A Descriptive Characterization of Tree-Adjoining Languages (Full Version) *

James Rogers
Dept. of Computer Science
Univ. of Central Florida
Orlando, FL 32816-2362, USA
jrogers@cs.ucf.edu

Abstract

Since the early Sixties and Seventies it has been known that the regular and context-free languages are characterized by definability in the monadic second-order theory of certain structures. More recently, these descriptive characterizations have been used to obtain complexity results for constraint- and principle-based theories of syntax and to provide a uniform model-theoretic framework for exploring the relationship between theories expressed in disparate formal terms. These results have been limited, to an extent, by the lack of descriptive characterizations of language classes beyond the context-free. Recently, we have shown that tree-adjoining languages (in a mildly generalized form) can be characterized by recognition by automata operating on three-dimensional tree manifolds, a three-dimensional analog of trees. In this paper, we exploit these automata-theoretic results to obtain a characterization of the tree-adjoining languages by definability in the monadic second-order theory of these three-dimensional tree manifolds. This not only opens the way to extending the tools of model-theoretic syntax to the level of TALs, but provides a highly flexible mechanism for defining TAGs in terms of logical constraints.

1 Introduction

In the early Sixties Büchi (1960) and Elgot (1961) established that a set of strings was regular iff it was definable in the weak monadic second-order theory of the natural numbers with successor (wS1S). In the early Seventies an extension to the context-free languages was obtained by Thatcher and Wright (1968) and Doner (1970) who established that the CFLs were all and only the sets of strings forming the yield of sets of finite trees definable in the weak monadic second-order theory of multiple successors (wSnS). These descriptive characterizations have natural application to constraint- and principle-based theories of syntax. We have employed them in exploring the language-theoretic complexity of theories in GB (Rogers, 1994; Rogers, 1997b) and GPSG (Rogers, 1997a) and have used these model-theoretic interpretations as a uniform framework in which to compare these formalisms (Rogers, 1996). They have also provided a foundation for an approach to principle-based parsing via compilation into tree-automata (Morawietz and Cornell, 1997). Outside the realm of Computational Linguistics, these results have been employed in theorem proving with applications to program and hardware verification (Henriksen et al., 1995; Biehl et al., 1996; Kelb et al., 1997). The scope of each of these applications is limited, to some extent, by the fact that there are no such descriptive characterizations of classes of languages beyond the context-free. As a result, there has been considerable interest in extending the basic results (Mönnich, 1997; Volger, 1997) but, prior to the work reported here, the proposed extensions have not preserved the simplicity of the original results.

Recently, in (Rogers, 1997c), we introduced a class of labeled three-dimensional tree-like structures (three-dimensional tree manifolds—3-TM) which serve simultaneously as the derived and derivation structures of Tree Adjoining-Grammars (TAGs) in exactly the same way that labeled trees can serve as both derived and derivation structures for CFGs. We defined a class of automata over these struc-

* This is the full version of a paper to appear in the proceedings of COLING-ACL'98 as a project note (Rogers, 1998).

UCF Technical Report:CS-TR-98-01

tures that are a natural generalization of tree-automata (which are, in turn, an analogous generalization of ordinary finite-state automata over strings) and showed that the class of tree manifolds recognized by these automata are exactly the class of tree manifolds generated by TAGs if one relaxes the usual requirement that the labels of the root and foot of an auxiliary tree and the label of the node at which it adjoins all be identical.

Thus there are analogous classes of automata at the level of labeled three-dimensional tree manifolds, the level of labeled trees and at the level of strings (which can be understood as two- and one-dimensional tree manifolds) which recognize sets of structures that yield, respectively, the TALs, the CFLs, and the regular languages. Furthermore, the nature of the generalization between each level and the next is simple enough that many results lift directly from one level to the next. In particular, we get that the recognizable sets at each level are closed under union, intersection, relative complement, projection, cylindrification, and determinization and that emptiness of the recognizable sets is decidable. These are exactly the properties one needs to establish that recognizability by the automata over a class of structures characterizes satisfiability of monadic second-order formulae in the language appropriate for that class. Thus, just as the proofs of closure properties lift directly from one level to the next, Doner's and Thatcher and Wright's proofs that the recognizable sets of trees are characterized by definability in wSnS lift directly to a proof that the recognizable sets of three-dimensional tree manifolds are characterized by definability in their weak monadic second-order theory (which we will refer to as wSnT3).

In this paper we carry out this program. In the next three sections we introduce 3-TMs and our uniform notion of automaton over tree manifolds of arbitrary (finite) dimension and sketch, as an example, proofs of closure under determinization, projection and cylindrification that are independent of the dimensionality. In Sections 5 and 6 we introduce wSnT3, the weak monadic second-order theory of n-branching 3-TM, and sketch the proof that the sets recognized by 3-TM automata are exactly the sets definable in wSnT3. This, when coupled with the characterization of TALs in Rogers (1997c), gives us our descriptive characterization of TALs: a set of strings is generated by a TAG (modulo the generalization of Rogers (1997c)) iff it is the (string) yield of a set of 3-TM definable in wSnT3. Finally, in Section 7 we look at how working in wSnT3 allows a potentially more natural means of defining TALs and, in particular, a simplified treatment of constraints on modifiers in TAGs.

2 Tree Manifolds

Tree manifolds are a generalization to arbitrary dimensions of Gorn's *tree domains* (Gorn, 1967). A tree domain is a set of node address drawn from $\mathbb{N}^*$ (that is, a set of strings of natural numbers) in which ε is the address of the root and the children of a node at address w occur at addresses $w0, w1, \ldots$, in left-to-right order. To be well formed, a tree domain must be downward closed wrt to domination, which corresponds to being prefix closed, and left sibling closed in the sense that if wi occurs then so does wj for all $j < i$. In generalizing these, we can define a one-dimensional analog as *string domains*: downward closed sets of natural numbers interpreted as string addresses. From this point of view, the address of a node in a tree domain can be understood as the sequence of string addresses one follows in tracing the path from the root to that node. If we represent $\mathbb{N}$ in unary (with n represented as 1^n) then the downward closure property of string domains becomes a form of prefix closure analogous to downward closure wrt domination in tree domains, tree domains become sequences of sequences of '1's, and the left-closure property of tree domains becomes a prefix closure property for the embedded sequences.

Raising this to higher dimensions, we obtain, next, a class of structures in which each node expands into a (possibly empty) tree. A, *three-dimensional tree manifold* (3-TM), then, is set of sequences of tree addresses (that is, addresses of nodes in tree domains) tracing the paths from the root of one of these structures to each of the nodes in it. Again this must be downward closed wrt domination in the third dimension, equivalently wrt prefix, the sets of tree addresses labeling the children of any node must be downward closed wrt domination in the second dimension (again wrt to prefix), and the sets of string addresses labeling the children of any node in any of these trees must be downward closed wrt domination in the first dimension (left-of, and, yet again, prefix).[1] Thus 3-TM, tree domains (2-TM), and string domains (1-TM) can be defined uniformly as d^{th}-order sequences of '1's which are hereditarily prefix closed. We will denote the set of all 3-TM as $\mathbb{T}^d$, so $\mathbb{T}^1 \subseteq \boldsymbol{\mathcal{P}}(1^*)$ is the set of all string domains, $\mathbb{T}^2 \subseteq \boldsymbol{\mathcal{P}}((1^*)^*)$ the set of all tree domains, and $\mathbb{T}^3 \subseteq \boldsymbol{\mathcal{P}}(((1^*)^*)^*)$ the set of all 3-TM, where $\boldsymbol{\mathcal{P}}(S)$ is the power set of S.[2]

For any alphabet Σ, a *Σ-labeled d-dimensional tree manifold* is a pair $\langle T, \tau \rangle$ where T is a d-TM and $\tau : T \to \Sigma$ is an assignment of labels in Σ to the nodes in T. We will denote the set of all Σ-labeled d-TM as $\mathbb{T}^d_\Sigma$.

3 Tree Manifold Automata

Mimicking the development of tree manifolds, we can define automata over labeled 3-TM as a generalization of automata over labeled tree domains which, in turn, can be understood as an analogous generalization of ordinary finite-state automata over strings (labeled string domains). A d-TM automaton with state set Q and alphabet Σ is a finite set:

$$\mathcal{A}^d \subseteq \Sigma \times Q \times \mathbb{T}^{d-1}_Q .$$

The interpretation of a tuple $\langle \sigma, q, \mathcal{T} \rangle \in \mathcal{A}^d$ is that if a node of a d-TM is labeled σ and $\mathcal{T}$ encodes the assignment of states to its children, then that node may be assigned state q.[3] A *run* of an d-TM automaton $\mathcal{A}$ on a Σ-labeled d-TM $\mathcal{T} = \langle T, \tau \rangle$ is an assignment $r : T \to Q$ of states in Q to nodes in T in which each assignment is licensed by $\mathcal{A}$. Note that this implies that a maximal node (wrt to the major dimension) labeled σ may be assigned state q only if there is a tuple $\langle \sigma, q, \varepsilon \rangle \in \mathcal{A}^d$ where ε is the empty $(d-1)$-TM. If we let $Q_0 \subseteq Q$ be any set of *accepting states*, then the set of (finite) Σ-labeled d-TM recognized by $\mathcal{A}$, relative to Q_0, is that set for which there is a run of $\mathcal{A}$ that assigns the root a state in Q_0:

$$\begin{aligned}
&\mathcal{A}(Q_0) \stackrel{\text{def}}{=} \\
&\quad \{\mathcal{T} = \langle T, \tau \rangle \mid T \text{ finite and} \\
&\qquad \exists r : T \to Q \text{ such that} \\
&\qquad\quad r(\varepsilon) \in Q_0 \text{ and for all } s \in T \\
&\qquad\quad \Big\langle \tau(s), r(s), \langle T, r \rangle|_{\mathrm{Ch}(T,s)} \Big\rangle \in \mathcal{A} \,\}
\end{aligned}$$

where $\mathrm{Ch}(T, s) = \{w \in \mathbb{T}^{(d-1)} \mid s \cdot \langle w \rangle \in T\}$ and

$$\begin{aligned}
&\langle T, r \rangle|_{\mathrm{Ch}(T,s)} = \\
&\qquad \langle \mathrm{Ch}(T, s), \{w \mapsto r(s \cdot \langle w \rangle) \mid w \in \mathrm{Ch}(T, s)\} \rangle .
\end{aligned}$$ [4]

A set of d-TM is *recognizable* iff it is $\mathcal{A}(Q_0)$ for some d-TM automaton $\mathcal{A}$ and set of accepting states Q_0.

4 Uniform Properties of Recognizable Sets

The strength of the uniform definition of d-TM automata is that many, even most, properties of the sets they recognize can be proved uniformly—independently of their dimension. For instance, let us say that the *depth* of a TM is the length of the longest sequence it includes (just the length of the top level sequence, independent of the length of the sequences it may contain). The *branching factor* of a TM *at a given dimension* is the maximum depth of the

[1]While this clearly iterates to obtain tree manifolds of any finite dimension, we are concerned only with the first three dimensions (four, counting points—zero-dimensional tree manifolds).

[2]In (Rogers, 1997c) we constructed tree-manifolds from headed strings in order to obtain a unique tree as the two-dimensional yield of a 3-TM. Here we treat this as a matter of interpretation, keeping the simple notion of tree-manifold and moving the issue of headedness into the relational structures we build on them.

[3]This is a "bottom-up" interpretation. There is an analogous "top-down" interpretation, but for all $d \geq 2$ automata that are deterministic under the top-down interpretation are strictly weaker than those that are non-deterministic, while those that are deterministic under the bottom-up interpretation are equivalent to the non-deterministic variety. It should be emphasized that the only place the distinction between top-down and bottom-up arises is in the definition of determinism. These automata are interpreted purely declaratively, as licensing assignments of states to nodes.

[4]In general, we will employ w and s in this manner where w denotes a sequence of some order and s denotes a sequence of sequences of the order of w (i.e., a sequence of the next higher order). Concatenation will always be interpreted as an operation on sequences of the same order. Thus, $s \cdot \langle w \rangle$ is a sequence of sequences in which the last sequence is w. We will also use t and v as we use s and w, and will employ p for sequences of the next higher order than s and t when needed.

structures it contains in that dimension. The (overall) *branching factor* of a d-TM is the maximum of its branching factors at all dimensions strictly less than d. For a 3-TM, then, the branching factor is the larger of the maximum depth of the trees it contains and the maximum length of the strings it contains. A TM is n-*branching* iff its branching factor is no greater than n. We will denote the set of all Σ-labeled, n-branching, d-TM as $\mathbb{T}_{\Sigma}^{n,d}$. A d-TM automaton is *deterministic* with respect to a branching factor n (in the bottom-up sense) iff

$$(\forall \sigma \in \Sigma, \mathcal{T} \in \mathbb{T}_{Q}^{n,d-1})(\exists! q \in Q)[\langle \sigma, q, \mathcal{T}\rangle \in \mathcal{A}].^{5}$$

It is easy to show, using a standard subset-construction, that (bottom-up) determinism does not effect the recognizing power of d-TM automata of any dimension. Given $\mathcal{A} \subseteq \Sigma \times Q \times \mathbb{T}_{Q}^{n,d-1}$, let

$$\begin{aligned}
&\hat{\mathcal{A}} \subseteq \Sigma \times \boldsymbol{\mathcal{P}}(Q) \times \mathbb{T}^{n,d-1}_{\boldsymbol{\mathcal{P}}(Q)} \\
&\quad \stackrel{\text{def}}{=} \quad \{\langle \sigma, Q_1, \langle T, \tau'\rangle\rangle \mid \\
&\qquad\qquad Q_1 \subseteq Q,\ \tau' : T \to \boldsymbol{\mathcal{P}}(Q), \\
&\qquad\qquad q \in Q_1 \iff \\
&\qquad\qquad\quad (\exists \tau : T \to Q)[\\
&\qquad\qquad\qquad \langle \sigma, q, \langle T, \tau\rangle\rangle \in \mathcal{A} \wedge \\
&\qquad\qquad\qquad (\forall x \in T)[\tau(x) \in \tau'(x)]\ ], \\
&\hat{Q}_0 \quad \stackrel{\text{def}}{=} \quad \{Q_i \subseteq Q \mid Q_i \cap Q_0 \neq \emptyset\}.
\end{aligned}$$

It is easy to verify that $\hat{\mathcal{A}}$ is deterministic and that $\hat{\mathcal{A}}(\hat{Q}_0) = \mathcal{A}(Q_0)$. More importantly, while the dimension of the TM automaton parameterizes the type of the objects manipulated by the proof, it has no effect on the way in which they are manipulated—the proof itself is essentially independent of the dimension.

Proof of closure of recognizable sets under projection and cylindrification is even easier. A *projection* is any (usually many-to-one) surjective map from one alphabet onto another. A *cylindrification* is an "inverse" projection. Let $\pi : \Sigma \to \Sigma'$ be any projection, $\mathcal{T} = \langle T, \tau\rangle$ a Σ-labeled d-TM and $\mathcal{A}$ an automaton over Σ-labeled d-TM. Then $\pi(\mathcal{T}) \stackrel{\text{def}}{=} \langle T, \pi \circ \tau\rangle$ and

$$\pi(\mathcal{A}) \stackrel{\text{def}}{=} \{\langle \pi(\sigma), q, \mathcal{T}\rangle \mid \langle \sigma, q, \mathcal{T}\rangle \in \mathcal{A}\}.$$

[5]The quantifier $\exists!$ should be read "exists exactly one".

It is easy to see that

$$\mathcal{T} \in \mathcal{A}(Q_0) \iff \pi(\mathcal{T}) \in \pi(\mathcal{A})(Q_0).$$

Similarly, if $\mathcal{A} \subseteq \Sigma' \times Q \times \mathbb{T}_{Q}^{(d-1)}$ let

$$\pi^{-1}(\mathcal{A}) \stackrel{\text{def}}{=} \{\langle \sigma, q, \mathcal{T}\rangle \mid \langle \pi(\sigma), q, \mathcal{T}\rangle \in \mathcal{A}\}.$$

Then $\pi(\mathcal{T}) \in \mathcal{A}(Q_0) \iff \mathcal{T} \in \pi^{-1}(\mathcal{A})(Q_0)$.

Similar uniform proofs can be obtained for closure of recognizable sets under Boolean operations and for decidability of emptiness.

5 wSnT3

We are now in a position to build relational structures on d-dimensional tree manifolds. Let T_n^d be the *complete* n*-branching* d*-TM*—that in which every point has a child structure that has depth n in all its $(d-1)$ dimensions. Let

$$\mathsf{T}_n^3 \stackrel{\text{def}}{=} \langle T_n^3, \triangleleft_1, \triangleleft_2, \triangleleft_3\rangle$$

where, for all $x, y \in T_n^3$:

$$\begin{aligned}
x \triangleleft_3 y &\stackrel{\text{def}}{\iff} y = x \cdot \langle s\rangle \\
x \triangleleft_2 y &\stackrel{\text{def}}{\iff} x = p \cdot \langle s\rangle \text{ and } y = p \cdot \langle s \cdot \langle w\rangle\rangle \\
x \triangleleft_1 y &\stackrel{\text{def}}{\iff} x = p \cdot \langle s \cdot \langle w\rangle\rangle \text{ and} \\
&\qquad\quad y = p \cdot \langle s \cdot \langle w \cdot v\rangle\rangle
\end{aligned}$$

where $p \in ((1^*)^*)^*,\ s \in (1^*)^*,\ w \in 1^*,\ v \in 1^+$ (which is to say that $x \triangleleft_i y$ iff x is the immediate predecessor of y in the i^{th} -dimension).

The *weak monadic second-order language* of T_n^3 includes constants for each of the relations (we let them stand for themselves), the usual logical connectives, quantifiers and grouping symbols, and two countably infinite sets of variables, one ranging over individuals (for which we employ lowercase) and one ranging over *finite* subsets (for which we employ uppercase). If $\varphi(x_1, \ldots, x_n, X_1, \ldots, X_m)$ is a formula of this language with free variables among the x_i and X_j, then we will assert that it is satisfied in T_n^3 by an assignment $\boldsymbol{s}$ (mapping the 'x_i's to individuals and 'X_j's to finite subsets) with the notation

$$\mathsf{T}_n^3 \models \varphi\,[\boldsymbol{s}].$$

A *sentence* is a formula with no free variables—formulae for which truth in T_n^3 is not contingent on an assignment. The set of all sentences of this language that are satisfied by T^3_n is the *weak monadic second-order theory of* T^3_n, denoted wSnT3.[6]

6 Definability in wSnT3

A set $\mathbb{T}$ of Σ-labeled 3-TM is definable in wSnT3 iff there is a formula $\varphi_{\mathbb{T}}(X_T, X_\sigma)_{\sigma\in\Sigma}$, with free variables among X_T (interpreted as the domain of a tree) and X_σ for each $\sigma \in \Sigma$ (interpreted as the set of σ-labeled points in T), such that

$$\begin{aligned}&\langle T, \tau\rangle \in \mathbb{T} \iff \\ &\quad \mathsf{T}^3_n \models \varphi_{\mathbb{T}}\,[X_t \mapsto T, X_\sigma \mapsto \{p \mid \tau(p) = \sigma\}].\end{aligned}$$

It should be reasonably easy to see how any recognizable set can be defined in this way. Suppose the i^{th} tuple of 3-TM automaton $\mathcal{A}$ is $\langle a, 0, {}_{0}{\diagup}^{1}{\diagdown}_{1}\rangle$. A local (depth one in its major dimension) 3-TM (labeled with both Σ and Q) is compatible with this iff its root satisfies

$$\begin{aligned}&\varphi_i(x) \equiv \\ &(\exists x_1, x_2, x_3)[X_T(x_1) \wedge X_T(x_2) \wedge X_T(x_3) \wedge \\ &\quad X_a(x) \wedge X_0(x) \wedge X_1(x_1) \wedge X_0(x_2) \wedge X_1(x_3) \wedge \\ &\quad (\forall y)[X_T(y) \rightarrow \\ &\qquad\quad (x \triangleleft_3 y \leftrightarrow (y \approx x_1 \vee y \approx x_2 \vee y \approx x_3) \wedge \\ &\qquad\quad\; x_1 \triangleleft_2 y \leftrightarrow (y \approx x_2 \vee y \approx x_3) \wedge \\ &\qquad\quad\; \neg x_2 \triangleleft_2 y \wedge \neg x_3 \triangleleft_2 y \wedge \\ &\qquad\quad\; x_2 \triangleleft_1 y \leftrightarrow y \approx x_3 \wedge \\ &\qquad\quad\; \neg x_3 \triangleleft_1 y \quad) \qquad]\end{aligned}$$

We can then require every node in X_T to be licensed by some tuple in $\mathcal{A}$ by requiring it to satisfy $\bigvee_i[\varphi_i(x)]$, the disjunction of such formulae for all tuples in $\mathcal{A}$. All that remains is to require the root to be labeled with an accepting state and to "hide" the states by existentially binding them:

$$\begin{aligned}&(\exists X_q)_{q\in Q}(\forall x)[\ (X_T(x) \rightarrow \textstyle\bigvee_i[\varphi_i(x)]) \wedge \\ &\qquad\qquad (\neg(\exists y)[y \triangleleft_3 x] \rightarrow \textstyle\bigvee_{q\in Q_0}[X_q(x)])\,].\end{aligned}$$

It is not hard to show that a Σ-labeled 3-TM $\mathcal{T}$ corresponds to a satisfying assignment for this formula iff there is a run of $\mathcal{A}$ on $\mathcal{T}$ which assigns an accepting state to the root.

The proof that every set of trees definable in wSnT3 is recognizable, while a little more involved, is essentially a lift of the proofs of Doner (1970) and Thatcher and Wright (1968). The initial step is to show that every formula in the language of wSnT3 can be reduced to equivalent formulae in which only set variables occur and which employ only the predicates $X \subseteq Y$ (with the obvious interpretation) and $X \triangleleft_i Y$ (satisfied iff X and Y are both singleton and the sole element of X stands in the appropriate relation to the sole element of Y). We can define, for instance,

$$\text{Empty}(X) \equiv (\forall Y)[Y \subseteq X \rightarrow X \subseteq Y]$$

and

$$\begin{aligned}&\text{Singleton}(X) \equiv \\ &\quad (\forall Y)[Y \subseteq X \rightarrow (\text{Empty}(Y) \vee X \subseteq Y)]\end{aligned}$$

Then $x \triangleleft_i y$ becomes

$$\text{Singleton}(X) \wedge \text{Singleton}(Y) \wedge X \triangleleft_i Y.$$

It is easy to construct 3-TM automata (over the alphabet $\boldsymbol{\mathcal{P}}(\{X, Y\})$) which accept trees encoding satisfying assignments for these atomic formulae. For example, assignments satisfying $X \triangleleft_3 Y$ in T^3_2 are in $\mathcal{A}(2)$ for $\mathcal{A}$:

$$\begin{array}{ll}\langle \emptyset, 0, \mathcal{T}\rangle, & \mathcal{T} \in \{\varepsilon,\ {}_{0}{\diagup}^{0}{\diagdown}_{0}\}, \\ \langle \{Y\}, 1, \mathcal{T}\rangle, & \mathcal{T} \in \{\varepsilon,\ {}_{0}{\diagup}^{0}{\diagdown}_{0}\}, \\ \langle \{X\}, 2, \mathcal{T}\rangle, & \mathcal{T} \in \{\ {}_{0}{\diagup}^{1}{\diagdown}_{0},\ {}_{1}{\diagup}^{0}{\diagdown}_{0},\ {}_{0}{\diagup}^{0}{\diagdown}_{1}\}, \\ \langle \emptyset, 2, \mathcal{T}\rangle, & \mathcal{T} \in \{\ {}_{0}{\diagup}^{2}{\diagdown}_{0},\ {}_{2}{\diagup}^{0}{\diagdown}_{0},\ {}_{0}{\diagup}^{0}{\diagdown}_{2}\}, \\ \langle \sigma, 3, \mathcal{T}\rangle, & \quad\text{otherwise}\ .\end{array}$$

The extension to arbitrary formulae (over these atomic formulae) can then be carried out by induction on the structure of the formulae using the closure properties of the recognizable sets.

7 Defining TALs in wSnT3

The signature of wSnT3 is inconvenient for expressing linguistic constraints. In particular, one of the strengths of the model-theoretic approach is the ability to define long-distance relationships without having to explicitly encode them in the labels of the intervening nodes. We can extend the immediate predecessor relations to relations corresponding to (proper) *above* (within the 3-TM), *domination* (within a tree), and *precedence* (within a set of siblings) using:

$$\begin{aligned}&x \mathrel{\bar{\triangleleft}_i} y \overset{\text{def}}{\iff} x \neq y \wedge (\exists X)[X(x) \wedge X(y) \wedge \\ &\quad (\forall z)[X(z) \rightarrow (z \approx y \vee (\exists! z')[X(z') \wedge z \triangleleft_i z'])]].\end{aligned}$$

[6] wS1T1 is equivalent to wS1S in the sense of interinterpretability, as is wS1Td for all d. wSnT2 is interinterpretable with wSnS for all $n \geq 2$.

Which simply asserts that there is a sequence of (at least two) points linearly ordered by $\triangleleft_i$ in which x precedes y.[7]

To extend these through the entire structure we have to address the fact that the two dimensional yield of a 3-TM is not well defined—there is nothing that determines which leaf of the tree expanding a node dominates the subtree rooted at that node. To resolve this, we extend our structures to include a set H picking out exactly one head in each set of siblings, with the "foot" of a tree being that leaf reached from the root by a path of all heads. Given H, it is possible to define $\triangleleft_2^+$ and $\triangleleft_1^+$, variations of dominance and precedence[8] that are inherited by substructures in the appropriate way. Let:

$$\begin{array}{l}\mathrm{Spine}_2(x) \overset{\mathrm{def}}{\Longleftrightarrow} \mathrm{H}(x) \wedge \\ \quad (\forall y)[y \bar{\triangleleft}_2 x \rightarrow (H(y) \vee \neg(\exists z)[z \triangleleft_2 y])\end{array}$$

and

$$x \triangleleft_i^* y \overset{\mathrm{def}}{\Longleftrightarrow} x \triangleleft_i^+ y \vee x \approx y.$$

Then

$$\begin{array}{l} x \triangleleft_2^+ y \overset{\mathrm{def}}{\Longleftrightarrow} \\ \quad (\exists x', y')[x' \triangleleft_3^* x \wedge y' \triangleleft_3^* y \wedge x' \bar{\triangleleft}_2 y' \wedge \\ \qquad\qquad (\forall z)[(x' \triangleleft_3^+ z \wedge z \triangleleft_3^* x) \rightarrow \mathrm{Spine}_2(z)]\ ] \end{array}$$

and

$$\begin{array}{l} x \triangleleft_1^+ y \overset{\mathrm{def}}{\Longleftrightarrow} \\ \quad (\exists x', y')[x' \triangleleft_2^* x \wedge y' \triangleleft_2^* y \wedge x' \bar{\triangleleft}_1 y'] \end{array}$$

At the same time, it is convenient to include the labels explicitly in the structures. A headed Σ-labeled 3-TM, then, is a structure:

$$\langle T, \triangleleft_i, \bar{\triangleleft}_i, \triangleleft_i^+, H, P_\sigma \rangle_{1 \leq i \leq 3,\, \sigma \in \Sigma},$$

where T is a rooted, connected subset of T_n^3 for some n.

With this signature it is easy to define the set of 3-TM that captures a TAG in the sense that their 2-dimensional yields—the set of maximal points wrt $\triangleleft_3^+$, ordered by $\triangleleft_2^+$ and $\triangleleft_1^+$—form the set of trees derived by the TAG. Note that obligatory (OA) and null (NA) adjoining constraints translate to a requirement that a node be (non-)maximal wrt $\triangleleft_3^+$. In our automata-theoretic interpretation of TAGs selective adjoining (SA) constraints are encoded in the states. Here we can express them directly: a constraint specifying the modifier trees which may adjoin to an N node, for instance, can be stated as a condition on the label of the root node of trees immediately below N nodes.

In general, of course, SA constraints depend not only on the attributes (the label) of a node, but also on the elementary tree in which it occurs and its position in that tree. Both of these conditions are actually expressions of the local context of the node. Here, again, we can express such conditions directly—in terms of the relevant elements of the node's neighborhood. At least in some cases this seems likely to allow for a more general expression of the constraints, abstracting away from the irrelevant details of the context.

Finally, there are circumstances in which the primitive locality of SA constraints in TAGs is inconvenient. Schabes and Shieber (1994), for instance, suggest allowing multiple adjunctions of modifier trees to the same node on the grounds that selectional constraints hold between the modified node and each of its modifiers but, if only a single adjunction may occur at the modified node, only the first tree that is adjoined will actually be local to that node. They point out that, while it is possible to pass these constraints through the tree by encoding them in the labels of the intervening nodes, such a solution can have wide ranging effects on the overall grammar. As we noted above, the expression of such non-local constraints is one of the strengths of the model-theoretic approach. We can state them in a purely natural way—as a simple restriction on the types of the modifier trees which can occur below (in the $\triangleleft_3^+$ sense) the modified node.

8 Conclusion

We have obtained a descriptive characterization of the TALs via a generalization of existing characterizations of the CFLs and regular languages. These results extend the scope of the model-theoretic tools for obtaining language-theoretic complexity results for constraint- and principle-based theories of syntax to the TALs and, carrying the generalization to arbitrary dimensions, should extend it to cover a wide range of mildly context-sensitive language classes. Moreover, the generalization is natural enough that the

[7] This is partly a consequence of the fact that assignments to X are required to be finite.

[8] Of course $\triangleleft_3^+$ is just $\bar{\triangleleft}_3$.

results it provides should easily integrate with existing results employing the model-theoretic framework to illuminate relationships between theories. Finally, we believe that this characterization provides an approach to defining TALs in a highly flexible and theoretically natural way.

References

M. Biehl, N. Klarlund, and T. Rauhe. 1996. Algorithms for guided tree automata. In Darrell Raymond, Derick Wood, and Sheng Yu, editors, *Automata implementation, WIA '96*, LNCS 1260, London, Ontario.

J. R. Büchi. 1960. Weak second-order arithmetic and finite automata. *Zeitschrift für mathematische Logik und Grundlagen der Mathematik*, 6:66–92.

John Doner. 1970. Tree acceptors and some of their applications. *Journal of Computer and System Sciences*, 4:406–451.

Calvin C. Elgot. 1961. Decision problems of finite automata design and related arithmetics. *Transactions of the American Mathematical Society*, 98:21–51.

Saul Gorn. 1967. 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.

J. G. Henriksen, J. Jensen, M. Jørgensen, N. Klarlund, R. Paige, T. Rauhe, and A. Sandhol. 1995. MONA: Monadic second-order logic in practice. In Ed Brinksma, editor, *Tools and Algorithms for the Construction and Analysis of Systems, TACAS '95*, LNCS 1019, Aarhus, Denmark.

P. Kelb, T. Margaria, M. Mendler, and C. Gsottberger. 1997. MOSEL: A flexible toolset for monadic second-order logic. In Ed Brinksma, editor, *Tools and Algorithms for the Construction and Analysis of Systems, TACAS '97*, LNCS 1217, Enschede, The Netherlands.

Uwe Mönnich. 1997. Adjunction as substitution: An algebraic formulation of regular, context-free and tree adjoining languages. In *Formal Grammar*, Aix-en-Provence, Fr.

Frank Morawietz and Tom Cornell. 1997. Representing constraints with automata. In *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics*, pages 468–475, Madrid, Spain.

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

James Rogers. 1996. A model-theoretic framework for theories of syntax. In *Proceedings of the 34th Annual Meeting of the Associaton for Computational Linguistics*, pages 10–16, Santa Cruz, CA.

James Rogers. 1997a. "Grammarless" phrase structure grammar. *Linguistics and Philosophy*, 20:721–746.

James Rogers. 1997b. On descriptive complexity, language complexity, and GB. In Patrick Blackburn and Maarten de Rijke, editors, *Specifying Syntactic Structures*, pages 157–184. CSLI Publications.

James Rogers. 1997c. A unified notion of derived and derivation structures in TAG. In *Proceedings of the Fifth Meeting on Mathematics of Language MOL5 '97*, Saarbrücken, FRG.

James Rogers. 1998. A descriptive characterization of tree-adjoining languages (project note). In *Proc. of the 17th International Conference on Computational Linguistics (COLING'98) and the 36th Annual Meeting of the Association for Computational Linguistics (ACL'98)*, Montreal. ACL. Project Note.

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

J. W. Thatcher and J. B. Wright. 1968. Generalized finite automata theory with an application to a decision problem of second-order logic. *Mathematical Systems Theory*, 2(1):57–81.

Hugo Volger. 1997. Principle languages and principle based parsing. Technical Report 82, SFB 340, Univ. of Tübingen.