Workshop TAG+5, Paris, 25-27 May 2000

A Logical Approach to Structure Sharing in TAGs

Adi Palm

Department of General Linguistics
University of Passau
D-94030 Passau

Abstract

Tree adjoining grammars (TAG) represent a derivational formalism to construct trees from a given set of initial and auxiliary trees. We present a logical language that simultaneously describes the generated TAG-tree and the corresponding derivation tree. Based on this language we formulate constraints indicating whether a tree and a derivation tree mean a valid TAG-generated tree. A method is presented that extracts the underlying TAG from an (underspecified) TAG-tree and its derivation. This leads to an alternative approach of representing shared structures by means of TAGs. The result is a more general representation of movement which requires no indices since it basically makes use of the properties of the adjunction operation.

1. Introduction

Recently, we find several approaches establishing a logical description of finite trees, e.g., first-order logic (Backofen *et al.*, 1995), dynamic logic (Kracht, 1995), temporal logic (Palm, 1999), monadic second-order logic (Rogers, 1998). However, most of them lead to the class of recognizable sets of trees (Thatcher & Wright, 1968). Provided a finite label domain this applies to all logical formalisms that are equal or weaker than the (weak) monadic second-order logic (Rabin, 1969). However, TAGs do not belong to this class, since TAGs are properly stronger than context-free grammars. But a set of trees is recognizable if and only if it can be recognized by tree automaton, which can be also encoded as a context-free grammar. Nevertheless, there are logical formalisms to specify structures beyond context-free derivations. For instance, Rogers proposes in (1999) and previous works a logical description of TAGs that is based on a 3-dimensional view of trees. The important issue of his approach is to combine the derived TAG-tree and its derivation tree to a single 3-dimensional structure.

Similarly, we propose a formal method to establish tree constraints outside the context-free paradigm that employs an additional tree structure that is linked with the tree in a particular manner. For TAGs we consider the corresponding TAG-derivation tree where each node of the derived TAG-tree is linked with the corresponding derivation node, e.g., if we adjoin the auxiliary tree β to the auxiliary tree α then we reach a derivation tree with the root m_α that has a single child m_β. Correspondingly, we link each node of the underlying initial tree α with the m_α node in the derivation tree and each node of the adjoined auxiliary tree β with the m_β node. Instead of labeling the nodes of the derivation tree with the name of the corresponding elementary tree and the tree address of the corresponding adjunction node, the former is sufficient due to these links. After adjoining a further β tree to the former β tree, the derivation tree includes a second m_β node below the first one. In addition, the nodes of the second β tree are linked with the second m_β node of the derivation tree. Obviously, the dominance relation in the derivation tree expresses nested auxiliary trees in the derived TAG-tree.

In contrast to Rogers' 3-dimensional trees, we keep the derived TAG-tree as a unit in order to

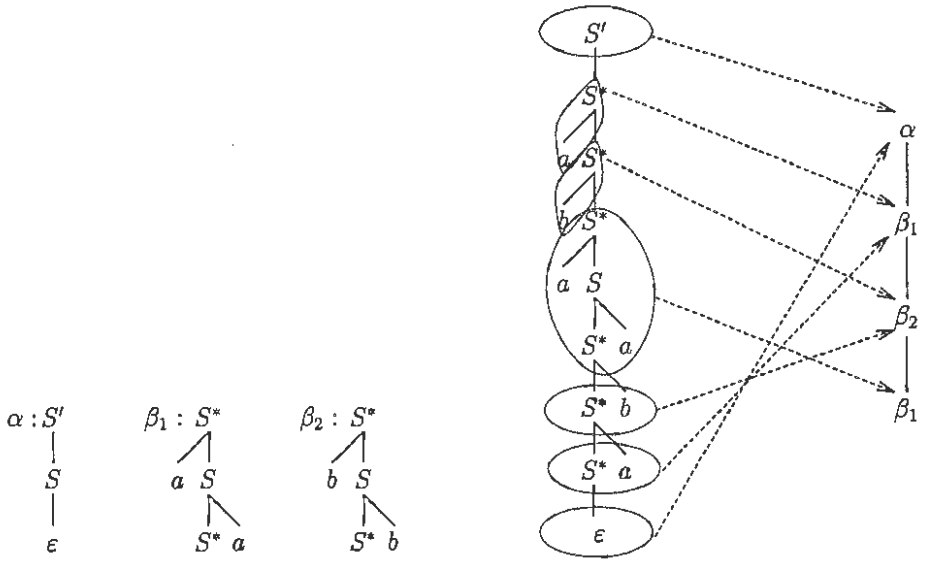

Figure 1: TAG generating the copy language and the derivation for *abaaba*

be able to access the TAG-tree directly without applying a particular projection (or a similar function) to the overall structure. Therefore we can still use one of the logical formalisms describing trees mentioned earlier to partially specify a set of TAG-trees. But if we want to make use of the special, non-context-free properties of TAGs, we must consider the links to refer to the corresponding nodes of the derivation tree. This linking function enables to specify sets of nodes in the TAG-tree, which we cannot describe in a formalism only capturing recognizable set of trees. As an illustrating example we consider a simple TAG generating the copy language $\{ww \mid w \in \{a,b\}^*\}$ (see Figure 1). Obviously, each occurrence of a letter in the first word shares the same auxiliary tree with the corresponding occurrence in the second word. In Figure 1 we find the corresponding TAG-tree and its derivation tree for the word *abaaba*.

In the approach presented here we make some important assumptions concerning TAGs. We employ a special node predicate *Adj* to indicate the adjunction nodes. Moreover, we take for granted that the root and the foot fails *Adj*, and the adjunction nodes do not immediately dominate each other. Therefore every adjoined tree is only bounded by nodes of the tree it was adjoined to. Instead of simple node labels we use a finite set Σ of unary node predicates. Hence, each node is labeled with the (finite) set of predicates that are valid for it. We may only adjoin an auxiliary tree at a node if this node, the root and the foot of this auxiliary tree share at least one common predicate. In the resulting tree the labels of the former root and the foot are the intersection of the labels of the adjunction node with the former labels of the root and foot, respectively. Finally, we consider the substitution, i.e., replacing a leaf with an elementary tree as a particular version of adjunction, where the foot of the adjoined tree remains a leaf.

2. A Logical Specification for TAGs and Their Derivations

Our specification language considers two structures, i.e., the resulting TAG-tree t and its derivation tree d, and the (total) function τ mapping the nodes of t to the corresponding nodes of d. We call the combined structure consisting of these components a *t/d-tree*, where the finite sets Σ and Σ_D denote the label domain for the TAG-tree and the derivation tree, respectively. In

detail, a t/d-structure $\langle T, D, \tau \rangle$ includes a Σ-labeled tree domain $T = \langle t, P_t \rangle$ for the TAG-tree, a Σ_D-labeled tree domain $D = \langle d, P_d \rangle$ and the linking function $\tau: t \rightarrow d$.
In order to specify a particular set of t/d-trees we employ a first-order style formalism which is similar to the one used in (Backofen *et al.*, 1995).[1] The resulting first-order language $L_{t/d}(\Sigma, \Sigma_D)$ includes the binary operators $\lhd$, $\prec$, $\lhd^*$, $\prec^*$, $\lhd_D$, $\lhd_D^*$ representing the immediate dominance relation, the sibling precedence relation and their transitive-reflexive closure for the TAG- and the derivation tree, respectively. The function τ maps each TAG-tree node to the corresponding derivation tree node. In addition, we introduce auxiliary predicates *root* and *foot* to mark the root and the foot of an elementary tree. Further, the predicate *leaf* indicates the leaves of an elementary tree that must keep this property during the whole derivation which is especially true for the foot of an initial tree.
Based on the first-order language $L_{t/d}(\Sigma, \Sigma_D)$ we can specify the formal properties of a well-formed t/d-tree. We consider the intended distribution of the linking function τ and the predicates *root* and *foot* when adjoining the auxiliary tree β to the tree α. Hence, the corresponding derivation tree includes two nodes m_α and m_β where $m_\alpha \lhd_D m_\beta$. Basically, for every derivation node m there is a unique root dominating a unique foot, either one referring to m $(T1)$. In addition, the root dominates all other nodes referring to m $(T2)$, and the foot dominates no other node referring to m $(T3)$. Finally, each node wearing the predicate *leaf* must be a leaf $(T4)$.

$$
\begin{aligned}
&(T1)\ \forall m \exists_1 n, n' \colon n \lhd^+ n' \wedge \tau(n) = m \wedge \tau(n') = m \wedge \mathit{root}(n) \wedge \mathit{foot}(n') \\
&(T2)\ \forall n, n' \colon \tau(n) = \tau(n') \wedge \mathit{root}(n) \Rightarrow n \lhd^+ n' \\
&(T3)\ \forall n, n' \colon \tau(n) = \tau(n') \wedge \mathit{foot}(n) \Rightarrow \neg\, n \lhd^+ n' \\
&(T4)\ \forall n, n' \colon \mathit{leaf}(n) \Rightarrow \neg\, n \lhd n' \\
&(T5)\ \forall n \lhd n' \colon (\tau(n) = \tau(n') \wedge \neg \mathit{root}(n) \wedge \neg \mathit{foot}(n')) \\
&\qquad \vee (\tau(n) \lhd_D \tau(n') \wedge \mathit{root}(n')) \vee (\tau(n') \lhd_D \tau(n) \wedge \mathit{foot}(n))
\end{aligned}
$$

where the quantifier $\exists_1$ denotes the unique existence. In $(T5)$ we consider the properties of pairs of immediately dominating nodes. Initially, an elementary tree is coherent, i.e., each node and each of its existing immediate neighbors are parts of the same elementary tree. But after adjoining a tree β at an adjunction node of α, this relationship is interrupted for α, namely between the root and the foot of β. Consequently, each pair of nodes n and n' with $n \lhd n'$ refer to the same elementary tree if neither $\mathit{foot}(n)$ nor $\mathit{root}(n')$ obtains. Otherwise, either n' is the root of β or n is the foot β, where α is the parent of β in the derivation tree, since according to the previous assumptions every adjoined tree is bounded by nodes of the tree we are adjoining to.
The constraints $(T1)$ to $(T5)$ sufficiently specify a valid TAG-tree and its derivation tree provided that either structure is a valid (ordered) finite tree. Hence, it must be possible to separate an arbitrary t/d-tree satisfying the above constraints into a corresponding set of elementary trees. We start this backward derivation at an arbitrary leaf m of the derivation tree. Following $(T1)$ there is a unique root n_r and a unique foot n_f in the TAG-tree marking the boundaries of the corresponding elementary tree. Due to $(T2)$ n_r dominates all nodes n with $\tau(n) = m$, and due to $(T3)$ n_f dominates none of them. Since m is a leaf, $(T5)$ asserts that all nodes dominated by n_r and not dominated by n_f refer to the same elementary tree m. Therefore we can undo the adjunction of the m-tree leading to an m-labeled auxiliary tree. We remove m in the derivation tree, and in the TAG-tree we replace the m-tree with a new adjunction node n_m referring to the parent of m and whose label is the union of the labels of n_r and n_f except *root* and *foot*. In the same manner we handle the remaining t/d-tree until a single derivation node

[1] Selecting first-order logic as the specifying formalism for both kind of structures should be considered as a working example rather than restricting our approach to this kind of logic. Nevertheless, one can employ all kinds of formalisms, e.g., monadic second order logic, that describe recognizable sets of (finite) trees.

remains which denotes the initial tree of the derivation. Finally we should note that this method does not ensure for elementary trees to be uniquely associated with their labels. However, an appropriate modification of the label domain Σ_D could assert this.
Consequently, every t/d-tree satisfying the constraints $(T1)$ to $(T5)$[2] is generated by a certain TAG whose necessary elementary trees result from the backward derivation described above. Since the backward derivation does not consider the inner structure of an elementary tree, i.e., the nodes satisfying neither *root* nor *foot*, this part of the considered TAG-tree can be underspecified. In that case the result of the backward derivation is underspecified, too. By a minor modified backward derivation which manages alternative results we could handle arbitrary underspecification as well.
Obviously, the TAG resulting from a backward derivation of an (underspecified) t/d-tree also generates other t/d-trees than the given one. More generally, one or more given (underspecified) t/d-trees may be considered as a system generating a TAG that recognizes at least these t/d-trees and its predecessors and successors in the TAG derivation. Thus, as a basic application the backward derivation can be employed to describe a particular property of TAG-trees by means of an underspecified t/d tree and to construct a corresponding set of elementary trees. As a linguistic application, we consider an underspecified t/d-tree describing a particular grammatical phenomenon, and hereafter, we achieve a corresponding TAG. Hence, we are able to obtain information on modeling syntactic properties by means of TAGs.

3. Representing Structure Sharing

Structure sharing is an important issue in natural language syntax. In general, it is necessary if a constituent occurs in a position that is different from the one licensing it (or at least a significant part of it). For instance, in the question "Which girl did we meet yesterday?", the object phrase "which girl" occurs in the sentence initial position rather than in the object position immediately after the verb, where it receives its case and θ-role. Typically, we represent structure sharing as a derivational process called movement, i.e., a moved phrase XP_i leaves a trace t_i at its former position; hence we write "[Which girl]$_i$ did we meet t_i yesterday?". Similarly, we handle topicalized objects, e.g., "[This nice girl]$_i$ we met t_i yesterday". However, the indices we use to indicate structure sharing give rise to a problem concerning the finiteness of the label domain Σ. In general, an arbitrary number of such indices may occur. This leads to an infinite number of necessary labels which we cannot handle in our $L_{t/d}$-formalism. However, we will illustrate how to handle structure sharing in TAGs without employing such indices.
Most TAG approaches to wh-movement and topicalization, e.g., XTAG (1999), assume an initial tree that describes the whole sentence structure including the moved phrase and its trace. Consequently, we require similar trees for all kinds of movement and sentence structures. Moreover the (structural) distance between the co-indexed nodes is bounded according to the specification in its initial tree. However, this method fails to represent long distance movement as in

Who$_i$ do we think that Bill knows that Rachel saw that John kissed t_i.

where the distance (within the tree) between the moved node and its trace is arbitrary since such a structure requires a series of adjunctions between the co-indexed nodes. In order to reach a more general approach to movement in TAGs, we apply the backward derivation to such crucial tree structures. In detail, we extend a given tree to a corresponding t/d-structure satisfying $(T1)$ to $(T5)$ by assigning appropriate values for τ, *root* and *foot*. As a basic property of t/d-trees we proposed that we store shared information in the derivation tree rather than using indices

[2]Obviously, (T4) can be ignored since the *leaf* predicate only prevents adjunctions beyond the tree considered. Nevertheless, to achieve a more restricted TAG we may assume that initially all leaves must satisfy *leaf*.

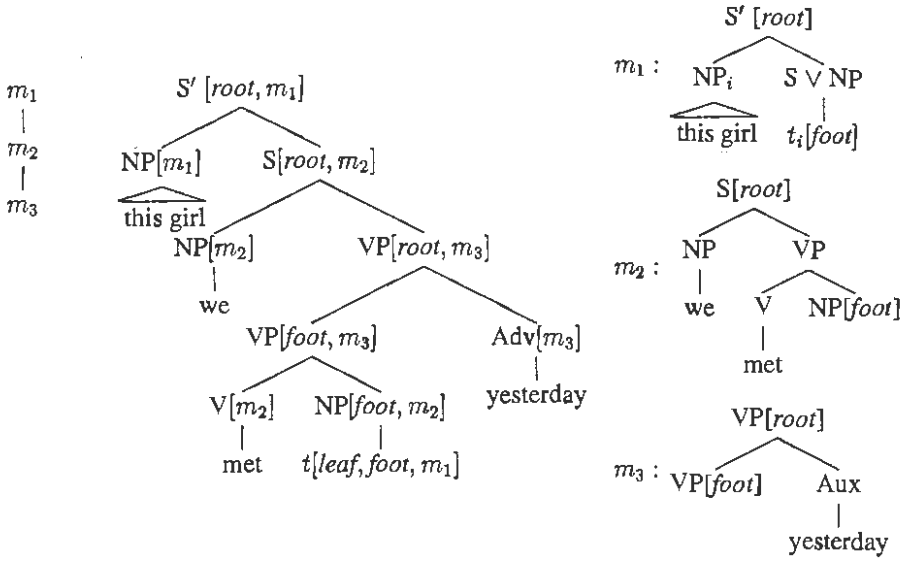

Figure 2: t/d-tree and TAG for "this girl we met yesterday"

in the derived TAG tree. Then we can express an arbitrary number of shared properties while keeping a finite label domain for the TAG tree. We presume therefore that fundamentally shared structures belong to the same elementary tree. Accordingly, we assign the moved noun phrase and its trace to the same elementary tree m_1 which must also include the S' node as its root. Note that this does not mean that S' must share any properties with the NP, actually m_1 only serves to store the common features of the NP and its trace. For the foot of m_1 we select the trace. Similarly, we obtain that S is the root of m_2 and the object NP its foot. Finally, the adverb is assigned to the auxiliary tree m_3 with both VP nodes as its root and foot. The resulting t/d-tree and the corresponding elementary trees is shown in (Figure 2) where we do not explicitly write the *leaf* predicate that is assigned to the trace and the lexical entries.

Generalizing the result obtained above, the starting and the landing position of a movement are part of the same elementary tree to which we must adjoin the structure occurring between. The distance between a moved phrase and its trace depends on the number and complexity of the elementary trees adjoined to the movement tree where additional constraints on the derivation tree can restrict this distance. However, adjoining the inner structure seems to be inconsistent to most other current TAG approaches to natural languages. Nevertheless, this conflict turns out to be only superficial if we assume initial trees where the position of the foot and the substitution nodes are underspecified. For instance, we consider the argument structure of a verb where the nodes for the arguments are marked for substitution or to be the foot. So we can move at most argument and the remaining ones must be substituted; for the moved argument we assume a corresponding substitution node at the landing position, too.

Since movement is not restricted to NPs we assume a more general elementary tree for movement where the category of the moved phrases is underspecified and the moved phrase must be inserted via substitution. Moreover if we select appropriate predicates for the adjunction node, we can specify the auxiliary trees that can be adjoined. Through the resulting elementary tree for movement we can express movement as a particular version of adjunction rather than as a lexical process. Since the moved phrase and its trace are linked by a corresponding elementary

tree no further co-indexing is necessary. As a result we obtain TAGs that require only a finite label domain. Thus, a corresponding the $L_{t/d}$-formula can specify such TAGs appropriately.

4. Conclusion

We have introduced a logical description of TAGs that simultaneously considers the derived and derivation tree, both of which are linked together via a special function. By the constraints $(T1)$ to $(T5)$ we have obtained a notion of TAG-validity that is applicable to arbitrary tree structures. In detail, we have established a backward derivation to verify whether an (underspecified) tree can be generated by a TAG. Using this method we have obtained an alternative approach to represent structure sharing without employing indices in TAG. This way we can describe structure sharing within the $L_{t/d}$ formalism, too. Formally seen, we focus the properties of the adjunction operation rather than putting together complex initial trees. As a further application, it should be possible to extend the backward derivation to a learning algorithm that extracts a TAG from a given tree corpus. Another obvious extension of our $L_{t/d}$ formalism emerges when we consider the derivation tree as a derived tree which is linked with a further derivation tree. This leads to Weir's hierarchy of control languages (1992).

An alternative approach to express structure sharing in TAGs is provided by several variants of *multi-component TAGs* (Weir, 1988; Rambow, 1994) where a set of elementary trees is simultaneously adjoined (or substituted). Obviously, such a set identifies its members. However, there may be an arbitrary number of such sets in a derived tree which means an arbitrary number of indices and, hence, an infinite label domain. Nevertheless our approach can be extended to such formalisms as long as the label domains are finite and the derivation trees are recognizable.

References

BACKOFEN R., ROGERS J. & VIJAY-SHANKER K. (1995). A first-order axiomatization of the theory of finite trees. *Journal of Logic, Language and Information*, 4, 5–39.

KRACHT M. (1995). Syntactic codes and grammar refinement. *Journal of Logic, Language and Information*, 4, 41–60.

PALM A. (1999). Propositional tense logic for finite trees. In *Proceedings of 6th Meeting on Mathematics of Language (MOL6)*.

RABIN M. (1969). Decidability of second-order theories and automata on infinite trees. *Transactions of the American Mathematical Society*, **141**, 1–35.

RAMBOW O. (1994). *Formal and Computational Aspects of Natural Language Syntax*. PhD thesis, Department of Computer and Information Science, University of Pennsylvania.

ROGERS J. (1998). *A Descriptive Approach to Language-Theoretic Complexity*. Stanford, California: CSLI.

ROGERS J. (1999). Generalized tree-adjoining grammars. In *Proceedings of 6th Meeting on Mathematics of Language (MOL6)*.

THATCHER J. & WRIGHT J. (1968). Generalized finite automata theory with an application to decision porblems of second-order logic. *Mathematical System Theory*, **2**, 57–81.

WEIR D. (1988). *Characterizing Mildly Context-Sensitive Grammar Formalisms*. PhD thesis, Department of Computer and Information Science, University of Pennsylvania, Phildelphia.

WEIR D. (1992). A geometric hierarchy beyond context-free grammars. *Theoretical Computer Science*, **104**, 235–261.

XTAG RESEARCH GROUP (1999). *A lexicalized tree-adjoining grammar for English*. technical report, Institut for Research in Cognitve Science, University of Pennsylvania, Philadelphia.