JOURNAL OF COMPUTER AND SYSTEM SCIENCES 10, 136–163 (1975)

Tree Adjunct Grammars*

ARAVIND K. JOSHI

Moore School of Electrical Engineering, University of Pennsylvania, Philadelphia, Pennsylvania 19174

LEON S. LEVY

University of Delaware, Newark, Delaware 19711

AND

MASAKO TAKAHASHI†

University of Pennsylvania, Philadelphia, Pennsylvania 19104

Received February 13, 1973

In this paper, a tree generating system called a tree adjunct grammar is described and its formal properties are studied relating them to the tree generating systems of Brainerd (*Information and Control* **14** (1969), 217–231) and Rounds (*Mathematical Systems Theory 4* (1970), 257–287) and to the recognizable sets and local sets discussed by Thatcher (*Journal of Computer and System Sciences* 1 (1967), 317–322; **4** (1970), 339–367) and Rounds. Linguistic relevance of these systems has been briefly discussed also.

1. INTRODUCTION

In this paper, we will describe a tree generating system called a tree adjunct grammar (tag) and study its formal properties relating them to the tree generating systems of Brainerd [3, 4] and Rounds [16], and also to the recognizable sets and local sets discussed by Thatcher [17, 18] and Rounds [16].

* This work was partially supported by NSF GS-35125, NSF GS-27, and the Advanced Research Seminar "Tree Mappings in Linguistics," sponsored by the Mathematical Social Science Board (1972). Some work on this paper was carried out by the first author while he was on leave at The Institute for Advanced Study as a Guggenheim Fellow.

We are indebted to the referee of this paper for his extremely valuable comments which helped us in improving the presentation.

† Present address: Tokyo Institute of Technology, Tokyo, Japan.

In Section 2, we will define the tag systems, following an idea of Parikh [15] and study their properties. These grammars surprisingly turn out to be more powerful than context-free grammars. Much of the basic notation for trees and subtrees in this section is from Brainerd [3] Gorn [6], and Rounds [16]. The relationship of tag languages to context-free, indexed, and context-sensitive languages is discussed in Section 3 and to recognizable sets in Section 4. Section 5 deals with an important subclass of tag's called simple tag's, and another subclass called linear tag's is discussed in Section 6. In Section 7, we introduce a generalization of tag's and study them with respect to the properties mentioned in the previous sections. A variant of a tag and a related open problem is discussed in Section 8.

These grammars were motivated by several linguistic considerations. Our main purpose in this paper is the study of the formal properties of these grammars. However, before proceeding with this study, we will describe very briefly the linguistic relevance of these systems:

1. A tree is a structural description of a sentence. A tag is thus a grammar of structural descriptions. There are two types of basic trees in a tag, the center trees and the adjunct trees. The center trees can be regarded as elementary (structured)[1] sentences and the adjunct trees have the semantic interpretation of modifiers.

   A given tree (structural description) may have more than one distinct derivation in a tag. These distinct derivations invariably turn out to have linguistic relevance. For example, the phrase structure tree corresponding to *big red house* has two distinct derivations in the tag corresponding to the two readings (*big* (*red* (*house*))) and ((*big*) (*red*) (*house*)), i.e., corresponding, respectively, to the cases when *big* modifies *red house* and when *big* and *red* modify *house* independently.

2. In a tag, each intermediate tree in the derivation is also a sentential tree, i.e., the derivation proceeds from a (structured) sentence to another (structured) sentence, in contract to the usual derivation in a phrase structure grammar. Further, the set of all (structured) sentences corresponding to the intermediate steps in the derivation of a (structured) sentence can be regarded as the set of all partial (structured) sentences underlying the given (structured) sentence. These two properties are of interest because they capture to some extent two key linguistic concepts which can be very informally stated as follows [5, 7]. Sentences are related to other sentences in systematic ways and complex sentences are related to simple sentences and can be viewed as composed of simpler sentences which have been subjected to appropriate deformations.

3. Simple tag's in Section 5 suggest a framework for formulating the question:

[1] A structured sentence is a sentence together with its tree.

How much hierarchical structure is necessary for sentence description? It turns out that most of the base components of currently available transformational grammars fall into the class of simple tag's.

A generalization of tag's (stag's in Section 7) provides a certain amount discontiguous constituent structure and it is quite adequate for most linguistic purposes [8].

2. A Tree Generating System (Tree Adjunct Grammar)

First, a notation for "trees." Let J^* be the free monoid generated by J (set of all natural numbers). Let the binary operation be denoted by $\cdot$ and the identity by 0. For $p, q \in J^*$, $p \leqslant q$ iff there is a $r \in J^*$ such that $q = p \cdot r$ and $p < q$ iff $p \leqslant q$ and $p \neq q$.

Let V be a finite alphabet and $\Sigma \subset V$. We call Σ the "terminal alphabet" and $V - \Sigma$ the "nonterminal alphabet."[2]

Definition 2.1. γ is a "tree over V" iff it is a function from D_γ into V where the domain D_γ is a finite subset of J^* such that (1) if $q \in D_\gamma$, $p < q$, then $p \in D_\gamma$; (2) if $p \cdot j \in D_\gamma$, $j \in J$, then $p \cdot 1, p \cdot 2, \ldots, p \cdot (j - 1) \in D_\gamma$.

We call elements in D_γ addresses of γ. If $(p, A) \in \gamma$ then we say that A is the label of the node at the address p in γ. We will often write this as $\gamma(p) = A$.

Definition 2.2. Let γ be a tree over V. A node q in γ(more precisely, $q \in D_\gamma$) is called a "terminal node" iff for all $p \in D_\gamma$, $q \not< p$. A node q in γ is an "interior (nonterminal) node" iff q is not a terminal node. A node whose address is 0 is called the "root node."

Let τ_V be the set of all trees over V such that if $\gamma \in \tau_V$ and $p \in D_\gamma$ is an interior node then $\gamma(p) \in V - \Sigma$. That is, interior nodes must be labeled with a nonterminal symbol. Terminal nodes may be labeled with a terminal or a nonterminal symbol.

Definition 2.3. Let $\gamma \in \tau_V$ and $p \in D_\gamma$. Then,

$$\gamma/p \triangleq \{(q, A) \mid (p \cdot q, A) \in \gamma, q \in J^*\},$$

$$\gamma\backslash p \triangleq \{(q, A) \mid (q, A) \in \gamma, p \not< q\}.$$

γ/p is called the "subtree" at p and $\gamma\backslash p$ is called the "supertree" of γ at p. Further, for $\gamma \in \tau_V$ and $p \in J^*$,

$$p \cdot \gamma \triangleq \{(p \cdot q, A) \mid (q, A) \in \gamma\}.$$

[2] We have here a ranked alphabet $\langle V, r \rangle$ where $r \subseteq V \times \omega$ is a finite relation called the ranking relation. If $r(\sigma, n)$ then we say that r has "rank" n and V_n denotes the set of symbols of rank n [18]. We also require V to satisfy the condition: $V_0 (= \Sigma) \cap V_i = \varnothing, i \neq 0$.

PROPOSITION 2.1.

1. *If* $\gamma \in \tau_V$, $\gamma \neq \varnothing$, *then* $0 \in D_\gamma$.
2. $\gamma = \gamma\backslash p \cup p \cdot (\gamma/p)$, *for* $\gamma \in \tau_V$ *and* $p \in D_\gamma$.
3. $\gamma\backslash p \in \tau_V$ *and* $\gamma/p \in \tau_V$, *for* $\gamma \in \tau_V$ *and* $p \in D_\gamma$.

However, $p \cdot \gamma \notin \tau_V$ *unless* $p = 0$, *for* $\gamma \in \tau_V$ *and* $p \in J^*$.

DEFINITION 2.4. The "yield" Y is a function from τ_V into V^* defined as follows.

$$Y(\gamma) = \gamma(0) \quad \text{if} \quad D_\gamma = \{0\}.$$

$$Y(\gamma) = Y(\gamma/1)\, Y(\gamma/2) \cdots Y(\gamma/j) \quad \text{if} \quad 1, 2, \ldots, j \in D_\gamma, \quad \text{and} \quad j+1 \notin D_\gamma.$$

V^* is the free monoid generated by V with identity ϵ. Thus $f(\gamma)$ is the string of the labels of the terminal nodes of γ.

DEFINITION 2.5. The "front" $\hat{\gamma}$ of γ in τ_V is defined as

$$\hat{\gamma} \triangleq \{(p, A) \in \gamma \mid p \prec q, \qquad \text{for any} \quad q \in D_\gamma\},$$

i.e., $\hat{\gamma}$ is the set of address–label pairs corresponding to the terminal nodes of γ.

DEFINITION 2.6. A sequence $\langle (p_0, A_0), (p_1, A_1), \ldots, (p_l, A_l)\rangle$ of elements of $\gamma \in \tau_V$ is a "path" of γ iff

$$p_0 = 0, \quad (p_l, A_l) \in \gamma, \quad p_i = p_{i-1} \cdot j_i \qquad \text{for some} \quad j_i \in J (i = 1, 2, \ldots, l).$$

l is the length of the path.

For $\gamma \in \tau_V$, the "path set" p_γ of γ is the set of all paths of γ. If $\tau' \subseteq \tau_V$, $P(\tau') = \bigcup_{\gamma \in \tau'} P_\gamma$. We call $P(\tau')$ the path set of τ'. We are now ready to define a tree generating system called a tree adjunct grammar (tag).

DEFINITION 2.7. A "tree adjunct grammar" (tag), $\mathcal{G}$ is a pair $\mathcal{G} = (\mathcal{C}, \mathcal{A})$ where $\mathcal{C}$ and $\mathcal{A}$ are finite subsets of τ_V satisfying the following conditions.

1. If $\gamma \in \mathcal{C}$ then $Y(\gamma) \in \Sigma^*$ and $\gamma(0) = S$ where S is a distinguished symbol in $V - \Sigma$.
2. If $\beta \in \mathcal{A}$ and $\alpha(0) = X$ then $X \in V - \Sigma$, $Y(\beta) \in \Sigma^* X \Sigma^+ \cup \Sigma^+ X \Sigma^*$ (where $\Sigma^+ = \Sigma\Sigma^*$)[3].

$\mathcal{C}$ is called the set of "center trees" and $\mathcal{A}$ the set of "adjunct trees." The set $\mathcal{C} \cup \mathcal{A}$ will be called the set of "basic trees" of $\mathcal{G}$.

[3] Thus the yield of an adjunct tree has at least one terminal symbol.

DEFINITION 2.8. Let β be an adjunct tree and $\gamma \in \tau_T$. Let $p \in D_\gamma$ and $\gamma(p) = \beta(0)$. Then β is "adjoinable to γ at p" and $\gamma\,[p, \beta]$ is the tree obtained from γ adjoining β at p defined as

$$\gamma[p, \beta] \triangleq \gamma\backslash p \cup p \cdot \beta \cup (p \cdot r) \cdot (\gamma/p),$$

where $r \in D_\beta$, $\beta(r) = \beta(0)$, and $(r, \beta(r)) \in \hat{\beta}$, i.e., r is the address of that node which is in the front of β and which has the label $\beta(0)$. This operation will be called "adjunction"

For a given $\mathcal{G} = (\mathcal{C}, \mathcal{A})$ we will write $\gamma \vdash_{\mathcal{G}} \gamma'$ iff for some β and some p, β is adjoinable to γ at p and $\gamma' = \gamma[p, \beta]$. $\overset{*}{\vdash}_{\mathcal{G}}$ is the reflexive, transitive closure of $\vdash_{\mathcal{G}}$.

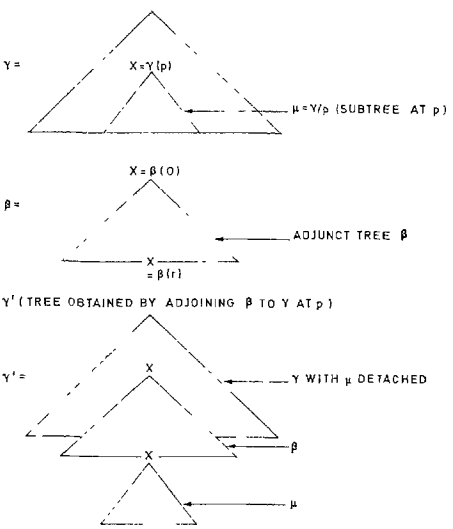


FIG. 1. Adjunction.

DEFINITION 2.9. Let $\mathcal{G} = (\mathcal{C}, \mathcal{A})$ be a tag. Then the "tree set"[4] of $\mathcal{G}$, $\tau(\mathcal{G})$ is defined as

$$\tau(\mathcal{G}) \triangleq \{\gamma \in \tau_T \mid \text{ for some } \alpha \in \mathcal{C},\ \alpha \overset{*}{\vdash}_{\mathcal{G}} \gamma\},$$

and the language of $\mathcal{G}$, $L(\mathcal{G})$ (a "tree adjunct language," tal)

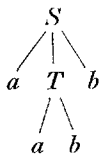
$$L(\mathcal{G}) \triangleq \{x \mid x = Y(\gamma). \quad \text{for some } \gamma \in \tau(\mathcal{G})\}.$$

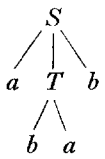
Let β be adjoinable to γ at p, then $\gamma[p, \beta](p) = \gamma(p) = \beta(0)$. Hence β is again adjoinable to $\gamma[p, \beta]$ at p. Write $\gamma[p, \beta]^n$ for $(\gamma[p, \beta]^{n-1})\ [p, \beta]\ (n > 1)$. Then by

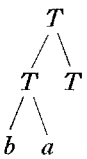
[4] In the terminology of Rounds [16], we can call the tree set of $\mathcal{G}$ the dendrolanguage of $\mathcal{G}$. Tree sets of tags are included in context-free dendrolanguages (see Section 4).

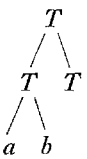
induction, for $n > 1$, β is adjoinable to $\gamma[p, \beta]^{n-1}$ at p if it is so to γ at p; hence $\gamma[p, \beta]^n$ is well defined. Further given $\mathcal{G} = (\mathcal{C}, \mathcal{A})$, if $\gamma \in \tau(\mathcal{G})$, and $\beta \in \mathcal{A}$, then for $n > 1$, $\gamma[p, \beta]^n \in \tau(\mathcal{G})$.

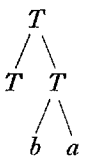
EXAMPLE 2.1. Let $\mathcal{G}_1 = (\mathcal{C}, \mathcal{A})$ where $\mathcal{C} = \{\alpha_1, \alpha_2\}$, and $\mathcal{A} = \{\beta_1, \beta_2, ..., \beta_6\}$.

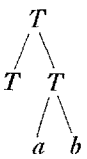
$\alpha_1 =$ 

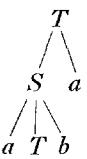
$\alpha_2 =$ 

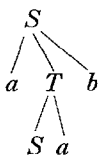
$\beta_1 =$ 

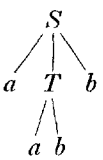
$\beta_2 =$ 

$\beta_3 =$ 

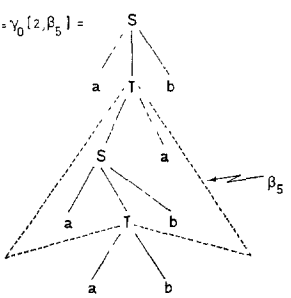
$\beta_4 =$ 

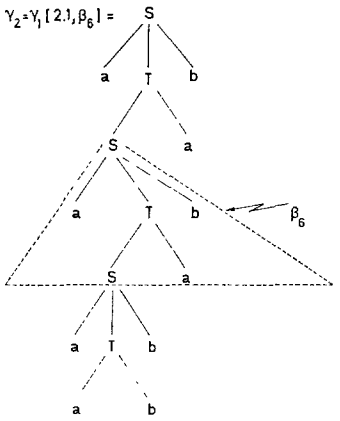
$\beta_5 =$ 

$\beta_6 =$ 

Let $\gamma_0 = \alpha_1 =$ 

Then

$\gamma_1 = \gamma_0[2, \beta_5] =$ 



and so forth.

Let $\mathscr{G}_2 = (\mathscr{C}, \mathscr{A})$ where $\mathscr{C} = \{\alpha_1\}$, and $\mathscr{A} = \{\beta_1, \beta_2\}$.

$$\alpha_1 = \begin{array}{c} S \\ | \\ e \end{array}$$

$$\beta_1 = \begin{array}{ccc} & S & \\ a & & T \\ & & b\ S\ c \end{array} \qquad \beta_2 = \begin{array}{ccc} & T & \\ a & & S \\ & & b\ T\ c \end{array}$$

Let $\mathscr{G}_3 = (\mathscr{C}, \mathscr{A})$ where $\mathscr{C} = \{\alpha_1\}$, and $\mathscr{A} = \{\beta_1\}$.

$$\alpha_1 = \begin{array}{c} S \\ | \\ e \end{array}$$

$$\beta_1 = \begin{array}{ccc} & S & \\ a & & S \\ & & b\ S\ c \end{array}$$

3. Relationship to Contex-Free and Context-Sensitive Languages

For a context-free grammar (cfg) G, let $\tau(G)$ be the set of all sentential derivation trees of G, i.e., trees whose roots are labeled with S, the initial symbol and whose terminal nodes are labeled with terminal symbols. Let $L(G)$ be the language of G. It is easy to characterize $\tau(G)$ in terms of a tag [10].

THEOREM 3.1. *For any* cfg G, *there is a tag* $\mathcal{G} = (\mathcal{C}, \mathcal{A})$ *such that* $\tau(\mathcal{G}) = \tau(G)$ *and* $L(\mathcal{G}) = L(G)$. *Further we can have the tag* $\mathcal{G}$ *such that the basic trees satisfy the restriction*: *If* $\alpha \in \mathcal{C}$ *then in any path in* α *no nonterminal appears more than once, and if* $\beta \in \mathcal{A}$ *then in any path in* β *no nonterminal appears more than once, not counting the nonterminal labeling the root node of* β.

In Example 2.1, $\mathcal{G}_1$ is a tag for the cfg $G = (V, \Sigma, P, S)$ where $V = \{S, T, a, b\}$, $\Sigma = \{a, b\}$, and $P = \{S \to aTb,\ T \to Sa,\ T \to TT,\ T \to ab,\ T \to ba\}$. $\mathcal{G}_2$ and $\mathcal{G}_3$ are not tag's for any cfg's.

For a given $\mathcal{G} = (\mathcal{C}, \mathcal{A})$, the size of a tree (in terms of the number of nodes) whose yield is a string, say $\omega \in L(\mathcal{G})$ is some linear combination of the sizes of trees in $\mathcal{C} \cup \mathcal{A}$. Thus we can construct in a straightforward manner a linear bounded automaton (not necessarily deterministic) which recognizes precisely $L(\mathcal{G})$. This argument works because the yield of an adjunct tree always has at least one terminal symbol (see Definition 2.7).[5]

THEOREM 3.2. *Let* $\mathcal{G} = (\mathcal{C}, \mathcal{A})$ *be a* tag. *Then the tal* $L(\mathcal{G})$ *is a context-sensitive language* (csl).

In Example 2.1, $L(\mathcal{G}_2)$ and $L(\mathcal{G}_3)$ are context-sensitive and not context-free. $L(\mathcal{G}_2) = \{\omega ec^n \mid n \geqslant 0$, ω is a string of a's and b's such that $\#a$'s $= \#b$'s $= n$, and for any initial proper substring of ω, $\#a$'s $\geqslant \#b$'s$\}$. $L(\mathcal{G}_2) \cap a^*b^*ec^* = \{a^nb^nec^n \mid n \geqslant 1\}$ which is context-sensitive but not context-free. Further $L(\mathcal{G}_3) = L(\mathcal{G}_2)$. The tag $\mathcal{G}_2$ satisfies the restriction in Theorem 3.1; however, the tag $\mathcal{G}_3$ does not satisfy it. This indicates that the restriction in Theorme 3.1 does not imply context-freeness of the corresponding language.

From the definition of tag and, in particular, the rule of adjunction it is easily seen that the well-known intercalation theorem (*uvwxy*-theorem) for cfl's can be easily extended to tal's.[6]

[5] If in Definition 2.7, we define an adjunct tree β such that $\gamma(\beta) \in \Sigma^* \times \Sigma^*$, then the above argument will not work. However, even in this case, Theorem 3.2 is true on account of the containment of tal' in indexed languages (see Section 4).

[6] Ogden [14] has presented a stronger version of the *uvwxy*-theorem. This theorem as well as the intercalation theorem in [10] can both be extended to tal's.

Hence, we can show that the context-sensitive language $\{a^n b^n c^n \mid n \geqslant 1\}$ is not a tal. An informal explanation of this fact is that in a tag we can keep track of more than two dependent counts but not of two or more dependent nestings.

Let $\mathscr{L}_{\text{csl}}$ $\mathscr{L}_{\text{cfl}}$ and $\mathscr{L}_{\text{tal}}$ be the classes of languages corresponding to context-sensitive grammars (csg) context-free grammars (cfg) and tag's, respectively.

COROLLARY 3.1. $\mathscr{L}_{\text{cfl}} \subsetneqq \mathscr{L}_{\text{tal}} \subsetneqq \mathscr{L}_{\text{csl}}$.

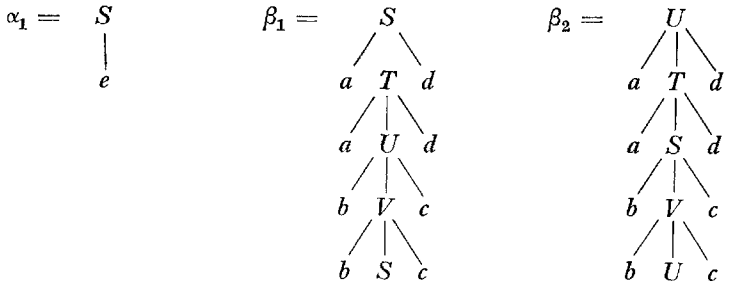
Remark. Let $\mathscr{L}_{\text{ind}}$ be the class of indexed languages of Aho [1]. Then, it can be shown that $\mathscr{L}_{\text{tal}} \subsetneqq \mathscr{L}_{\text{ind}}$ (see Section 4).

4. Relationship to Recognizable Sets and Context-Free Dendrogrammars

A set of (labeled) trees, R, is "recognizable" if three is a bottom-to-top tree automaton which recognizes precisely the trees in R [16, 17]. A set of trees is "local" if it is the set of all derivation trees of cfg.

Let R be a recognizable set of trees. Let $P(R)$ be the path set of R (see Definition 2.6). Rounds [16] has shown that if R is recognizable then $P(R)$ is regular (i.e., a finite state language). We now see why tag's are more powerful than cfg's. Let $\mathscr{G}$ be a tag and $\tau(\mathscr{G})$ the corresponding tree set. The path set of $\tau(\mathscr{G})$ is generated by rules of the form $X \to X\psi X$, or $X \to X\psi a$ where $X \in V - \Sigma$, $\psi \in (V - \Sigma)^*$, and $a \in \Sigma$. It is thus possible to get a nonregular path set.

EXAMPLE 4.1. Let $\mathscr{G} = (\mathscr{C}, \mathscr{A})$ where $\mathscr{C} = \{\alpha_1\}$, and $\mathscr{A} = \{\beta_1, \beta_2\}$.



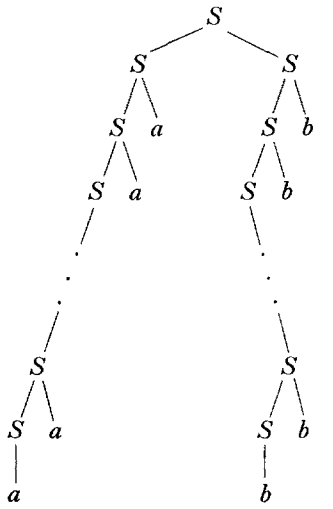
The path set $P(\tau(\mathscr{G}))$ is nonregular, for under the homomorphism, h which erases S and U, we have

$$h(P(\tau(\mathscr{G}))) \cap T^*V^*e = \{T^n V^n e \mid n \geqslant 0\},$$

which is nonregular.

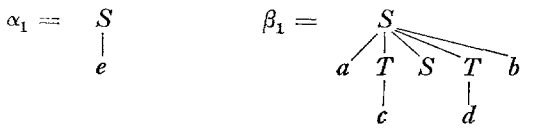
THEOREM 4.1. *The tree sets of* tag*'s and the recognizable sets are incomparable and their intersection properly contains the local sets.*

Proof. The set of a tag $\mathcal{G}$ for $L(\mathcal{G})$ which is not a cfl is clearly not recognizable. Now consider a set of trees of the following form.



i.e., a set of trees whose linear prefix representation is $\{S^{m+1}a^mS^nb^n \mid m, n \geqslant 1\}$. This set is recognizable but it is not a tree set for any tag because in any tag for this tree set the adjuncts which allow us to iterate a and b, respectively, will be adjoinable to each other and thus, a's and b's will be mixed in the yield string.[7]

Local sets are recognizable. Also they correspond to cfg's. They are clearly tree sets of tag's (see Theorem 3.1). Thus local sets are contained in the intersection of recognizable sets and tree sets of tag's. The containment is proper as can be seen from the following example. Let $\mathcal{G} = (\mathcal{C}, \mathcal{A})$ be a tag where $\mathcal{C} = \{\alpha_1\}$, and $\mathcal{A} = \{\beta_1\}$.



$\tau(\mathcal{G})$ is recognizable but it is not local. ∎

Remark. Although recognizable sets are closed under boolean operations, tree sets of tag's are not. This follows from the fact that tree sets of tag's are not closed under union.

[7] See Section 8 for a variant of tag which captures this recognizable set.

Let $\mathcal{G}_1 = (\mathcal{C}_1, \mathcal{A}_1)$ where $\mathcal{C}_1 = \{\alpha_1\}$, and $\mathcal{A}_1 = \{\beta_1\}$.

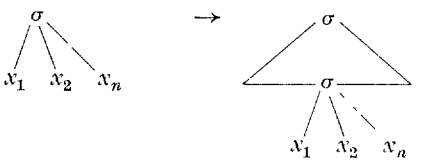
and $\mathcal{G}_2 = (\mathcal{C}_2, \mathcal{C}_2)$ where $\mathcal{C}_2 = \{\alpha_1\}$, and $\mathcal{A}_2 = \{\beta_1\}$.

The set of trees $\tau(\mathcal{G}_1) \cup \tau(\mathcal{G}_2)$ is not a tree set for any tag, since there is no way to keep the a's and c's separate.

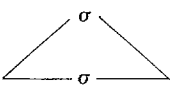
It is easy to show that the tree adjunct languages (tal's) are closed under union, set product, and Kleene star.

For some additional results concerning recognizable sets and tree sets of tag's, see Section 5 (Theorem 5.1) and Section 6 (Theorem 6.3, Corollary 6.1).

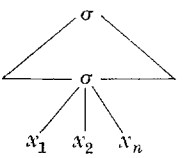
A tag can be regarded as a restricted case of the context free dendrogrammar of Rounds [16]. Corresponding to an adjunction rule, we will have a rule of the context free dendrogrammar of the following form (adopting Rounds' notation).



where σ is a nonterminal,



is a tree $\tau_\Sigma(\phi)$, and



is a tree in $\tau_\Sigma(x_1 x_2, \ldots, x_n)$ with indices $x_1, x_2, \ldots, x_n$.

Let "proj-tag sets" be the set of all tree sets obtained from tree sets of tags by

projection.[8] Then we have the following relationship among the various tree sets ($\rightarrow$ denotes projection).

$$\begin{array}{ccccc} \text{local sets} & & \subsetneq & & \text{tag sets} \\ \downarrow & & & & \downarrow \\ \text{recognizable sets} & & \subsetneq & \text{proj-tag sets} & \subsetneq \text{ context free dendrolanguages} \end{array}$$

Since the yields of context-free dendrolanguages are precisely the indexed languages [16], it follows that $\mathscr{L}_{\text{tal}} \subsetneq \mathscr{L}_{\text{ind}}$. Proper containment follows from the fact that $L = \{a^n b^n c^n \mid n \geqslant 1\}$ is an indexed language but not a tal.

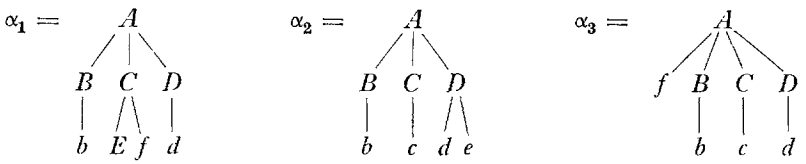
5. Simple Tag's

When measuring the depth of a tree we will ignore the branches between the terminal and the preterminal nodes. The motivation for this is that these correspond to terminal rules ($A \rightarrow \alpha$ is a terminal rule if $A \in V - \Sigma$ and $\alpha \in \Sigma^*$) and terminal rules are used to insert lexical items (e.g., as in $N \rightarrow$ *John*) or fixed strings of lexical items. These rules are also called lexical rules. It is clear that lexical rules do not contribute to hierarchical constituent structure.

We now introduce simple tag's which are a subclass of tag's. A simple tag has, in a sense, a minimal hierarchical structure.

Definition 5.1. A tree α is "simple" iff the depth of α is 1, ignoring the branches between terminal and preterminal nodes.

Example 5.1.



$(A, B, C, D, E \in V - \Sigma;\ b, c, d, e, f \in \Sigma)$

α_1 is not simple but α_1 and α_2 are simple.

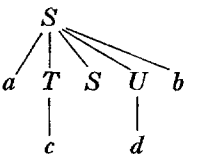
[8] A set of trees τ_1 over an alphabet V_1 is a "projection" of set of trees τ_2 over an alphabet V_2 if τ_1 is obtained from τ_2 by relabeling nodes, through an onto mapping $\pi = V_2 \rightarrow V_1$, which changes alphabets [17]. τ_2 is said to be the "inverse projection" of τ_1 if it is the largest set of trees with labels in V_2 whose projection is τ_1.

DEFINITION 5.2. A tag $\mathcal{G} = (\mathcal{C}, \mathcal{A})$ is "simple" iff all basic trees of $\mathcal{G}$ are simple.

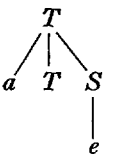
EXAMPLE 5.2. $\mathcal{G} = (\mathcal{C}, \mathcal{A})$ where $\mathcal{C} = \{\alpha_1\}$, and $\mathcal{A} = \{\beta_1, \beta_2\}$.

$\alpha_1 =$

$\beta_1 =$



$\beta_2 =$

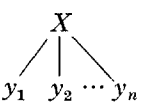


Although tag's in general are more powerful than context-free grammars, simple tag's are not.

THEOREM 5.1. *Let* $\mathcal{G} = (\mathcal{C}, \mathcal{A})$ *be a simple* tag. *Then the tree set of* $\mathcal{G}$, $\tau(\mathcal{G})$ *is recognizable.*[9]

Proof. We can construct a (nondeterministic) bottom-up tree automaton [17, 18] which recognizes precisely $\tau(\mathcal{G})$, as follows.

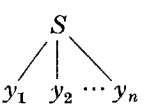
State set: $\{q_0\} \cup \{q_x \mid x \in \Sigma\} \cup \{q_f\} \cup \{q(X, y_1, y_2, \ldots, y_n) \mid X \in V - \Sigma;$ $y_1, y_2, \ldots, y_n \in \Sigma$ such that



is a subtree of some basic tree of $\mathcal{G}$.$\}$

Initial state set: $\{q_0\}$

Final state set: $\{q_f\} \cup \{q_{(S, y_1, y_2, \ldots, y_n)} \mid S \in V - \Sigma$, the distinguished symbol; $y_1, y_2, \ldots, y_n \in \Sigma$ such that



is a center tree of $\mathcal{G}$.$\}$

[9] This theorem holds even if $\mathcal{C}$ is not simple.

Transitions:

(i) $x \to \cdot\, q_x, \qquad x \in \Sigma$

q_0

(ii) $X \to \cdot\, q_{(X,y_1,y_2,\ldots,y_n)}$

$q_{y_1}\ q_{y_2}\ \cdots\ q_{y_n}$

$$X \in V - \Sigma,\ q_{y_j} \in \{q_x \mid x \in \Sigma\},\ j = 1, 2, \ldots, n$$

(iii) We say that

$q_{y_1}\ q_{y_2}\ \cdots\ q_{y_n}$

"corresponds" to a basic tree of $\mathcal{G}$ if, under the mapping $q_x \to x$, $x \in \Sigma$, and

$q_{(Y,y_1,y_2,\ldots,y_n)} \to$ $y_1\ y_2\ \cdots\ y_n$ $\qquad Y \neq X$

and

$$\cdot\, q_{(Y,y_1,y_2,\ldots,y_n)} \to \cdot\, X, \qquad Y = X$$

the tree

$q_{y_1}\ q_{y_2}\ \cdots\ q_{y_n}$

maps into a basic tree of $\mathcal{G}$.

If

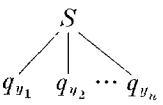
$q_{y_1}\ q_{y_2}\ \cdots\ q_{y_n}$

corresponds to a basic tree and $q_{y_k} = q_{(X,z_1,z_2,\ldots,z_m)}$ for some k, then

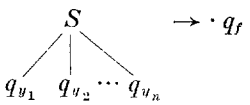
$\to \cdot\, q_{(X,z_1,z_2,\ldots,z_m)}$

$q_{y_1}\ \cdots\ q_{y_2}\ q_{y_n}$

If



corresponds to a center tree of $\mathcal{G}$, then



It is easy to check that the bottom-up tree automaton described above recognizes precisely $\tau(\mathcal{G})$. ∎

COROLLARY 5.1. *Let $\mathcal{G}$ be a simple* tag. *Then the path set $P(\tau(\mathcal{G}))$ is regular.*

COROLLARY 5.2. *The language $L(\mathcal{G})$ of a simple* tag *$\mathcal{G}$ is context-free. (A tal which is the language of a simple tag will be called a simple tal; thus a simple tal is context-free.)*

COROLLARY 5.3. *The equivalence problem for tree sets of simple* tag*'s is decidable.*

From Corollary 5.2, we see that if a cfl L is a simple tal then it can be "described" with a minimal hierarchical structure.[10] We now show, however, that not every cfl is a simple tal.

DEFINITION 5.2. A terminal symbol is said to be of "bounded occurrence" in L iff the number of times it occurs in any word in the language L is bounded by a constant.

DEFINITION 5.3. A nonterminal symbol, say X is self-dominating iff there is an adjunct tree whose root has label X.

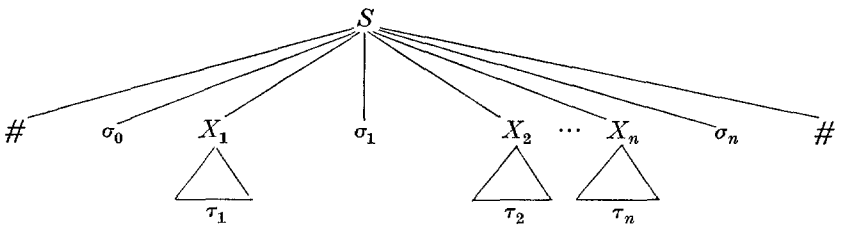
PROPOSITION 5.1. *For any* tag *$\mathcal{G}$ there is a weakly equivalent* tag *$\mathcal{G}'$ (i.e., $L(\mathcal{G}) = L(\mathcal{G}')$) such that the set of nonterminals of $\mathcal{G}'$ (i.e., $V' - \Sigma'$) is just the set $\{S\} \cup \{X \mid X$ is self-dominating$\}$. Note that S may or may not be self-dominating, but, even if it is not, it cannot be eliminated.*

PROPOSITION 5.2. *If a terminal symbol is of bounded occurrence, then it cannot occur at the leaves of (i.e., at the front of) an adjunct tree (this is so because an adjunct tree can be adjoined arbitrarily many times). Thus a symbol of bounded occurrence can occur only at the leaves of a center tree.*

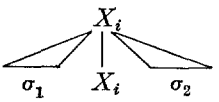
[10] These results can be considered as belonging to the area of syntactic complexity. For some other results in this area see (Bar-Hillel, *et al.* [2]).

LEMMA 5.1. *Let* $L = \{\#(^n a(^m)^m a)^n \# \mid m, n \geqslant 1\}$. L *is a* cfl *but not a simple* tal.

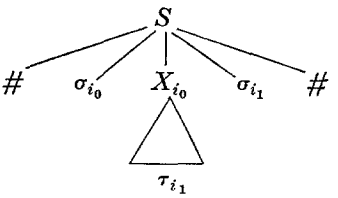
Proof. Let L be simple tal, i.e., $L = L(\mathscr{G})$ where $\mathscr{G} = (\mathscr{C}, \mathscr{A})$ is a simple tag. The symbols a and $\#$ are of bounded occurrence. Hence, center trees of $\mathscr{G}$ must be of the form:



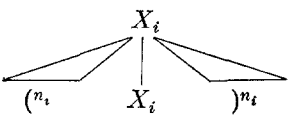
where σ_i, $\tau_i \in \{a, (,)\}^*$, $i = 0, 1, 2, \ldots, n$, $X_i \in V - \Sigma$, $i = 1, 2, \ldots, n$. Further, there cannot be any adjunct trees with root labeled S. We now observe that L is a parenthesis language; hence in any adjunct tree



we must have at least one '(' in σ_1 and at least one ')' in σ_2 (see Levy [11]). In fact, $\sigma_1 = (^k$, and $\sigma_2 =)^k$, for some $k > 1$. At most one X_i can occur in a center tree, since otherwise a ( ) ( ) parentheses structure could be generated; therefore, any tree in $\mathscr{C}$ must be of the form:



and any tree in $\mathscr{A}$ must be of the form:



In a center tree, either both occurrences of a are in τ_{i_1}, or one is in σ_{i_0} and the other in σ_{i_1}. Hence, for any tree derived from a given center tree, either m or n is fixed. Hence, we cannot have a simple tag for L. ∎

Thus, we have the following.

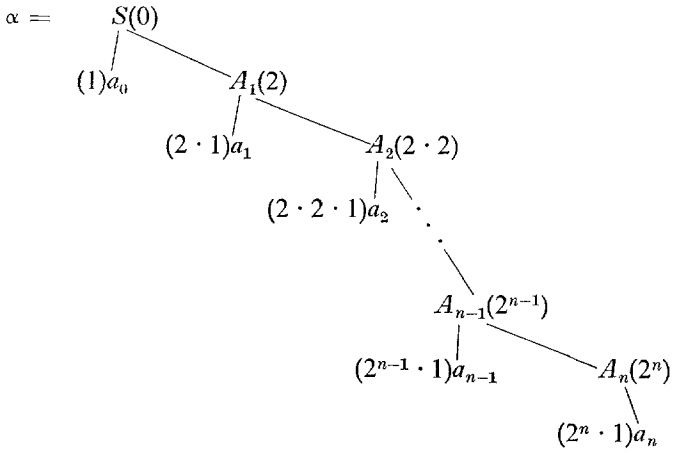
THEOREM 5.2. *The class of simple* tal*'s is properly contained in* $\mathscr{L}_{\text{cfl}}$.

It is easily shown that an infinite hierarchy of languages is obtained, depending on the maximum allowable depths of the basic trees in a tag.

Although not all cfl's are simple tals, cfl's which are finite state languages (regular) are simple tal's.

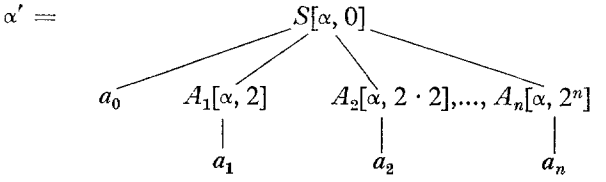
THEOREM 5.3. *If* L *is a regular language then there is a* tag $\mathscr{G}$ *such that* $L(\mathscr{G}) = L$ *and* $\mathscr{G}$ *is simple.*

Proof. Let L be a regular language and let G be a cfg such that $L(G) = L$. Since L is regular, we can take G to be a right linear grammar, i.e., the rules of G are of the form: $A \to a$, $A \to aB$, where $A, B \in V - \Sigma$. From G we can obtain a tag $\mathscr{G} = (\mathscr{C}, \mathscr{A})$ such that $L(\mathscr{G}) = L(G) = L$. The basic trees of $\mathscr{G}$ may not be all simple, however. A typical center tree, say α, will be of the form:



$S, A_1, \ldots, A_n \in V - \Sigma$; $a_0, a_1, \ldots, a_n \in \Sigma$. At each node, the address of that node is indicated inside parentheses. 2^m means $2 \cdot 2 \cdot 2 \cdots \cdot 2$ (m times). From Theorem 3.1, we know that $S, A_1, \ldots, A_n$ will be all distinct.

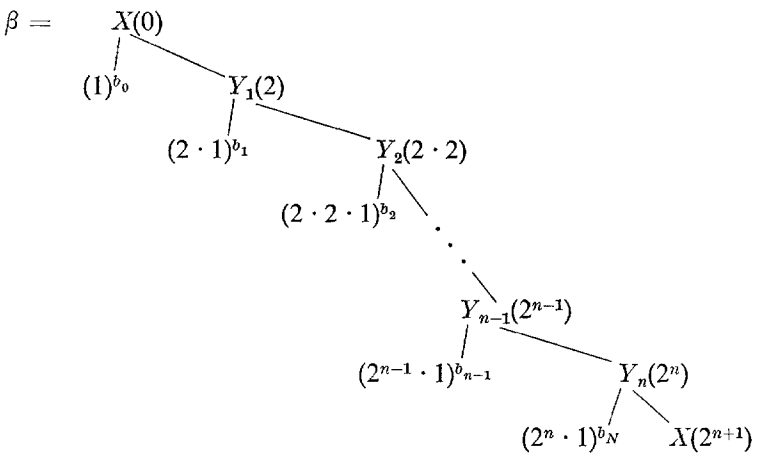
We will now transform α into α' in the following manner.



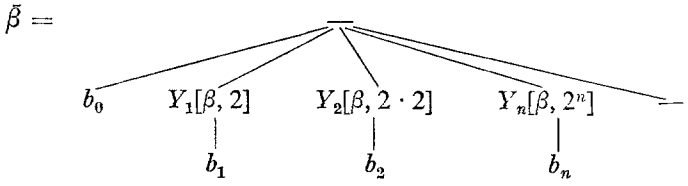
where $S[\alpha, 0]$, $A_1[\alpha, 2]$ $A_2[\alpha, 2 \cdot 2], \ldots, A_n[\alpha, 2^n]$ are new nonterminals not in $V - \sum$. A nonterminal, e.g., $A_2[\alpha, 2 \cdot 2]$ essentially records the information about a particular

occurrence of A_2 in the tree α at the address $2 \cdot 2$. Note that α' is a simple tree; hence, we call α' a "simplification" of α. Let $\mathscr{C}' = \{\alpha' \mid \alpha' \text{ is a simplification of some } \alpha \in \mathscr{C}\}$.

A typical adjunct tree, say β, in $\mathscr{G}$ will be of the following form.



X, Y_1, Y_2,..., $Y_n \in V - \Sigma$; $b_0 b_2$,..., $b_n \in \Sigma$. Y_1, Y_2,..., Y_n will be all distinct and also distinct from X (see Theorem 3.1). We call β an X-type tree. A tree $\bar{\beta}$ (called a "simple skeleton" of β) is obtained as follows. $\bar{\beta}$ is a simplification of β except that the root node in $\bar{\beta}$ and the terminal node in $\bar{\beta}$ corresponding to the address 2^{n+1} in β are not labeled (i.e., left blank). Since β is an X-type adjunct tree, we call $\bar{\beta}$ an X-type simple skeleton.



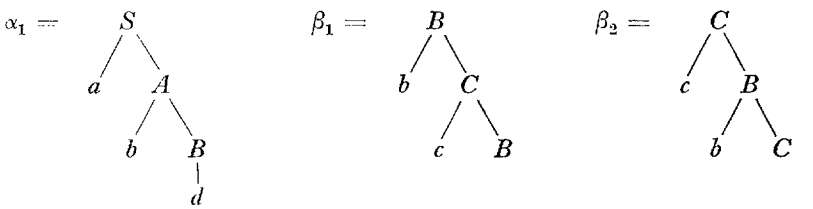
where $Y_1[\beta, 2]$, $Y_2[\beta, 2 \cdot 2]$,..., $Y_n[\beta, 2^n]$ are new nonterminals not in $V - \Sigma$.—denotes an unlabeled node. Note that $\bar{\beta}$ is a simple tree. Let $\bar{\mathscr{A}} = \{\bar{\beta} \mid \bar{\beta} \text{ is a simple skeleton of some } \beta \in \mathscr{A}\}$.

Let U be the set of all the new nonterminals introduced in constructing $\mathscr{C}'$ from $\mathscr{C}$ and $\bar{\mathscr{A}}$ from $\mathscr{A}$. A nonterminal in U, say $X[\gamma, p]$, where $X \in V - \Sigma$, γ is a basic tree and p is an address in γ, will be called an X-type nonterminal. We now obtain a set of simple adjunct trees, $\mathscr{A}'$, as follows. Let $\bar{\beta} \in \bar{\mathscr{A}}$ be an X-type simple skeleton. For each X-type nonterminal, say $X[\gamma, p]$ we obtain an adjunct tree, say β' from $\bar{\beta}$ by labeling the unlabeled nodes in $\bar{\beta}$ by $X[\gamma, p]$. We will say that β' is obtained by "filling in" $\bar{\beta}$. Note that the filling in is always with nonterminals of the appropriate type

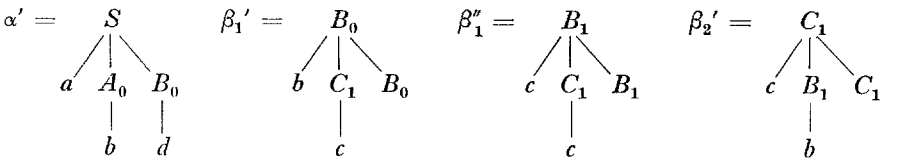
depending on the type of $\bar{\beta}$. Let $\mathcal{A}' = \{\beta' \mid \beta'$ is obtained by filling in some $\bar{\beta} \in \bar{\mathcal{A}}\}$.

Now let $\mathcal{G}' = (\mathcal{C}', \mathcal{A}')$ be a tag where $\mathcal{C}'$ and $\mathcal{A}'$ are constructed as above. $\mathcal{G}'$ is a "simple" tag. We will now show that $L(G) = L(\mathcal{G}')$. From the construction of $\mathcal{G}'$ it is clear that $L(G) \subset L(\mathcal{G}')$. We now want to show that $L(\mathcal{G}') \subseteq L(G)$. This is easily shown by induction on the number of steps in the derivation in $\mathcal{G}'$. Let $x \in L(\mathcal{G}')$, $\gamma' \in \tau(\mathcal{G}')$ such that $Y(\gamma') = x$, and let γ' be derived in one step, i.e., γ' must be a center tree, i.e., a tree in $\mathcal{C}'$. Clearly, there is some center tree, say γ, in $\mathcal{C}$ such that $Y(\gamma) = Y(\gamma') = x$. Assume that if γ' is derived in n steps in $\mathcal{G}'$ then there is a derivation in n steps in $\mathcal{G}$ of a tree γ such that $Y(\gamma') = Y(\gamma)$. Now let γ' be derived in $\mathcal{G}'$ in $n+1$ steps, i.e., there is a derivation $\gamma_1' \vdash_{\mathcal{G}'} \gamma_2' \vdash_{\mathcal{G}'} \ldots, \vdash_{\mathcal{G}'} \gamma_n' \vdash_{\mathcal{G}'} \gamma_{n+1}' = \gamma'$ where $\gamma_1' \in \mathcal{C}'$. From the inductive hypothesis there is a derivation $\gamma_1 \vdash_{\mathcal{G}} \gamma_2, \ldots, \vdash_{\mathcal{G}} \gamma_n$, where $\gamma_1 \in \mathcal{C}$, such that $Y(\gamma_n') = Y(\gamma_n)$. (In fact, we have $Y(\gamma_1') = Y(\gamma_1)$, $Y(\gamma_2') = Y(\gamma_2), \ldots, Y(\gamma_n') = Y(\gamma_n)$.) Let γ_{n+1}' be derived from γ_n' by adjoining an adjunct tree, say, $\beta' \in \mathcal{A}'$ to γ_n' at address p, i.e., $\gamma_{n+1}' = \gamma_n'[p, \beta']$. From the construction of $\mathcal{G}'$ it follows that there is an address q in γ_n such that $\gamma_n(q) = \gamma_n'(p) = X$, say, and further there is an adjunct tree, say $\beta \in \mathcal{A}$ such that β is an X-type adjunct tree, $\gamma_{n+1} = \gamma_n[q, \beta]$, and $Y(\gamma_{n+1}) = Y(\gamma_{n+1}')$. Hence, $L(\mathcal{G}') \subseteq L(G)$ and therefore $L(G) = L(\mathcal{G}')$. ∎

EXAMPLE 5.3. Let $G = (V, \Sigma, P, S)$ be a cfg, where $\Sigma = \{a, b, c, d\}$, $V - \Sigma = \{S, A, B, C\}$, and $P = \{S \to aA, A \to bB, B \to bC, C \to cB, B \to d\}$. G is a right-linear cfg and hence $L(G)$ is regular. A corresponding tag is $\mathcal{G} = (\mathcal{C}, \mathcal{A})$ where $\mathcal{C} = \{\alpha_1\}$, and $\mathcal{A} = \{\beta_1, \beta_2\}$.



This tag is not simple. However, the following is a simple tag for $L(G)$. $\mathcal{G}' = (\mathcal{C}', \mathcal{A}')$ where $\mathcal{C}' = \{\alpha'\}$, and $\mathcal{A} = \{\beta_1', \beta_1'', \beta_2'\}$.



A_0, B_0, B_1 and C_1 are new nonterminals not in $V - \Sigma$. $L(G) = L(\mathcal{G}')$. From Theorem 5.3 and Corollary 5.2 we have the following.

COROLLARY 5.4. *Let L be a regular language. Then there is a* cfg $G = (V, \Sigma, P, S)$ *such that $L(G) = L$ and the rules of G are of the form $X \to a$, $X \to a\psi X$ where $X \in V - \Sigma$, $a \in \Sigma$, and $\psi = \epsilon$, the null string or $\psi = Y_1 Y_2 \cdots Y_n$, $Y_k \in V - \Sigma$, $k = 1, 2, \ldots, n$ such that for $i \neq j$, $Y_i \neq Y_j$ and $Y_i \neq X$, $i, j = 1, 2, \ldots, n$.*

6. LINEAR TAG'S

DEFINITION 6.1. A tree is "linear" iff at any depth there is at most one nonterminal. (Alternatively, a tree is linear iff for any pair of nonterminals, say X_1, X_2, in the tree, either X_1 dominates X_2 or X_2 dominates X_1 where A dominates B means that there is a path from A to B.) A tag $\mathcal{G} = (\mathcal{C}, \mathcal{A})$ is a "linear" tag iff all basic trees of $\mathcal{G}$ are linear. A language L is a "linear tal" iff $L = L(\mathcal{G})$ for some linear tag $\mathcal{G}$.

EXAMPLE 6.1. In Example 2.1, $\mathcal{G}_2$ and $\mathcal{G}_3$ are linear tag's, but $\mathcal{G}_1$ is not. In Example 5.3, $\mathcal{G}$ is a linear tag but $\mathcal{G}'$ is not.

THEOREM 6.1. *Linear* cfl's $\subsetneq$ *Linear* tal's. *Linear* tal's *are not comparable with* cfl's.

Proof. If L is a linear cfl then L has a linear cfg, say $\mathcal{G}$. Then every derivation tree in G is linear; hence, linear cfl's $\subset$ linear tal's. The language $L(\mathcal{G}_2)$ of $\mathcal{G}_2$ in Example 6.1 is not a linear cfl (in fact, it is a csl). Therefore, linear cfl's $\subsetneq$ linear tal's.

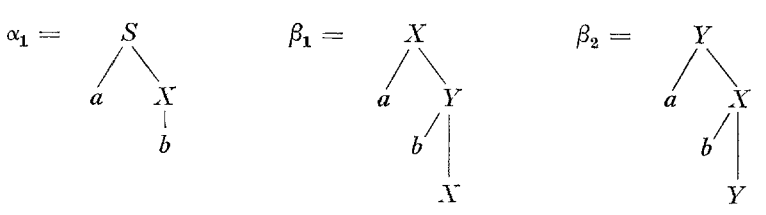
Let $L = \{\# a^m \# b^m \# c^n \# d^n \# \mid m, n \geqslant 1\}$. L is cfl. We will now show that L is not a linear tal. The $\#$ is a symbol of bounded occurrence (see Definition 5.2) and thus it can only occur on the leaves of a center tree. If L has a linear tag $\mathcal{G}$, then all trees in $\tau(\mathcal{G})$ (which include the center trees of $\mathcal{G}$) are linear. Now any adjunct tree whose yield has a in it must have the yield of the form $a^k X b^k$ where $X \in V - \Sigma$, otherwise, we can easily generate words not in L. We will call X an (a, b) variable. Similarly, any adjunct tree whose yield has a c in it must have the yield of the form $c^s Y d^s$ where $Y \in V - \Sigma$. We will call Y a (c, d) variable. The adjunct trees must be either of these two forms and the two sets of (a, b) and (c, d) variables must be disjoint. Further, in any tree in $\tau(\mathcal{G})$ it is not possible for an (a, b) variable to dominate a (c, d) variable or vice versa (see Definition 6.1). But since m, n are unbounded there must be trees in the tree set of any tag for L with both (a, b) and (c, d) variables without one dominating the other; hence the tag must be nonlinear. Thus linear tal's are not comparable with cfl's. ∎

DEFINITION 6.2. A tree is right-linear (left-linear) iff it is linear and at any depth the nonterminal is the rightmost (leftmost) symbol at that depth. A tag $\mathcal{G}$ is right-linear (left-linear) iff all the basic trees of $\mathcal{G}$ are right-linear (left-linear). A tag $\mathcal{G}$ is one-sided linear iff it is either right-linear or a left-linear tag. A language L is one-sided linear iff there is a one-sided linear tag $\mathcal{C}$ such that $L(\mathcal{G}) = L$.

THEOREM 6.2. *Regular languages* $\subsetneq$ *one-sided linear* tal's $\subsetneq$ cfl's.

Proof. By the technique of "simplification" in the proof of Theorem 5.5, it can be shown that if $\mathcal{G}$ is a one-sided linear tag then there is a simple tag $\mathcal{G}'$ such that $L(\mathcal{G}) = L(\mathcal{G}')$. $\mathcal{G}'$ is not necessarily a one-sided tag, however. By Corollary 5.2, the language of a simple tag is a cfl. Hence, one-sided linear tal's $\subset$ cfl's. Proper containment follows from the example in Theorem 6.1.

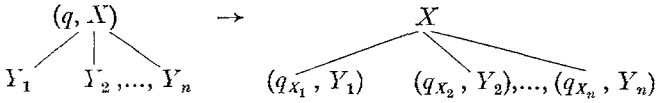
If L is a regular language then L has right-linear (left-linear) cfg and hence a one-sided linear tag. Thus regular languages $\subset$ one-sided linear tal's. Proper containment follows from the fact that $L_2 = \{w \mid w \in \{a, b\}^*$ and # a's (i.e., the number of a's) in W = # b's in w and in any initial proper substring of w, # a's > # b's$\}$ is a one-sided linear tal but not a regular language. The fact that L_2 is not regular is obvious. The following is a one-sided linear tag for L_2. Let $\mathcal{G} = (\mathcal{C}, \mathcal{A})$ where $\mathcal{C} = \{\alpha_1\}$, and $\mathcal{A} = \{\beta_1, \beta_2\}$.



Then $L(\mathcal{G}) = L_2$. ∎

THEOREM 6.3. *Let* $\mathcal{C}$ *be a one-sided linear* tag. *If* $\tau(\mathcal{G})$ *is a recognizable set then* $L(\mathcal{G})$ *is regular.*

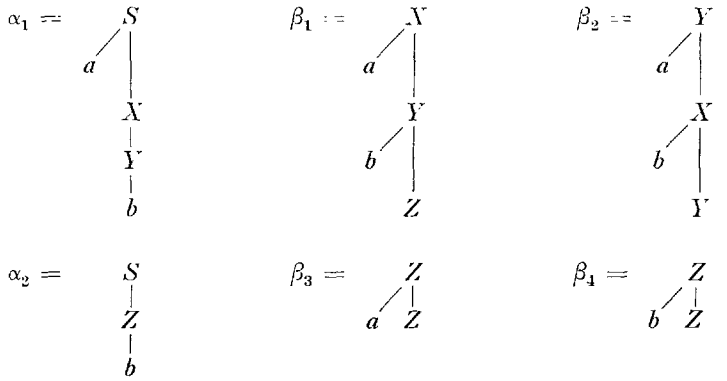
Proof. We can construct a deterministic top–down automaton which will accept precisely an inverse projection of $\tau(\mathcal{G})$ ([13] and Theorem 2.2). The rules of the automaton are of the form:



where $X \in V - \Sigma$, $Y_1, Y_2, \ldots, Y_n \in V$, and q, $q_{X_1}, q_{X_2}, \ldots, q_{X_n}$ are states. Then $(q, X) \rightarrow (q_{X_1}, Y_1)\ (q_{X_2}, Y_2), \ldots, (q_{X_n}, Y_n)$ will be a rule of a cfg, say G, for $L(\mathcal{G})$ where (q, X) will be a nonterminal of G and $(q_{X_1}, Y_1), (q_{X_2}, Y_2), \ldots, (q_{X_n}, Y_n)$ will be terminal or nonterminal symbols of G. If $\mathcal{G}$ is a one-sided linear tag then all trees in $\tau(\mathcal{G})$ will also be one-sided linear; hence the rules of G will be one-sided linear (i.e., either all right-linear or all left-linear). $L(\mathcal{G})$ is regular. ∎

COROLLARY 6.1. *If the tree set* $\tau(\mathcal{G})$ *of a one-sided linear* tag $\mathcal{G}$ *is such that* $L(\mathcal{G})$ *is not regular then* $\tau(\mathcal{G})$ *is not recognizable.*

The converse of Corollary 6.1 is not true, however. Let $\mathcal{G} = (\mathcal{C}, \mathcal{A})$ where $\mathcal{C} = \{\alpha_1, \alpha_2\}$, and $\mathcal{A} = \{\beta_1, \beta_2\}$.



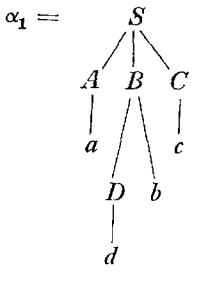
$\mathcal{G}$ is a one-sided linear tag. $L(\mathcal{G})$ is regular, but $\tau(\mathcal{G})$ is not recognizable.

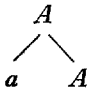
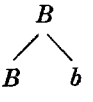
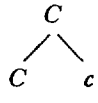
7. A Generalization of Tree Adjunct Grammar

We will generalize a tag so that it will be possible to adjoin one or more adjunct trees simultaneously during each adjunction operation.

Definition 7.1. "A simultaneous tag (stag)," $\mathcal{G}$ is a pair $\mathcal{G} = (\mathcal{C}, \mathcal{A})$ where $\mathcal{C}$ is a finite set of center trees and $\mathcal{A}$ is a finite set of "adjunct sets" where each adjunct set is a finite set of adjunct trees.

Example 3.1. Let $\mathcal{G} = (\mathcal{C}, \mathcal{A})$ where $\mathcal{C} = \{\alpha_1\}$, and $\mathcal{A} = \{\beta_1, \beta_2, \beta_3\}$.



$\beta_1 = \{\beta_{11}, \beta_{12}, \beta_{13}\}$, $\beta_{11} =$  $\beta_{12} =$  $\beta_{13} =$ 

$\beta_2 = \{\beta_{21}, \beta_{22}\}$, $\beta_{21} =$ 

$\beta_{22} =$ 

$\beta_3 = \{\beta_{31}\}$, $\beta_{31} =$ 

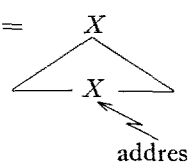
β_1, β_2, and β_3 are "adjunct sets." β_{11}, β_{12}, and β_{13} will be called components of β_1. Similarly, β_2 has β_{21} and β_{22} as its components. β_3 has only one component, viz, β_{31}.

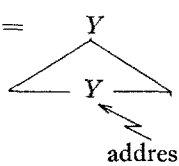
Let β_i be an adjunct set with k components, i.e., $\beta_i = \{\beta_{i1}, \beta_{i2}, \ldots, \beta_{ik}\}$, and let $\gamma \in \tau_\nu$. Let $\rho_1, \rho_2, \ldots, \rho_k \in D_\gamma$, $\rho_i \neq \rho_j$, for $i \neq j$, and $\gamma(\rho_1) = \beta_{i1}(0)$, $\gamma(\rho_2) = \beta_{i2}(0), \ldots, \gamma(\rho_k) = \beta_{ik}(0)$. Then, speaking informally, we say that $\beta_i = \{\beta_{i1}, \beta_{i2}, \ldots, \beta_{ik}\}$ is "adjoinable" to γ at $\rho_1, \rho_2, \ldots, \rho_k$ and γ' is the tree obtained from γ by adjoining (according to Definition 2.8) β_{i1} at ρ_1, β_{i2} at $\rho_2, \ldots$, and β_{ik} at ρ_k, "simultaneously."

First, let us consider the case when $\rho_1, \rho_2, \ldots, \rho_k$ are mutually nondominating (i.e., for any pair, ρ_i, ρ_j, it is not the case that ρ_i dominates ρ_j or ρ_j dominates ρ_i). It is clear that the order of adjoining the components of β_i is irrelevant as long as β_{i1} is adjoined at ρ_1, β_{i2} is adjoined at $\rho_2, \ldots$, and β_{ik} is adjoined at ρ_k. But in order to give a precise definition for adjunction by using Definition 2.8 we will have to impose some ordering (arbitrary) and let us assume that β_{i1} is adjoined first, then β_{i2}, etc. Hence, γ' is obtained as follows.

$$\gamma_1' = \gamma[\rho_1, \beta_{i1}],\ \gamma_2' = \gamma_1'[\rho_2, \beta_{i2}], \ldots,\ \gamma_k' = \gamma_{k-1}'[\rho_k, \beta_{ik}] = \gamma'.$$

For the case when one node dominates another, say, ρ_r dominates ρ_s, we adjoin β_{is} at ρ_s first and then β_{ir} at ρ_r, i.e., we start adjoining at the lowest node first and work upwards. We may still speak informally of adjoining "simultaneously" because of Lemma 7.1 below. For simplicity let β_i have only two components, i.e., $\beta_i = \{\beta_{i1}, \beta_{i2}\}$.

Let $\beta_{i1} =$ 

and $\beta_{i2} =$ 

i.e., $(r_1, X) \in \hat{\beta}_{i1}$ and $(r_2, Y) \in \hat{\beta}_{i2}$. Then from Definitions 2.3 and 2.8, we have the next lemma.

LEMMA 7.1. *Let* $\beta_i = \{\beta_{i1}, \beta_{i2}\}$, $\gamma \in \tau_\nu$, *and* $\rho_1, \rho_2 \in D_\gamma$. *Let* $\rho_1 > \rho_2$, *i.e.,* ρ_2 *dominates* ρ_1. *Let* $\gamma_1' = \gamma[\rho_1, \beta_{i1}]$, $\gamma_2' = \gamma_1'[\rho_2, \beta_{i2}] = \gamma'$ *and* $\gamma_1'' = \gamma[\rho_2, \beta_{i1}]$, $\gamma_2'' = \gamma[\rho_1 \cdot r_2, \beta_{i2}] = \gamma''$. *Then* $\gamma' = \gamma''$, *i.e., the tree obtained by adjoining* β_{i1} *at* ρ_1

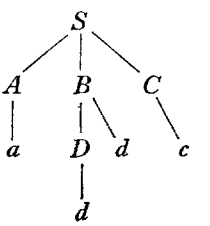
first and then adjoining β_{i2} at ρ_2 is the same as the tree obtained by adjoining β_{i2} at ρ_2 and then adjoining β_{i1} at $\rho_1 \cdot r_2$. (Note that the node at $\rho_1 \cdot r_2$ is the same node which previously had the address ρ_1 in γ.)

We can now define adjunction in a stag. Let $\beta_i = \{\beta_{i1}, \beta_{i2}, \ldots, \beta_{ik}\}$, $\gamma \in \tau_V$, $\rho_1, \rho_2, \ldots, \rho_k \in D_\gamma$, $\rho_i \neq \rho_j$, for $i \neq j$, and $\gamma(\rho_1) = \beta_{i1}(0)$, $\gamma(\rho_2) = \beta_{i2}(0), \ldots, \gamma(\rho k) = \beta_{ik}(0)$. Assume that $\rho_1, \rho_2, \ldots, \rho_k$ are ordered bottom to top and left to right[11] in the tree γ.

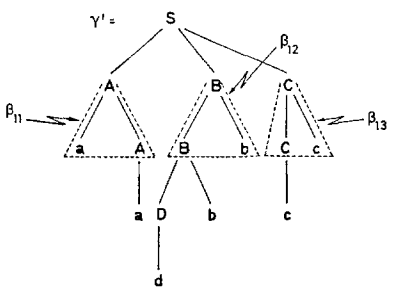
DEFINITION 7.2. If $\beta_i = \{\beta_{i1}, \beta_{i2}, \ldots, \beta_{ik}\}$ is adjoinable to γ at $\rho_1, \rho_2, \ldots, \rho_k$ then $\gamma' = \gamma[\rho_1, \rho_2, \ldots, \rho_k, \beta_i]$, the resultant tree, γ' is obtained from γ as follows. Let $\gamma_1' = \gamma[\rho_1, \beta_{i1}]$, $\gamma_2' = \gamma_1'[\rho_2, \beta_{i2}], \ldots, \gamma_k' = \gamma'_{k-1}[\rho_k, \beta_{ik}]$. Then $\gamma' = \gamma_k'$.

EXAMPLE 3.2. Consider the stag $\mathcal{G}$ of Example 3.1. Let

$\gamma = \alpha_1 =$

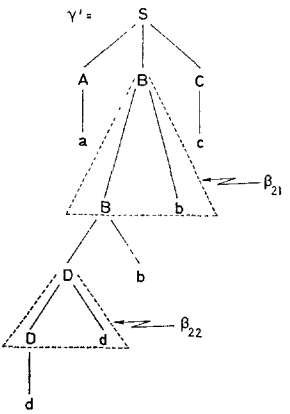


Let β_1 be adjoined to γ at $\rho_1(=1)$, $\rho_2(=2)$, and $\rho_3(=3)$. Then the resulting tree γ' is



Again, let $\gamma = \alpha_1$ as before. Let β_2 be adjoined to γ at $\rho_1(=2)$, and $\rho_2(=2 \cdot 1)$. Note that the node at ρ_1 dominates the node at ρ_2. Then the resulting tree γ' is (ρ_1 dominates ρ_2; hence β_{22} is adjoined first and then β_{21})

[11] That is, for $i = 2, \ldots, k$, either ρ_i dominates ρ_{i-1} or ρ_i is to the right of ρ_{i-1} in the tree γ.



Definition 2.9 can be easily extended to stag's. Let $\mathcal{G} = (\mathcal{C}, \mathcal{A})$ be a stag and let $\tau(\mathcal{G})$ be the "tree set" of $\mathcal{G}$ and $L(\mathcal{G})$ be the corresponding language ("simultaneous tree adjunct language," stal). Let $\mathcal{L}_{\text{stal}}$ be the class of languages corresponding to stag's. We then have the following.

THEOREM 7.1. $\mathcal{L}_{\text{cfl}} \subsetneq \mathcal{L}_{\text{tal}} \subsetneq \mathcal{L}_{\text{stal}} \subsetneq \mathcal{L}_{\text{csl}}$.

Proof. From Corollary 2.1 we have $\mathcal{L}_{\text{cfl}} \subsetneq \mathcal{L}_{\text{tal}} \subsetneq \mathcal{L}_{\text{csl}}$. Tag's are special cases of stag's; in a tag, each adjunct set has exactly one adjunct tree. Hence, $\mathcal{L}_{\text{tal}} \subsetneq \mathcal{L}_{\text{stal}}$. Given a stag $\mathcal{G}$, it is easy to construct a nondeterministic linear bounded automaton which recognizes precisely $L(\mathcal{G})$. Hence, $\mathcal{L}_{\text{stal}} \subseteq \mathcal{L}_{\text{csl}}$. It is also easily shown that $L = \{a^{n^2} \mid n \geq 1\}$ is not a stal by examining the growth rate of the lengths of strings in the language of a stag. Thus $\mathcal{L}_{\text{stal}} \subsetneq \mathcal{L}_{\text{csl}}$.

This argument works because the yield of each adjunct tree has at least one nonterminal in it (see Definition 2.7). If in Definition 2.7, we define an adjunct tree β such that $\gamma(\beta) \in \Sigma^* X \Sigma^*$, then, however, the above argument does not hold. We conjecture that even in this case $\mathcal{L}_{\text{stal}} \subsetneq \mathcal{L}_{\text{csl}}$. Note that the corresponding theorem for $\mathcal{L}_{\text{tal}}$ (Theorem 3.2) is true in this case (see footnote 4). ∎

Analogous to a "simple tag" (Definition 5.2), we can define a "simple stag" as a stag all of whose trees are simple (see Joshi [8] for an application of a simple stag).

8. A VARIANT OF A TAG

In Section 4, it was shown that the tree sets of tag's and recognizable sets are incomparable (Theorem 4.1). In this section, we will give a different formulation of tag's which has the advantage that many recognizable sets which are not tree sets

of tag's are captured in this new characterization. Whether or not all recognizable sets can be characterized in this way is still an open probem.

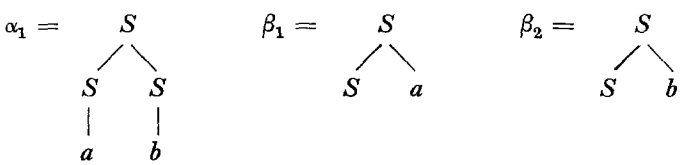
An adjunct tree β is adjoinable to a tree γ at address p if the label at p, i.e., $\gamma(p) = \beta(0)$. In the new formulation we will make adjunction depend not only on the identity of labels ($\gamma(p) = \beta(0)$) but also on the basic tree (i.e., in $\mathscr{C} \cup \mathscr{A}$) to which the node at p belongs. This characterization is a direct generalization of some of the basic ideas in Joshi *et al.* [11]. In order to distinguish this new formulation from the old one we will refer to the earlier formulation by tag_1 and the new one by tag_2.

DEFINITION 8.1. A tag_2 is a triple $\mathscr{G} = (\mathscr{C}, \mathscr{A}, J)$ where $\mathscr{C}$ is a finite set of center trees, $\mathscr{A}$ is a finite set of adjunct trees ($\mathscr{C} \cup \mathscr{A}$ is the set of basic trees), and J is a finite set of adjunction rules.[12]

An adjunction rule $u \in J$ is a triple $u = (\gamma_i, \gamma_j, \xi)$ where $\gamma_i \in \mathscr{C} \cup \mathscr{A}$, $\gamma_j \in \mathscr{A}$, and ξ is an address in γ_i. γ_i is called the host of u, γ_j, the adjunct of u, and ξ, the point of adjunction in the host. Of course, it is understood that $\gamma_i(\xi) = \gamma_j(0)$. Thus adjoinability of γ_j at ξ depends not only on the fact that the labels match but also on the fact that the node at ξ is a node in one of the basic trees as specified by u.

As before, $\tau(\mathscr{G})$ is a set of all trees derived from trees in $\mathscr{C}$. The definition of derivation is more complicated now. We will not give a detailed precise definition. It can be formulated along the same lines as in Joshi *et al.* [11]. We will illustrate the idea by means of an example.

EXAMPLE 8.1. Let $\mathscr{G} = (\mathscr{C}, \mathscr{A}, J)$ be a tag_2 where $\mathscr{C} = \{\alpha_1\}$, $\mathscr{A} = \{\beta_1, \beta_2\}$, and $J = \{u_1, u_2\}$.



$$u_1 = (\alpha_1, \beta_1, 1) \qquad u_2 = (\alpha_1, \beta_2, 2)$$

Note that although $\beta_1(0) = \beta_2(0) = \alpha_1(1) = \alpha_2(2)$, β_1 adjoins to α_1 only at address 1 and β_2 adjoins α_1 only at address 2. It is clear that by repeated applications of u_1 and u_2 (not necessarily the same number of times) one obtains a tree set whose linear prefix representation is $\{S^{m+1}a^mS^nb^n \mid m, n \geqslant 1\}$ (See the example in the proof of Theorem 4.1). This set is recognizable but it is not a tree set of tag_1 (Theorem 4.1), although as we see now, it is a tree set of tag_2. The following results are easily established.

[12] Such a tag is used for representing the base component of the transformational grammar in Joshi [9].

THEOREM 8.1. *Let* $\tau(\mathcal{G})$ *be a tree set of* $\mathcal{G}$ *where* $\mathcal{G}$ *is a* tag_1. *Then there is a* tag_2 $\mathcal{G}'$ *such that* $\tau(\mathcal{G}') = \tau(\mathcal{G})$.

THEOREM 8.2. *Let* proj-tag_2 *sets be the set of all tree sets obtained from tree sets of* tag_2 *by projection. Then (from Theorem A.*1 *and the diagram at the end of Section* 4), *Recognizable sets* $\subsetneq$ proj-tag_1 *sets* $\subsetneq$ proj-tag_2 *sets*

THEOREM 8.3. *There are recognizable sets which are tree sets of* tag_2 *but not of* tag_1.

Open problem. Let R be a recognizable set. Does there exist a tag_2 $\mathcal{G}$ such that $\tau(\mathcal{G}) = R$?

REFERENCES

1. A. V. AHO, Indexed grammars: an extension of the context-free grammars, *J. Assoc. Comput. Mach.* **15** (1968), 647–671.
2. Y. BAR-HILLEL, A. KASHER, AND E. SHAMIR, "Measures of syntactic complexity," Tech. Report No. 13, Applied Logic Branch, The Hebrew University of Jerusalem, Jerusalem, Israel, 1963.
3. W. S. BRAINERD, Tree generating regular systems, *Information and Control* **14** (1969), 217–231.
4. W. S. BRAINERD, Semi-Thue systems and representations of trees, *in* "Proceedings Tenth Annual IEEE Symposium on Switching and Automata Theory," Waterloo, Canada, 1969.
5. N. CHOMSKY, "Syntactic Structures," Monton, The Hague, Netherlands, 1957.
6. S. GORN, Explicit definitions and linguistic dominoes, *in* "Proceedings of the Systems and Computer Science Conference," University of Western Ontario, 1965, pp. 77–115.
7. Z. S. HARRIS, "Mathematical Structures of Language," Interscience, New York, 1968.
8. A. K. JOSHI, How much hierarchical structures is necessary for sentence description?, *in* "Transformationelle Analyse" (S. Plötz, Ed.), Athenaum–Verlag (Harcourt Brace), Frankfurt, 1972.
9. A. K. JOSHI, "A Class of Transformational Grammars" (M. Gross, M. Halle, and M. P. Schützenberger, Eds.), Monton, The Hague, Netherlands, 1973.
10. A. K. JOSHI AND M. TAKAHASHI, "A characterization of the derivation trees of a context-free grammar and an intercalation theorem," Tech. Report, The Moore School of Electrical Engineering, University of Pennsylvania, 1971.
11. A. K. JOSHI, S. KOSARAJU, AND H. M. YAMADA, String adjunct grammars: Parts I and II, *Information and Control* **21** (1972), 93–116; 235–260.
12. L. S. LEVY, Tree adjunct, parenthesis, and distributed adjunct grammars, "Proceedings of the International Symposium on Theory of Machines and Computations," Haifa, 1971, pp. 127–143, Academic Press, New York, 1971.
13. L. S. LEVY AND A. K. JOSHI, Some results in tree automata, *Math. Systems Theory* **6** (1972).
14. W. OGDEN, A helpful result for proving inherent ambiguity, *Math. Systems Theory* **2** (1968), 191–194.
15. R. J. PARIKH, On context-free languages, *J. Assoc. Comput. Mach.* **13** (1969), 570–581.

16. W. C. ROUNDS, Mappings and grammars on trees, *Math. Systems Theory* **4** (1970), 257–287.
17. J. W. THATCHER, Characterizing derivation trees of a context-free grammar through a generalization of finite automata theory, *J. Comput. System Sci.* **1** (1967), 317–322.
18. J. W. THATCHER, Generalized sequential transducer, *J. Comput. System Sci.* **4** (1970), 339–367.

Printed in Belgium