# SEQUENTIAL TREE AUTOMATA

by

Raymond T. Yeh and Peter Ng

Department of Computer Sciences
The University of Texas at Austin
Austin, Texas 78712
U.S.A.

March 14, 1974

## Abstract

The concept of a "tree-walking" automaton is introduced in this paper. It is shown that our model is equivalent in recognition power to generalized finite automaton [4] which recognizes trees by "running" down from the root to frontier of the tree in parallel. The correspondence between pushdown tree automata and context-free tree grammar is also demonstrated.

## I. Introduction

In this paper, we introduce another generalization of string recognition automata to tree recognition automata. In contrast to the concept of a generalized finite automaton [4], which recognizes a rooted, labelled tree by "running" down from the root to the frontier of the tree in parallel, our model walks on the tree and reads only one input symbol at a time. Yet, surprisingly enough, our model is equivalent to the parallel operating model just mentioned in recognition power. The novelty of our model lies in the fact that it has the ability to delete edges and hence is able to know whether it has traversed the whole input tree.

Once the concept of the tree walking automaton is introduced, the usual technique and theorems in string recognition automaton seems to hold and apply. In this paper, we will only concern ourself with the correspondence between context-free tree grammars and pushdown tree automata.

## II. Context-free Tree Grammar

The concept of regular and context-free grammars generating trees will be introduced in this section. Some preliminary notions about trees will first be introduced.

A <u>ranked alphabet</u> is a pair $(\Sigma, r)$, where $\Sigma$ is a finite set and $r$ is a function mapping $\Sigma$ into the set of non-negative integers. We set $\Sigma_n = r^{-1}(n)$. Usually, when there is no confusion, we will simply denote a rank alphabet $(\Sigma, r)$ by $\Sigma$. Let $X$ be a fixed countable set $\{x_1, x_2, \ldots\}$ such that $X \cap \Sigma_0 = \emptyset$. Then the set of trees $T_\Sigma(X)$, generated by a ranked alphabet $\Sigma$, indexed by $X$, is defined inductively as the smallest set containing $\Sigma_0 \cup X$ such that whenever $\sigma \in \Sigma_n$ and $t_1, t_2, \ldots, t_n \in T_\Sigma(X)$, then $\sigma(t_1 t_2 \ldots t) \in T_\Sigma(X)$. We will let $T_\Sigma$ to denote the set $T_\Sigma(\phi)$. If $t$ is a tree, then the frontier of $t$, denoted by $||t||$ , is the string composed of a left-to-right concatenation of labels of the leaves of $t$. A $\Sigma$-dendrolanguage is any subset of $T_\Sigma$.

A context-free grammar (cfg) $G$ is a 4-tuple $[N,T,\sigma,P]$, where $N$ and $T$ are finite sets called nonterminals and terminals, respectively. $\sigma \in N$ and $P$ is a finite set of ordered pairs $(A,x)$ called productions, where $A \in N$ and $X \in (N \cup T)^* - \{\Lambda\}$. A cfg is ranked if whenever $(A,x)$ and $(A,y)$ are in $P$, then $x$ and $y$ are of equal length.

If $G$ is a ranked cfg, then we may form a ranked alphabet $\Sigma$ by letting $\Sigma_0 = T$ and for $n > 0$, $\Sigma_n = \{A \in N \mid (A,x) \in P \text{ and length } (x) = n\}$.

Using this ranked alphabet, we can define the set of derivation trees $D_A^G$ associated with any $A \in \Sigma$ as follows:

a) $D_A^G = \{a\}$, for $a \in \Sigma_0$;

b) if $(A,x) \in P$ and $x = x_1 \ldots . x_n$ such that $t_i \in D_{x_i}^G$, $1 < i < n$, then $A(t_1 \ldots t_n) \in D_A^G$.

For any ranked cfg $G = [N,T,\sigma,P]$, we will refer to $D^G = D_\sigma^G$ as a derivation dendrolanguage.

In the sequel, we will assume that $\Sigma$ is a ranked alphabet.

Definition 1 - A context-free tree grammar (cftg) over a ranked alphabet $\Sigma$ is a quadruple $G = [N,\Sigma,\sigma,P]$, where

i) N is a finite set of nonterminals;

ii) $\Sigma$ is a finite set of terminals such that $N \cap \Sigma = \emptyset$;

iii) $\sigma \in N$ is the starting symbol.

iv) P is a finite set of rules called productions of the form $A \to \alpha$, where $A \in N$ and $\alpha \in T_\Sigma(N)$.

G is called a regular tree grammar (rtg) if every production $A \to \alpha$ satisfies the condition that at most one element of N occurs in $\alpha$.

Example 1. Let $G = [\{A', B', C'\} \Sigma, A', P]$, where $\Sigma = \{S, A, B, a, b\}$ such that $\Sigma_0 = \{a,b\}$. $\Sigma_1 = \{A,B\}$, $\Sigma_2 = \{A,B,S\}$, and $\Sigma_3 = \{A,B\}$.

P consists of the following productions

$$A' \to S(bB') \mid S(aC')$$
$$B' \to A(a) \mid A(aA') \mid A(bA'A')$$
$$C' \to B(b) \mid B(bA') \mid B(aC'C'):$$

Definition 2 - Let G be a cftg. A dendroid form of G is defined recursively as follows:

(i) $\sigma$ is a dendroid form, if $\sigma \in \Sigma$

(ii) If $\alpha(\alpha_1 \ldots \alpha_n)$ is a dendroid form, and for some i, $\alpha_i \to t_i$ is a production in P, then $\alpha(\alpha_1 \ldots \alpha_{i-1} t_i \alpha_{i+1} \ldots {}_n)$ is a dendroid form.

(iii) If $t \in T_\Sigma(N)$ is a dendroid form and it contains an element A such that $A \to \alpha$ is a production in P, then $t'$, which is obtained from t by replacing A by $\alpha$, is a dendroid form.

A dendroid form t is said to directly generate another dendroid form $t'$, denoted by $t \Rightarrow t'$ iff either $t = \sigma$ and $\sigma \to t'$ is in P, or or $t'$ is obtained from $t$ by either (ii) or (iii) above. We say $t$ generates $t'$ iff there exists a finite sequence $t_0 = t, t_1, \ldots, t_k = t'$ such that $t_i \Rightarrow t_{i+1}$, for $0 < i < k$.

We will denote by $\overset{*}{\Rightarrow}$ the transitive closure of the relation .

Definition 3 - Let $G$ be a cftg. The dendrolanuage generated by $G$, denoted by $T(G)$, is the set of trees $t \in T_\Sigma$ such that $\sigma \overset{*}{\Rightarrow} t$.

Example 2 - Let $G$ be the cftg given in example 1. Then $T(G)$ is precisely the set of derivation trees generated by the cfg $G' = [\{S,A,B\},\{a,b\},S,P']$, where $P'$ consists of the following productions:

$$S \to bA \mid aB$$
$$A \to a \mid aS \mid bAA$$
$$B \to b \mid bS \mid aBB$$

Definition 4 - Two cftg $G_1$ and $G_2$ are said to be equivalent iff $T(G_1) = T(G_2)$.

Theorem 1 - Every cftg $G$ is equivalent to a normal form cftg $G'$ such that productions of $G'$ are either of the form $A \to \alpha(\alpha_1 \ldots \alpha_n)$, $\alpha \in \Sigma_n$, $\alpha_i \in N \cup \Sigma_0$, or of the form $A \to \beta$, $\beta \in N \cup \Sigma_0$.

Proof: Let $G = [N,\Sigma,\sigma,P]$ be a given cftg. Construct $G' = [N',\Sigma,\sigma,P']$ as follows. $N' = N \cup \{X_a \mid a \in \Sigma - \Sigma_0\}$. $P'$ consists of all the productions in the desired form plus others obtained by the following procedures.

Let $A \to a(t_1 \ldots t_n)$ be a production in $P$ not of the desired form. Then let $A \to a(\alpha_1 \ldots \alpha_n) \in P'$, where

(i) $\alpha_i = t_i$, if $t_i \varepsilon \Sigma_0 \cup N$;

(ii) $\alpha_i = X_{a_i}$, if $t_i = a_i\,(t_1'\ldots t_m')$, $a_i \varepsilon \Sigma_{m>0}$ and $t_i' \varepsilon T_\Sigma(N)$.

Now apply (i) and (ii) to the productions of the form $X_{a_i} \to a_i(t_1'\ldots t_m')$.

Clearly the procedure terminates and that $T(G) = T(G')$.||

Example 3 - Following the procedure given in the proof of theorem 1, a production $A \to a(b(a(b)c)\beta)$ is replaced by three productions

$A \to a(X_b\beta)$, $X_b \to b(X_a c)$, $X_a \to a(b)$.

Theorem 2 - Corresponding to each cftg $G = [N,\Sigma,\sigma,P]$, there exists a cfg $G' = [N',\Sigma',\sigma',P']$ and a projection $\Pi:\Sigma \to N' \cup \Sigma'$ such that $\Pi(T(G)) = D^{G'}_{\sigma'}$. Conversely, if $G'$ is a given cfg, then $D^{G'} = T(G)$ for some cftg $G$.

Proof: Let $G$ be a cftg in normal form. Define a cfg $G'$ as follows.

$N' = \{\sigma\} \cup \{(A,a) \mid A \to a(\alpha_1\ldots\alpha_n) \varepsilon P\}$

$\sigma' = \sigma$

$P'$ consists of the following productions:

i) $\alpha \to X_1X_2\ldots X_n$, if $\alpha \to a(\alpha_1\alpha_2\ldots\alpha_n)$ P, where

$X_i = \alpha_i$ if $\alpha_i \varepsilon \Sigma_0$, and $X_i = (A,a')$ if $\alpha_i = A$ and $A \to a'(\beta_1\ldots\beta_m)$ P.

ii) $(A,a) \to X_1X_2\ldots X_n$, if $A \to a(\alpha_1\alpha_2\ldots\alpha_n) \varepsilon P$, where $X_i = \alpha_i$ if $\alpha_i \varepsilon \Sigma_0$, and $X_i = (\alpha_i,a')$ if $\alpha_i \varepsilon N$ and $\alpha_i \to a'(\beta_1\ldots\beta_n) \varepsilon P$.

Now, let $\Pi: \Sigma \to N' \cup \Sigma'$ such that

$$\Pi(\alpha) = \begin{cases} (A,\ ), & \text{if } A \to (\alpha_1\alpha_2\ldots\alpha_n) \varepsilon P \\ \alpha, & \text{if } \alpha \to \alpha(\alpha_1\ldots\alpha_n) \varepsilon P \\ \alpha, & \text{if } \alpha \varepsilon \Sigma_0. \end{cases}$$

Clearly, $\Pi(T(G) = D^{G'}$.

Conversely, let $G' = [N', \Sigma', \sigma', P']$ be a cfg. Construct a cftg $G = [N, \Sigma, \sigma, P]$ as follows:

$N = \{A \mid A' \varepsilon N\}$, $\Sigma = N' \cup \Sigma'$, and $P$ consists of the productions $A \rightarrow A'(\beta_1 \ldots \beta_n)$, if $A' \rightarrow \alpha_1 \ldots \alpha_n \varepsilon P'$, where $\beta_i = \alpha_i$, if $\alpha_i \varepsilon \Sigma'$, and $\beta_i = A$, if $\alpha_i = A'$.

Clearly, $D^{G'} = T(G)$.||

Corallary 1 - Let $G$ be a cftg. Then the language $\{ \|t\| \mid t \varepsilon T(G)\}$ is context-free.

We remark here that the frontiers of the trees generated by a regular tree grammar need not be a regular language. For example, consider the regular grammar $G = [\{\sigma\}, \{a,b,d\}, \sigma, P]$, where $\Sigma_0 = \{a,b\}$, $\Sigma_2 = \Sigma_3 = \{d\}$, and P consists of the productions

$$\sigma \rightarrow d(a \sigma b) \mid d(ab).$$

The language $\{\|t\| \mid t \varepsilon T(G)\} = \{a^n b^n \mid n > 1\}$ is context-free.

## III. Pushdown Tree Automata

In this section, we will introduce a recognizer of trees which "walks" on input trees. Correspondence between this class of recognizers and cftg will be proved.

Definition 5 - A pushdown tree automaton (p.t.a.) is a 7-tuple $\underline{P} = [Q, \Sigma, \Gamma, \delta, q_0, z_0, F]$, where $Q, \Sigma, \Gamma$ and $F \subseteq Q$ are finite sets of states, input alphabet, pushdown symbols, and final state set. $q_0 \varepsilon Q$ and $z_0 \varepsilon Z$ are the initial state and starting pushdown symbol, respectively. $\delta$ is a partial function mapping $Q \times \{\Sigma \cup \{\Lambda\}\} \times \Gamma$ into finite subsets of $Q \times \Gamma^* \times \{-1,0,1\} \times \{0,1\}$.

We may view a p.t.a. $\underline{P}$ as a pushdown automaton whose inputs are labelled, rooted trees. We assume that the $\underline{P}$ has a read head with an arrow attached to it. Initially, it is assumed that the read head is reading the label of the root $v_0$ of its input tree with its reading arrow coinciding with the leftmost edge $(v_0,u)$ incident from the root. The automaton makes an atomic move as follows. If $(q',\gamma,d_1,d_2) \varepsilon \delta(q,a,z)$, it means that the automaton was in state $q$ with $z$ as the topmost symbol of the Pushdown stack, and the reading arrow is coinciding with an edge $(u,v)$ such that the label of $u$ is $a$. $\underline{P}$ then changes to state $q'$, replaces $z$ by $\gamma$, moves its reading arrow according to $d_1 \varepsilon\{-1,0,1\}$ deleting edges according to $d_2 \varepsilon\{0,1\}$.

If $d_1 = -1$, then the reading arrow reverses its direction. This means that if the read head were reading $a$, the label of $u$, and the arrow is pointing in the direction $(u,v)$ for some $v$, then the read head is reading the label of $v$ and reading arrow is now pointing in the direction of $(v,u)$. If $d_1 = 0$, then the reading arrow remains stationary. If $d_1 = 1$, then the read head is stationary but the reading arrow will make a counterclockwise swing until it meets another edge. Thus, if the read head is on a vertex $u$ which has successors $v_1,v_2 \ldots v_k$ and predecessor $u'$ such that the reading arrow was on the edge $(u,v_i)$ then after the transition, the reading head is on the edge $(u,v_{i+1})$, if $i < k$, and $(u,u')$, if $i = k$.

If $d_2 = 1$, this means that the reading head will delete the edge it is on. If $d_2 = 0$, no deletion will occur.

A Configuration of a p.t.a. P is a 4-tuple $(T,q,a,Zx)$, where $T$ is the undeleted portion of the input tree, $q$ is the current state of P, a is the current input symbol being read, and $Z\alpha$ is the content of the pushdown stack.

Let $\vdash_{P}$ (or simply $\vdash$ when no confusion arises) be a binary relation defined on the set of all configurations of P such that $(T,q,a,z\alpha) \vdash (T',q',b,\gamma\alpha)$ if $(q',\gamma,d_1,d_2) \in \delta(q,a,z)$ where $b$ is the symbol read by means of a $d_1$ operation, and $T'$ is obtained from $T$ by $d_2$ operations. Let $\overset{*}{\vdash}$ be the transitive closure of $\vdash$.

Definition 6 - Let P be a p.t.a A tree $T \in T_{\Sigma}$ is said to be recognizable by P iff

$$(T,q_0,a,z_0) \overset{*}{\vdash} (\Lambda_a,q,a,\alpha)$$

for some $q$ F. Where $a$ is the label of root of $T$ and $\Lambda a$ is a degenerate tree which contains a single node with label a.

The tree dendrolanguage recognizable by P, denoted by L(P), is the set of all trees recognizable by P.

Remark: By the usual technique, it is easy to show that the set of trees recognizable by a p.t.a. P by emptying stack is the same as L(P). In other words,

$$L(P) = \{T \mid (T,q_0,a,z_0) \overset{*}{\vdash} (\Lambda_a,q,a,\Lambda) \text{ for some } q \in Q \text{ and } a \text{ is the label of the root of } T\}.$$

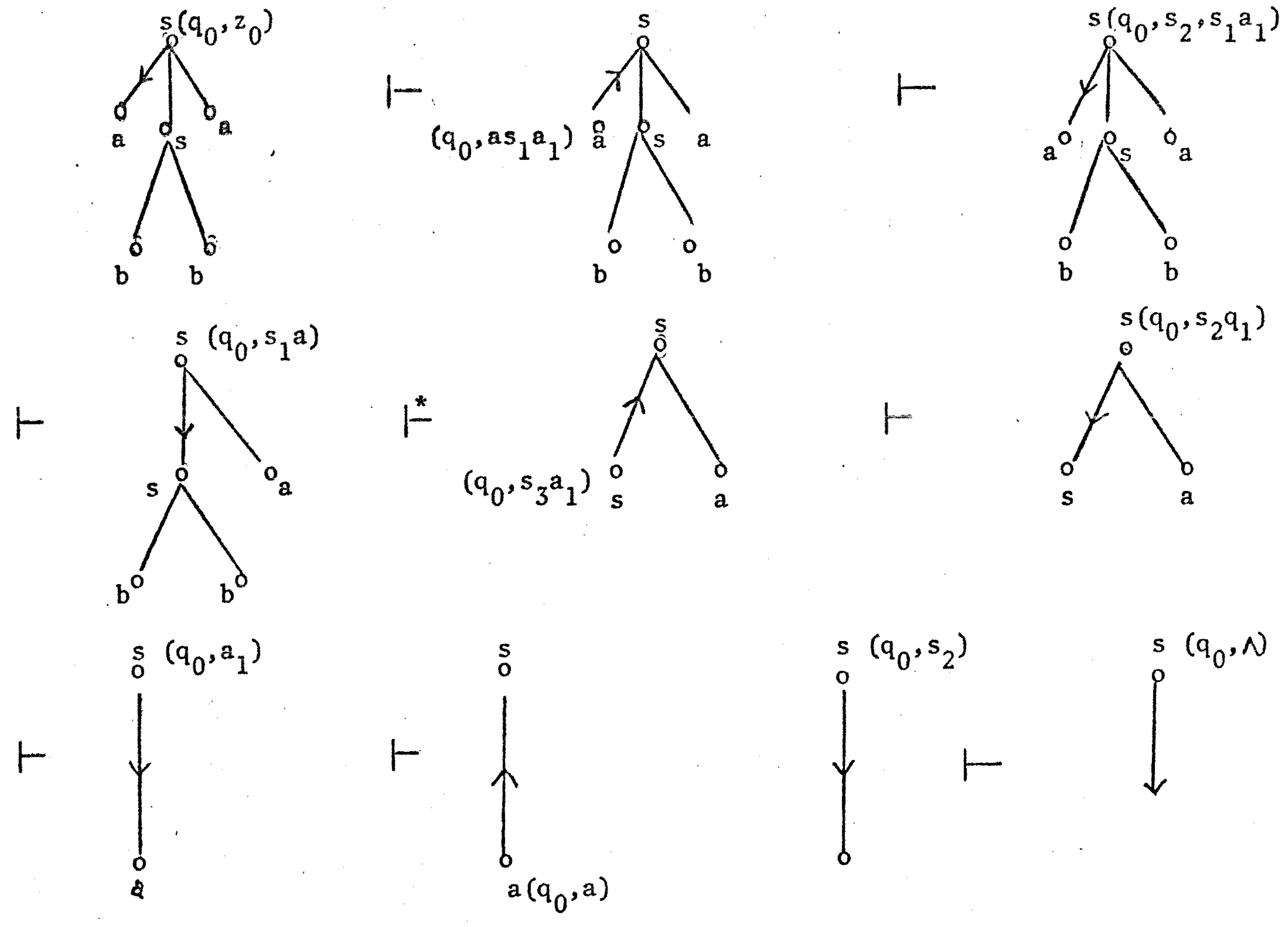
Example 4 - Let $P = [\{q_0\},\{a,b,s\},\{z_0,a,a_1,b,b_1,s,s_1,s_2,s_3\}, \delta,q_0,z_0,\phi]$ be a given p.t.a. with $\delta$ given in table 1. It is easily seen that $L(P) = D^G$, where $G = [\{s\},\{a,b\},s,P]$, where P consists of the following productions.

$$s \rightarrow asa \mid bsa \mid aa \mid bb.$$

| Q | Σ | Γ | Q | Γ* | $d_1$ | $d_2$ |
|---|---|---|---|---|---|---|
| $q_0$ | s | $Z_0$ | $q_0$ | $as_1a_1$ | -1 | 0 |
| $q_0$ | s | $Z_0$ | $q_0$ | $bs_1b_1$ | -1 | 0 |
| $q_0$ | s | $Z_0$ | $q_0$ | $aa_1$ | -1 | 0 |
| $q_0$ | s | $Z_0$ | $q_0$ | $bb_1$ | -1 | 0 |
| $q_0$ | s | s | $q_0$ | $a_1s_1a_1s_3$ | 1 | 0 |
| $q_0$ | s | s | $q_0$ | $b_1s_1b_1s_3$ | 1 | 0 |
| $q_0$ | s | s | $q_0$ | $a_1a_1s_3$ | 1 | 0 |
| $q_0$ | s | s | $q_0$ | $b_1b_1s_3$ | 1 | 0 |
| $q_0$ | s | $a_1$ | $q_0$ | a | -1 | 0 |
| $q_0$ | s | $b_1$ | $q_0$ | b | -1 | 0 |
| $q_0$ | s | $s_1$ | $q_0$ | s | -1 | 0 |
| $q_0$ | s | a | $q_0$ | $s_2$ | -1 | 0 |
| $q_0$ | s | b | $q_0$ | $s_2$ | -1 | 0 |
| $q_0$ | s | $s_3$ | $q_0$ | $s_2$ | -1 | 0 |
| $q_0$ | s | $s_2$ | $q_0$ | Λ | 1 | 1 |
| $q_0$ | s | $s_2$ | $q_0$ | Λ | 0 | 1 |

Table 1 - Transition table for P

We will illustrate a computation sequence of P by the following diagrams:



<u>Lemma 1</u> - Let $G$ be a cfg. Then there exists a p.t.a. $\underline{P}$ such that $L(\underline{P}) = D^G$.

Proof: We construct $\underline{P}$ to simulate the leftmost derivations in $G$.

Let $G = [N,\Sigma,\sigma,P]$. Construct $\underline{P} = [\{q_0\},\Sigma',\Gamma,\delta,q_0,z_0,\{\phi\}]$ such that $\Sigma' = N \cup \Sigma$, $\Gamma = N \cup \Sigma \cup \{B^1,B^2,B^3 \mid B \in N\} \cup \{a^1 \mid a \in \Sigma\}$. $\delta$ is defined such that corresponding to each production $B \to \alpha_1 \ldots \alpha_n$, $\alpha_i \in N \cup \Sigma$, in $P$, we have

$$\left.\begin{array}{ll} 1) & (q_0,B,z_0) \vdash (q_0,\alpha_1\alpha_2^1\alpha_3^1\ldots\alpha_n^1,-1,0) \\ 2) & (q_0,B,B^2) \vdash (q_0,\Lambda,0,-1) \end{array}\right\} \text{ if } B = \sigma$$

3) $(q_0,B,B) \vdash (q_0,\alpha_1^1\alpha_2^1\ldots\alpha_n^1 B^3,1,0)$

4) $(q_0,\alpha_i,\alpha_i^1) \vdash (q_0,\alpha_i,-1,0)$

5) $(q_0,\alpha_i,\alpha_i) \vdash (q_0,B^2,-1,0)$, for $\alpha_i \in \Sigma$

6) $(q_0,\alpha_i,\alpha_i^3) \vdash (q_0,B^2,-1,0)$, for $\alpha_i \in N$.

It can be shown in a straightforward fashion by induction that

$$t \in D^G \Longleftrightarrow (t,q_0,a,z_0) \overset{*}{\vdash} (\Lambda_a,q,a,\Lambda). \|$$

As a consequence of theorem 2 and lemma 1, we have the following result.

<u>Corollary 2</u> - If $G$ is a cftg, then $T(G) = L(\underline{P})$ for some one state p.t.a. $\underline{P}$.

Since the deletion function of a p.t.a. corresponds to marking the edges of an input tree, and that all p.t.a. under discussion are non-deterministic, we have the following straightforward result.

Lemma 2 - Every one state p.t.a. P is equivalent to a one state p.t.a. which recognizes trees by always performing a preorder traversal and deleting terminal edges as its reading head encounters them.

Theorem 3 - Corresponding to each one state p.t.a $\underline{P}$, there exists a cftg G such that $L(\underline{P}) = T(G)$.

Proof: By lemma 2, we may assume that $\underline{P}$ is a preorder p.t.a since $\underline{P}$ must perform deletion when after reading an input which belongs to $\Sigma_0$, the only transition which contributes to the generation of trees is of the form

$(q,a,\gamma,1,0) \in \delta(q,a,z)$ or $(q,a,\gamma,-1,0) \in \delta(q,a,z)$

Thus, corresponding to each such transitions, we add the production $z \rightarrow a(\gamma)$ to G.||

Lemma 3 - A dendrolanguage is p.t.a. recognizable iff it is one state p.t.a. recognizable.

Proof: Let $\underline{P} = [Q,\Sigma,\Gamma,\delta,q_0,z_0,\phi]$ be a p.t.a. Construct a one-state p.t.a. $\underline{P}' = [\{q_0\},\Sigma,\Gamma',\delta',q_0z_0',\phi]$ as follows:

$\Gamma' = \{[q,z] \mid q \in Q, z \in \Gamma\}$

$z_0' = [q_0,z_0]$

$\delta'$ is defined as follows:

i) wherever $(P,\Lambda,d_1,d_2) \in \delta(q,a,z)$, then $(q_0,\Lambda,d_1,d_2) \in \delta'(q_0,a,[q,z])$

ii) $\delta'(q_0,a,[q,z])$ contains $[(q_0,[p,z_1] [q_2,z_2]\ldots(q_m,z_m), d_1,d_2]$, $P,q_2,\ldots,q_m \in Q$ wherever $(P,z_1\ldots z_m,d_1,d_2) \in \delta(q,a,z)$. ||

As a consequence of previous results, we have

Theorem 4 - A dentrolanguage L is p.t.a. recognizable iff it is generated by a cftg.

As a consequence of [1,2,4], we have the following result.

Corollary 3: The following statements are equivalent.

1. L is p.t.a recognizable;
2. L is recognizable by a generalized finite automata [4];
3. L is recognizable by a finite recursive tree recognizer [2];
4. L is generated by a cftg;
5. L is generated by a regular tree system [1 ].

To complete this section, we given the following definition and the straightforward result.

Definition 7 - A finite tree automata (f.t.a) is a 5-tuple $A = [Q,\Sigma,\delta,q_0,F]$, where Q, and F are finite sets of states, input alphabet, and final state set. $q_0 \in Q$ is called the initial state, and $\delta$: $Q \times \Sigma \to Q \times \{-1,0,1\} \times \{0,1\}$ is the transition function of A.

The concepts of "configuration" and "recognition" for A can be defined in a way similar to that for a p.t.a. and hence will not be given here.

By observing that the trees generated by a regular tree grammar is really "linear" in the sense that all edges of each tree are incident with a vertex in the longest path of the tree. This observation leads to the following straightforward result. (A detailed proof of this result is quite tedious, however.)

Theorem 5 - A dendrolanguage L is recognizable by a f.t.a. iff it is generated by a regular tree grammar.

## IV. Concluding Remarks

It was established in the previous sections that a correspondence between the walking automata and the generating grammars do exist. It is our conjecture that the usual Chomsky hierarchy generalizes to the tree case using our model. Furthermore, our model generalizes to the case of hierarchical graphs [3] which is really the motivation for this paper.

## V. Acknowledgment

This paper is supported in part by the National Science Foundation grant GJ-36424, and by the Joint Services Electronics Program under research grant F44 620-71-C-0091.

## VI. References

1. Brainard, W. S., "Tree Generating Regular Systems," Information and Control, 14 (1969), 217-231.

2. Ng, P. and Yeh, R. T., "Tree Transformations Via Finite Recursive Transition Machines," Proc. Int. Symposium on Math. Foundations of Computer Science, High Tatras, Czech. (1973), 273-277.

3. Pratt, T. W., "Pair Grammars, Graph Languages and String-to-Graph Translations," J. Computer and System Sciences (1971), 560-599.

4. Thatcher, J. W. and Wright, J. B., "Generalized Finite Automata Theory with an Application to a Decision Problem of Second-order Logic," Math. Systems Theory, 2 (1968) 57-81.