leq i \leq \frac{n}{2} - 2$, there are clauses

$$\langle\gamma\rangle(L_1, \ldots, L_i, \epsilon, L_{i+1}, \ldots, L_{\frac{n}{2}-2}, X, R_{\frac{n}{2}-2}, \ldots, R_{i+1}, \epsilon, R_i, \ldots, R_1) \rightarrow \langle\gamma\rangle(\epsilon, L_1, \ldots, L_{\frac{n}{2}-2}, X, R_{\frac{n}{2}-2}, \ldots, R_1, \epsilon)$$

And for each auxiliary γ and for all i, $1 \leq i \leq \frac{n}{2} - 1$, there are clauses

$$\langle\gamma\rangle(L_1, \ldots, L_i, \epsilon, L_{i+1}, \ldots, L_{\frac{n}{2}-1}, R_{\frac{n}{2}-1}, \ldots, R_{i+1}, \epsilon, R_i, \ldots, R_1) \rightarrow \langle\gamma\rangle(\epsilon, L_1, \ldots, L_{\frac{n}{2}-1}, R_{\frac{n}{2}-1}, \ldots, R_1, \epsilon)$$

(3) **Predicates** $\langle \gamma_1 \gamma_2 \cdots \gamma_m \rangle$
For each $\langle \gamma_1 \gamma_2 \cdots \gamma_m \rangle$ with $m \geq 2$ occurring in the clauses constructed so far:
Define sets of variables

$\mathcal{L} := \{L_1(\gamma_1), \ldots, L_{\frac{n}{2}}(\gamma_1), L_1(\gamma_2), \ldots, L_{\frac{n}{2}}(\gamma_m)\}$, and
$\mathcal{R} := \{R_1(\gamma_1), \ldots, R_{\frac{n}{2}}(\gamma_1), R_1(\gamma_2), \ldots, R_{\frac{n}{2}}(\gamma_m)\}$, and
three other pairwise different variables $X_1, X_2, X \notin \mathcal{L} \cup \mathcal{R}$.

Define for all $x \in \mathcal{L}^* : R(x) := y \in \mathcal{R}^*$ such that if $L_j(\gamma_i)$ is the kth letter of x, then $R_j(\gamma_i)$ is the $(|x| - k + 1)$th letter of y.
For all $w \in \mathcal{L}^*$ such that

(a) each $L \in \mathcal{L}$ occurs exactly once in w,

(b) for all $1 \leq i_1 < i_2 \leq m$, $L_{\frac{n}{2}}(\gamma_{i_1})$ is to the right of $L_{\frac{n}{2}}(\gamma_{i_2})$ in w, and

(c) for all $1 \leq i \leq m$ and $1 \leq j_1 < j_2 \leq \frac{n}{2}$, $L_{j_1}(\gamma_i)$ is to the left of $L_{j_2}(\gamma_i)$ in w,

and for all $x_1, \ldots, x_{\frac{n}{2}} \in \mathcal{L}^*$ with $x_1 \cdots x_{\frac{n}{2}} = w$,
there is the following clause:
If γ_1 is an initial tree, a clause with $L_{\frac{n}{2}}(\gamma_1)$ eliminated from the $x_1, \ldots, x_{\frac{n}{2}}$:

$$\begin{aligned}
&\langle \gamma_1 \cdots \gamma_m \rangle (x_1, \ldots, x_{\frac{n}{2}}\, X\, R(x_{\frac{n}{2}}), \ldots, R(x_1)) \rightarrow \\
&\quad \langle \gamma_1 \rangle (L_1(\gamma_1), \ldots, L_{\frac{n}{2}-1}(\gamma_1), X, R_{\frac{n}{2}-1}(\gamma_1), \ldots, R_1(\gamma_1)) \\
&\quad \langle \gamma_2 \rangle (L_1(\gamma_2), \ldots, L_{\frac{n}{2}}(\gamma_2), R_{\frac{n}{2}}(\gamma_2), \ldots, R_1(\gamma_2)) \\
&\quad \vdots \\
&\quad \langle \gamma_m \rangle (L_1(\gamma_m), \ldots, L_{\frac{n}{2}}(\gamma_m), R_{\frac{n}{2}}(\gamma_m), \ldots, R_1(\gamma_m))
\end{aligned}$$

If γ_1 is an auxiliary tree, a clause

$$\begin{aligned}
&\langle \gamma_1 \cdots \gamma_m \rangle (x_1, \ldots, x_{\frac{n}{2}}, R(x_{\frac{n}{2}}), \ldots, R(x_1)) \rightarrow \\
&\quad \langle \gamma_1 \rangle (L_1(\gamma_1), \ldots, L_{\frac{n}{2}}(\gamma_1), R_{\frac{n}{2}}(\gamma_1), \ldots, R_1(\gamma_1)) \\
&\quad \vdots \\
&\quad \langle \gamma_m \rangle (L_1(\gamma_m), \ldots, L_{\frac{n}{2}}(\gamma_m), R_{\frac{n}{2}}(\gamma_m), \ldots, R_1(\gamma_m))
\end{aligned}$$

(4) These are all clauses.

Sketch of Proof. We do not give the whole proof of the correctness of the construction, but we sketch the principal steps:

Mainly, two lemmas, concerning, respectively, the clauses constructed under paragraph 2 above and under paragraph 3 above, are shown:

Lemma 5
For each elementary tree γ in G with decoration string σ_γ as defined above:

There is a γ' derived from γ with yield w (if γ is an initial tree) or $\langle w_l, w_r \rangle$ (if γ is an auxiliary tree with w_l on the left and w_r on the right of the foot node) such that

- all leaves in γ' have terminal labels;
- there are no OA constraints in γ';

- for all node positions p in γ at which substitutions or adjunctions took place, the elementary trees $\gamma_1^{(p)}, \gamma_1^{(p)}, \ldots, \gamma_m^{(p)}$ (in that order) were substituted/adjoined at the node at position p in γ (these are all trees attached to this node).

iff

There is a $\langle\gamma\rangle$-clause in G' corresponding to the attachments to γ in this derivation in the way described in the construction, with σ' being the decoration string of γ without the symbols for the nodes to which nothing was attached such that

There is an instantiation $f : \{t' \mid t' \text{ is an occurrence of some } t \in T \text{ in } \sigma'\} \cup V \rightarrow \{\langle i,j\rangle \mid 0 \leq i \leq j \leq |w|\}$ as defined in Definition 8, and the following hold for f:

- For each substitution node μ in γ with position p, the yield of the trees $\gamma_1^{(p)}, \gamma_1^{(p)}, \ldots, \gamma_m^{(p)}$ and everything added to them is the connected substring $f(S_\mu)$, even if the derivation of γ' from γ is part of a larger derivation.
- For each internal node μ in γ with position p at which adjunctions took place, the yield of the trees $\gamma_1^{(p)}, \gamma_1^{(p)}, \ldots, \gamma_m^{(p)}$ consists of the two connected substrings $\langle f(L_\mu), f(R_\mu)\rangle$, even if the derivation of γ' from γ is part of a larger derivation.
- If adjunctions at the root took place, then the yield of the trees $\gamma_1^{(p)}, \gamma_1^{(p)}, \ldots, \gamma_m^{(p)}$ consists of the substrings $\langle f(L_1)f(L_2)\cdots f(L_{\frac{n}{2}}), f(R_{\frac{n}{2}})\cdots f(R_1)\rangle$, and this yield can be disconnected if the derivation of γ' from γ is part of a larger derivation; it can be separated into disconnected substrings $f(L_1), f(L_2), \ldots, f(L_{\frac{n}{2}}), f(R_{\frac{n}{2}}), \ldots, f(R_1)$.

Proof by induction on structure of γ.

Lemma 6

For all $\gamma_1, \gamma_2, \ldots, \gamma_m$:

There is a derivation in G of a tree with yield w in which $\gamma_1, \gamma_2, \ldots, \gamma_m$ (in that order) attach to some node μ in an elementary tree γ such that the yield of the trees $\gamma_1, \gamma_2, \ldots, \gamma_m$ is separated into at most n disconnected substrings (ranges) of w. If γ_1 is an auxiliary tree, it is separated into the substrings $l_1, \ldots, l_{\frac{n}{2}}, r_{\frac{n}{2}}, \ldots, r_1$; otherwise ($\gamma_1$ initial), the substrings are $l_1, \ldots, l_{\frac{n}{2}-1}, x, r_{\frac{n}{2}-1}, \ldots, r_1$.

iff

There is a $\langle\gamma_1\gamma_2\cdots\gamma_m\rangle$-clause as described in paragraph 3 of the construction above such that there is an instantiation f of the clause such that

- If γ_1 is initial, then $f(x_1) = l_1, \ldots, f(x_{\frac{n}{2}-1}) = l_{\frac{n}{2}-1}, f(x_{\frac{n}{2}} X R(x_{\frac{n}{2}})) = x, f(R(x_{\frac{n}{2}-1})) = r_{\frac{n}{2}-1}, \ldots, f(R(x_1)) = r_1$, and if γ_1 is auxiliary, then $f(x_1) = l_1, \ldots, f(x_{\frac{n}{2}}) = l_{\frac{n}{2}}1, f(R(x_{\frac{n}{2}})) = r_{\frac{n}{2}}, \ldots, f(R(x_1)) = r_1$.
- If γ_1 is initial, then its yield in w consists of the $n-1$ substrings (ranges) $f(L_1(\gamma_1)), \ldots, f(L_{\frac{n}{2}-1}(\gamma_1)), f(X), f(R_{\frac{n}{2}-1}(\gamma_1)), \ldots, f(R_1(\gamma_1))$.
- For all auxiliary γ_i, $1 \leq i \leq m$, the yield of γ_i in w consists of the n substrings (ranges) $f(L_1(\gamma_i)), \ldots, f(L_{\frac{n}{2}}(\gamma_i)), f(R_{\frac{n}{2}}(\gamma_i)), \ldots, f(R_1(\gamma_i))$.

Proof by induction on m.

The whole theorem can then be proven using these two lemmas.

Acknowledgments

For valuable suggestions, helpful comments and fruitful discussions of the subject of this article, we would like to thank Anne Abeillé, Pierre Boullier, David Chiang, Eric de la Clergerie, Chung-Hye Han, Aravind Joshi, Tony Kroch, Seth Kulick, Maribel Romero and SinWon Yoon. Furthermore, we are really grateful to three anonymous reviewers who gave many very helpful comments and whose suggestions for improvements influenced considerably the final form of the article.

References

Becker, Tilman, Aravind K. Joshi, and Owen Rambow. 1991. Long-distance scrambling and tree adjoining grammars. In *Proceedings of ACL-Europe*, Berlin.

Becker, Tilman, Owen Rambow, and Michael Niv. 1992. The derivational generative power of formal systems, or Scrambling is beyond LCFRS. Technical Report IRCS-92-38, Institute for Research in Cognitive Science, University of Pennsylvania.

Bertsch, Eberhard and Mark-Jan Nederhof. 2001. On the complexity of some extensions of RCG parsing. In *Proceedings of the Seventh International Workshop on Parsing Technologies*, pages 66–77, Beijing, October.

Boullier, Pierre. 1998a. A generalization of mildly context-sensitive formalisms. In *Proceedings of the Fourth International Workshop on Tree Adjoining Grammars and Related Formalisms (TAG+4)*, pages 17–20, University of Pennsylvania, Philadelphia.

Boullier, Pierre. 1998b. A proposal for a natural language processing syntactic backbone. Technical Report 3342, Institut National de Recherche en Informatique et en Automatique (INRIA) Rocquencourt.

Boullier, Pierre. 1999. On TAG parsing. In *Proceedings of TALN 99: Sixième Conférence Annuelle sur le Traitement Automatique des Langues Naturelles*, pages 75–84, Cargèse, Corse, July.

Boullier, Pierre. 2000. Range concatenation grammars. In *Proceedings of the Sixth International Workshop on Parsing Technologies (IWPT2000)*, pages 53–64, Trento, Italy, February.

Boullier, Pierre. 2001. From contextual grammars to range concatenation grammars. In *Proceedings of the Sixth Conference on Formal Grammar and Seventh Conference on Mathematics of Language (FG/MOL'01)*, Helsinki, August.

Candito, Marie-Hélène and Sylvain Kahane. 1998. Can the TAG derivation tree represent a semantic graph? An answer in the light of meaning-text theory. In *Proceedings of the Fourth International Workshop on Tree Adjoining Grammars and Related Formalisms,* IRCS Report 98–12, pages 25–28, University of Pennsylvania, Philadelphia.

Dras, Mark, David Chiang, and William Schuler. 2004. On relations of constituency and dependency grammars. *Research on Language and Computation*, 2(2):281–305.

Frank, Robert. 1992. *Syntactic Locality and Tree Adjoining Grammar: Grammatical, Acquisition and Processing Perspectives*. Ph.D. thesis, University of Pennsylvania.

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

Gerdes, Kim. 2002. *Topologie et grammaires formelles de l'allemand*. Ph.D. thesis, Université Paris 7.

Joshi, Aravind K. 1985. Tree adjoining grammars: How much context sensitivity is required ro provide reasonable structural descriptions? In D. Dowty, L. Karttunen, and A. Zwicky, editors, *Narural Language Parsing*. Cambridge University Press, Cambridge, pages 206–250.

Joshi, Aravind K. 1987. An introduction to tree adjoining grammars. In A. Manaster-Ramer, editor, *Mathematics of Language*. John Benjamins, Amsterdam, pages 87–114.

Joshi, Aravind K., Tilman Becker, and Owen Rambow. 2000. Complexity of scrambling: A new twist to the competence/performance distinction. In Anne Abeillé and Owen Rambow, editors, *Tree Adjoining Grammars: Formalisms, Linguistic Analyses and Processing*. Center for the Study of Language and Information (CSLI) Publications, Stanford, CA, pages 167–181.

Joshi, Aravind K., Leon S. Levy, and Masako Takahashi. 1975. Tree adjunct grammars. *Journal of Computer and System Science*, 10:136–163.

Joshi, Aravind K. and Yves Schabes. 1997. Tree-adjoining grammars. In G. Rozenberg and A. Salomaa, editors, *Handbook of Formal Languages*. Springer, Berlin, pages 69–123.

Joshi, Aravind K. and K. Vijay-Shanker. 1999. Compositional semantics with lexicalized tree-adjoining grammar (LTAG): How much underspecification is necessary? In H. C. Blunt and E. G. C. Thijsse, editors,

Proceedings of the Third International Workshop on Computational Semantics (IWCS-3), pages 131–145, Tilburg, The Netherlands.

Kallmeyer, Laura. 2001. Local tree description grammars: A local extension of TAG allowing underspecified dominance relations. *Grammars*, 4:85–137.

Kallmeyer, Laura. 2002. Using an enriched TAG derivation structure as basis for semantics. In *Proceedings of TAG+6 Workshop*, pages 127–136, Venice, May.

Kallmeyer, Laura and Aravind K. Joshi. 2003. Factoring predicate argument and scope semantics: Underspecified semantics with LTAG. *Research on Language and Computation*, 1(1–2):3–58.

Kallmeyer, Laura and Sinwon Yoon. 2004. Tree-local MCTAG with shared nodes: Word order variation in German and Korean. In *Proceedings of TAG+7*, Vancouver.

Kulick, Seth Norman. 2000. *Constraining Non-local Dependencies in Tree Adjoining Grammar: Computational and Linguistic Perspectives*. Ph.D. thesis, University of Pennsylvania.

Rambow, Owen. 1994a. *Formal and Computational Aspects of Natural Language Syntax*. Ph.D. thesis, University of Pennsylvania.

Rambow, Owen. 1994b. Multiset-valued linear index grammars: Imposing dominance constraints on derivations. In *Proceedings of ACL*, Las Cruces, NM.

Rambow, Owen and Young-Suk Lee. 1994. Word order variation and tree-adjoining grammars. *Computational Intelligence*, 10(4):386–400.

Rambow, Owen, K. Vijay-Shanker, and David Weir. 1995. D-tree grammars. In *Proceedings of ACL*, Cambridge, MA.

Rambow, Owen, K. Vijay-Shanker, and David Weir. 2001. D-tree substitution grammars. *Computational Linguistics*, 27(1):87–121.

Rogers, James. 1994. *Studies in the Logic of Trees with Applications to Grammar Formalisms*. Ph.D. thesis, University of Delaware.

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

Schabes, Yves and Stuart M. Shieber. 1994. An alternative conception of tree-adjoining derivation. *Computational Linguistics*, 20(1):91–124.

Vijay-Shanker, K. 1987. *A Study of Tree Adjoining Grammars*. Ph.D. thesis, University of Pennsylvania.

Vijay-Shanker, K. 1992. Using descriptions of trees in a tree adjoining grammar. *Computational Linguistics*, 18(4):481–517.

Vijay-Shanker, K. and Aravind K. Joshi. 1988. Feature structures based tree adjoining grammar. In *Proceedings of COLING*, pages 714–719, Budapest.

Weir, David J. 1988. *Characterizing mildly context-sensitive grammar formalisms*. Ph.D. thesis, University of Pennsylvania.

XTAG Research Group. 1998. A lexicalized tree adjoining grammar for English. Technical Report IRCS 98-18, Institute for Research in Cognitive Science, University of Pennsylvania.