A Unified Notion of Derived and Derivation Structures in TAG (Preliminary Version)

James Rogers, University of Central Florida, `jrogers@cs.ucf.edu`

Generalizing CFGs

It has been recognized for some time that TAGs can be viewed as a particular sort of generalization of CFGs. Vijay-Shanker, Weir, and Joshi [VSWJ87] and Weir [Wei88], for instance, note a number of parallels between CFLs and TALs—both in their formal properties and in the fact that a number of characterizations of the CFLs can be generalized to provide characterizations of the TALs and TAG tree sets. Some of these generalizations iterate naturally, constructing infinite hierarchies of languages falling between the CFLs and the CSLs. This view of CFLs and TALs as adjacent levels of natural hierarchies of language classes is attractive in that it supports the transfer of results between the two and provides a route to more powerful formalisms which come with many of their properties already established.

But there is a discontinuity between the CFGs and the TAGs and their higher-order generalizations. The relationship between the set of structures that characterize the derivations of a CFG (i.e., its derivation trees) and the set of structures it derives is extremely simple regardless of whether we understand it to derive trees (in which case the relationship is identity) or strings (in which case it is the yield). This is true because the derivation structures embody the effect of the derivation steps—the expansion of a non-terminal into a string—directly. This is not the case for the traditional derivation structures for TAGs, where a derivation step involves the expansion of a non-terminal into a tree. If the derivation structures are taken to be trees themselves they cannot directly embody the actual expansion. Rather they just encode which trees are adjoined, which tree they adjoin into, and at which node of that tree the adjunction takes place. To obtain the derived tree from the derivation tree one must actually carry out these adjunctions. Thus the derived structure is not a substructure of the derivation structure. This separation of derived and derivation structures is extremely useful in that it abstracts away from the form of the derived structures. This allows formalisms which derive quite distinct classes of structures to be compared. Indeed, such comparison is the main thrust of Weir's dissertation. It does, however, introduce an asymmetry that mars the analogy between CFGs and TAGs and, we argue here, masks regularities that can be identified in the progression from regular sets, through CFLs and TALs, and beyond.

In this paper we take the parallel between string substitution in CFGs and tree adjunction in TAGs absolutely at face value. Just as we understand a CF rewrite rule to license the expansion of a node in a tree into a string of child nodes, we interpret an auxiliary tree in a TAG as licensing the expansion of a node in a three-dimensional tree-like structure into a tree of child nodes. Because we define these structures uniformly across an arbitrary range of dimensions, we provisionally refer to them as (labeled) *tree manifolds*. These (3-dimensional) tree manifolds encode derivations in TAGs in precisely the same way that trees (2-dimensional tree manifolds) encode derivations in CFGs. There is, in fact, an easy mapping between TAG derivation trees of the traditional sort and these structures. Note that there are two distinct notions of immediate dominance in these structures, the immediate dominance relations in the elementary trees of the TAG and the relationship between a node and the nodes of the tree that expands it. When we take the maximal set of points with respect to this second relation—the two-dimensional yield of the three-dimensional structure—we get exactly the phrase-structure tree the derivation generates. When we then take the maximal set of points with respect to the domination relation within that tree—the one-dimensional yield of the two-dimensional structure—we get the string the derivation generates. In this way the operation taking the derivation structure into the derived phrase-structure tree is a higher-dimensional analog of the operation taking the derivation structure of a CFG into the string it generates. We take, then, these 3-dimensional tree manifolds to be both the derived and derivation structures of the TAG in exactly the same way that derivation trees can be taken to be both the derived and derivation structures of a CFG.

Tree Manifolds

We build the formal notion of labeled tree manifolds by analogy with *tree domains* [Gor67] in which nodes in a tree are assigned addresses which are strings in $\mathbb{N}^*$ with the root at address ε and the children of the node at address w at $w0$, $w1$,... in left-to-right order. A *labeled* tree domain includes a mapping assigning labels from some alphabet Σ to the nodes in the domain. Tree domains are subject to two well-formedness properties: they must be downward closed ($wv \in T \Rightarrow w \in T$ for all $w, v \in \mathbb{N}^*$) and they must be left-sibling closed ($wi \in T$ and $j < i \Rightarrow wj \in T$ for all $w \in \mathbb{N}^*$, $i, j \in \mathbb{N}$). These two properties can be expressed uniformly if one employs unary encoding for the natural numbers. Thus $n \in \mathbb{N}$ is encoded as 1^n and node addresses become sequences of strings of '1's (e.g., the address 1031 becomes $\langle\langle 1\rangle, \langle\rangle, \langle 111\rangle, \langle 1\rangle\rangle$).[1] The left-sibling closure property then becomes a downward closure property at the level of these strings of '1's: $s \cdot \langle wv\rangle \in T \Rightarrow s \cdot \langle w\rangle \in T$ for all $w, v \in \{1\}^*$, $s \in (\{1\}^*)^*$. From this perspective, strings are just a specialization of the notion of labeled tree domain—a prefix closed set of sequences of '1's (rather than sequences of such sequences) along with a labeling function.

One can interpret the address of a node in a tree domain as the sequence of string addresses one follows in traversing the path from the root to that node. This gives us the point of view we need in generalizing to higher dimensions. An address of a node in a 3-dimensional tree manifold is the sequence of tree addresses one follows in traversing the path to it from the root. Thus a 3-dimensional tree manifold is a set of sequences of sequences of sequences of '1's

[1] We could be pedantic and employ sequences of empty sequences $\langle\rangle$, but the notation is difficult enough to read as it stands.

(a third-order sequence of '1's). As with the first two levels this set of sequences must also be prefix closed. In general, an i-dimensional tree manifold will be a set of i^{th} -order sequences of '1's that is hereditarily prefix closed.

There is, however, a problem here. In the case of tree domains we are expanding nodes to linear structures. There is no ambiguity about how these structures fit together; the rightmost node dominated by a given node must immediately precede the leftmost node dominated by its immediate right sibling. In the case of 3-dimensional tree manifolds, however this is not the case. In expanding a node into a tree, any one of the nodes in the yield of that tree could be taken to dominate the subtree rooted at the original node.[2] If the set of maximal points (in the third dimension) are to form a coherent tree, each of the trees expanding a node must have a distinguished node at which the subtree will be spliced; in TAG terminology there must be a designated foot node. Projecting this down to the string level one gets a requirement that each string must be headed. Thus we will take our string addresses to be sequences of 'l's, designating positions to the left of the head, or 'r's, designating positions to the right of the head. Tree addresses are, again, sequences of such sequences, and the spine of a tree is just the path that follows heads at each step. The designated foot node, then, is the maximal node at an address that is a sequence of empty sequences. Note that we do not need to modify the downward closure property in any way.[3]

Tree Manifold Grammars and Automata

This notion of tree manifolds gives us a natural hierarchy of grammars. A grammar at level n is a finite set of labeled *local* (depth one in the major dimension) n-dimensional tree manifolds. The set of structures derived by such a grammar is the set of labeled n-dimensional tree manifolds that can be constructed by concatenating the local tree manifolds in the grammar. Note that maximal nodes of the tree manifold form *trivial* tree manifolds—consisting of a node and an empty set of children. As these must each be licensed by a tree manifold included the grammar just like every other local tree manifold, we can identify the set of symbols labeling trivial tree manifolds in the grammar as the terminals of the grammar, albeit terminals that potentially may be rewritten. Similarly, we will usually be interested in sets of tree manifolds in which the root is labeled with (one of) a designated (set of) start symbol(s). At the 2-dimensional level this gives us a straightforward generalization of CFGs in the form of sets of local trees [GS84]. A 1-dimensional tree manifold grammar consists of pairs of labels licensing the expansion of labeled nodes into labeled nodes to form strings, with the label of each node in the string depending only on the label of its predecessor. These generate the *strict 2-locally testable* languages, a much weaker class

[2] Note that our operation expanding nodes to trees is adjunction-like—the subtree rooted at the node is preserved as a single subtree, its children may not be split into multiple subtrees.

[3] While this approach is successful for the 3-dimensional case, its generalization to the higher-dimensional cases is yet incomplete. For our purposes here the first four levels suffice. (0-dimensional tree manifolds, of course, are just points.)

than the regular languages. At the 3-dimensional level we get TAGs generalized in the sense that the label of the node at which the adjunction is taking place, the label of the root of the adjoined tree, and the label of its foot may all be distinct. The fact that the possibility of adjunction depends only on the label of the node makes these grammars look superficially like *pure* TAGs—those without adjoining constraints. The requirement that maximal nodes be licensed, however, has the same effect as Obligatory Adjoining (OA) constraints while the fact that auxiliary trees may adjoin at dissimilarly labeled nodes has the same effect on generative capacity as Null Adjoining (NA) constraints. While one cannot, in the strictest sense, directly express Selective Adjoining (SA) constraints in these grammars, as we will see shortly they can generate any tree set generated by a TAG with such constraints. By generalization from the 2-dimensional terminology, we will refer to the sets licensed by these grammars as *local* sets of tree manifolds.

We also get a natural hierarchy of automata. An n-dimensional tree manifold automaton is a finite set of pairs associating labels (in the alphabet Σ) and local n-dimensional tree manifolds labeled with states from a finite set Q. A run of the automaton on a Σ-labeled n-dimensional tree manifold is an assignment of states to the nodes of the manifold in which the states assigned to each local manifold are associated by the automaton with the label of its root.[4] Such an automaton accepts a labeled n-dimensional tree manifold relative to some set of initial states iff there is a run of the automaton in which the root is assigned a state in that set. In the one dimensional case we get ordinary Finite State Automata. In the two dimensional case we get exactly the (non-deterministic) tree automata and in the three dimensional case we get exactly TAGs with adjoining constraints, modulo the generalization permitting adjunction of auxiliary trees at dissimilarly labeled nodes. Again by generalization from the 2-dimensional terminology, we will refer to the sets licensed by these automata as *recognizable* sets of tree manifolds.

It should be noted that the hierarchy of languages associated with the grammars and automata of this hierarchy is *not* the hierarchy of Linear Context-Free Re-writing Systems of Weir's dissertation [Wei88], but rather appears to be closer to the hierarchies of [VSWJ87].

Analysis

The key property of the local sets is the fact that the set of structures that may expand a node depends only on the label of that node—in other words, the features that constrain the form of the structure must be explicit in its labeling. This is the property that distinguishes them from the recognizable sets, where these features may be "hidden" in the state. It is this ability to hide features that allows us to capture SA constraints directly in the automata. Note, though, that a simple lift of Thatcher's [Tha67] proof that the yield languages of recognizable sets of trees are CFLs establishes that 3-dimensional tree manifold grammars and automata are equivalent in the sets of trees they

[4] Here we have notions of licensing of terminal nodes and restriction to tree manifolds with roots assigned states in a distinguished set analogous to the similar notions in the grammars.

yield (and, *a fortiori*, in the sets of strings they yield). The construction of the proof also dispels the apparent paradox in the fact that tree manifold grammars can directly express OA and NA constraints but not SA constraints while the strong generative capacity of pure TAGs is generally understood to be unaffected by SA constraints and extended by OA or NA constraints. In capturing SA constraints in a pure TAG one must extend the labels to explicitly encode the constraints in exactly the way that Thatcher's construction extends the labels of the grammar to encode states. In both contexts the classes of tree sets are equivalent modulo a projection. Of course, in the case of tree manifolds there is no need to extend the labels of the maximal nodes, and so trees they yield may have the original labels. The distinction between the local and recognizable sets arises at the level of the derivation structure—at the level of trees in the CF case, at the level of tree manifolds in the TAG case. This level has to do with the way in which the elementary structures are analyzed, with the structure of strings in CFGs, with the structure of trees in TAGs. It is at this level that the properties of elementary trees can be formalized and it is at this level that the linguistic significance of the distinction between local and recognizable sets of tree manifolds will arise if it exists.

One can limit the grammars to generate only pure TAG tree sets by restricting them to local tree manifolds in which the root node and the root and foot of the child tree are identically labeled, and by requiring all nodes to be licensed to have null expansions. Thus the distinction between pure TAGs and TAGs in general, from this point of view, is a consequence of stipulations that have been placed on the form of the grammar. To the extent that the formalism is intended to capture characteristics of natural language, such stipulations are explicit expressions of the theory of syntax it embodies. Here they are consequences of two of the most basic aspects of the linguistic motivation of TAGs. The restriction on the labeling of the local tree manifolds is a consequence of the fact that auxiliary trees are intended to capture recursive structures. The requirement that every label be licensed as a terminal is a consequence of fact that initial trees are intended to capture the minimal non-recursive structures of the languages—and thus every "sentential form" of a derivation will be in the language. Adjoining constraints, in part, have the effect of moderating the formal effect of these restrictions.

Schabes and Shieber [SS94] have proposed an alternative notion of derivation in TAG in which multiple trees may be adjoined at a single node. One of their motivations is the fact that when a phrase is multiply modified the auxiliary trees encoding the modifiers must nest when they adjoin to the node rooting that phrase. Nevertheless, the significant relationships are between the modifier trees and the modified node. By admitting derivation trees with each of the modifiers adjoined at the same node, those relationships can be expressed locally even though the trees are not locally related in the derived tree. From the standard point of view, the restriction that no more than one tree adjoin to a node is primarily stipulative. If we take our derivation structures to be tree manifolds, however, the restriction becomes a simple physical fact. Schabes and Shieber's conception of derivation can, nonetheless, be accommodated if one recognizes that their goal is to express what, from the point of view of tree manifolds, are non-local relationships.[5] As shown by Joshi and Levy [JL77] there is a very rich class of such constraints that can be employed in defining sets of trees without leaving the realm of recognizable sets. In fact, in their original conception, adjoining constraints were expressed by such *domination* and *proper analysis predicates* [JLT75]. By lifting Joshi and Levy's proofs to the level of tree manifolds one can show that such predicates do not extend the capacity of TAGs with only SA, OA and NA constraints.

This leads us, at last, to the original motivation for these studies. We expect that tree manifolds will provide us with the class of models needed to extend the model theoretic techniques we have applied to local and recognizable sets of trees to apply to TAG tree sets. This approach allows the properties of the derived and derivation structures to be expressed as ordinary logical predicates, providing an extremely natural way of capturing linguistic intuitions, including those that might be expressed by domination and proper analysis predicates. More importantly, this approach provides a uniform framework for formalizing linguistic theories. Whereas in the past its applicability has been limited to theories that license CF (and regular) string languages, the extension to TAG tree sets and higher dimensional generalizations will enable us to address a range of mildly context-sensitive formalisms.

References

[Gor67] Saul Gorn. Explicit definitions and linguistic dominoes. In John F. Hart and Satoru Takasu, editors, *Systems and Computer Science, Proceedings of the Conference held at Univ. of Western Ontario, 1965*. Univ. of Toronto Press, 1967.

[GS84] Ferenc Gécseg and Magnus Steinby. *Tree Automata*. Akadémiai Kiadó, Budapest, 1984.

[JL77] Aravind K. Joshi and Leon S. Levy. Constraints on structural descriptions: Local transformations. *SIAM Journal of Computing*, 6(2):272–284, 1977.

[JLT75] Aravind K. Joshi, L. S. Levy, and M. Takahashi. Tree adjunct grammars. *Journal of Computer and System Sciences*, 10:136–163, 1975.

[Shi94] Stuart M. Shieber. Restricting the weak-generative capacity of synchronous tree-adjoining grammars. *Computational Intelligence*, 10(4):371–385, 1994.

[SS90] Stuart M. Shieber and Yves Schabes. Synchronous tree-adjoining grammars. In *Proceedings of the 13th International Conference on Computational Linguistics*, volume 3, pages 253–258, Helsinki, 1990. Association for Computational Linguistics.

[SS94] Yves Schabes and Stuart M. Shieber. An alternative conception of tree-adjoining derivation. *Computational Linguistics*, 20(1):91–124, 1994.

[5] In the case of synchronous TAGs, it appears that the variation between Shieber and Schabes's original definition [SS90] and the revised definition of [Shi94, SS94] manifests itself as a variation in the permissible mappings between tree manifolds much as the modifications to the revised definition explored in [Shi94] are expressed as variations in the permissible mappings between derivation structures.

[Tha67] J. W. Thatcher. Characterizing derivation trees of context-free grammars through a generalization of finite automata theory. *Journal of Computer and System Sciences*, 1:317–322, 1967.

[VSWJ87] K. Vijay-Shanker, David J. Weir, and Aravind K. Joshi. On the progression from context-free to the tree adjoining languages. In A. Manaster-Ramer, editor, *Mathematics of Language*, pages 389–401. John Benjamins, Amsterdam, 1987.

[Wei88] David J. Weir. *Characterizing Mildly Context-Sensitive Grammar Formalisms.* PhD thesis, University of Pennsylvania, 1988.