ADULT LANGUAGES OF L SYSTEMS AND THE CHOMSKY HIERARCHY

Adrian Walker

Department of Computer Science

State University of New York at Buffalo

Introduction

The concept of an L system was first introduced by Lindenmayer [59, 60] as "a theoretical framework within which intercellular relationships can be discussed, computed, and compared". The concept has proved to be a fruitful one, and has opened up a new area of interdisciplinary research. Much of the work to date on L systems is reported in Herman and Rozenberg [45]. The motivation for the present paper is the thought that, since L systems are proving so useful as a framework for studying biological growth and development, perhaps they can also be used to study the ways in which organisms achieve and maintain relatively stable adult states. Thus while the emphasis in work on L systems to date has been on all the strings derivable from an initial string, we shall focus in this paper on just those strings which renew themselves dynamically once they have been derived.

Notation

We write λ for the empty string, $|\alpha|$ for the length of a string α (e.g. $|\lambda| = 0$), and $\#V$ for the number of elements in a set V. If α is a string we write the set of symbols occurring in α as sym α, e.g. sym abbac = {a, b, c}. If L is a set of strings we write sym L for $\bigcup_{\alpha\varepsilon L}$ sym α. We write the number of occurrences of the symbol a in a string α as $\#_a(\alpha)$, e.g. $\#_a(\text{abbac}) = 2$.

We abbreviate context free grammar, context sensitive grammar, linear bounded automaton, and Turing machine as CFG, CSG, LBA and TM respectively. We require that if $\alpha \to \beta$ is a production of a CSG then $|\alpha| \leq |\beta|$. We write the classes of context free, context sensitive languages not containing λ, and recursively enumerable languages as L(CF), L(CS) and L(RE) respectively. Otherwise we use the notation of Hopcroft and Ullman† for phrase structure grammars.

If δ is a mapping from a set of strings into a set of sets of strings, we say that $\delta^0(\alpha) = \{\alpha\}$, and for each $i \geq 0$ $\delta^{i+1}(\alpha) = \delta\delta^i(\alpha)$. We say that $\delta^*(\alpha) = \bigcup_{i=0}^{\infty} \delta^i(\alpha)$.

Definitions

A <u>0L system</u> is a 3-tuple $H = \langle V, \delta, S\rangle$ where V is an alphabet, $S \in V$ and δ is a mapping from V^* into the finite subsets of V^* defined as follows. There is a table Q of productions $Q \subset V \times V^*$ such that for each $b \in V$ there is a $\beta \in V^*$ such that $\langle b, \beta\rangle \in Q$. $\delta(\lambda) = \{\lambda\}$ and for $\alpha = a_1 \ldots a_n$, $\delta(\alpha) = Q(a_1) \ldots Q(a_n)$. If for each production $\langle b, \beta\rangle \in Q$ $\beta \neq \lambda$ we say that H is a propagating 0L system, or <u>P0L system</u> for short.

A <u>2L system</u> is a 4-tuple $H = \langle V, \delta, g, S\rangle$ where V and S are as in a 0L system, g is a symbol not in V, and δ is a mapping from V^* into the finite subsets of V^* defined as follows. There is a table Q of productions $Q \subset V_g V V_g \times V^*$, where $V_g = V \cup \{g\}$ such that for each $abc \in V_g V V_g$ there is a $\beta \in V^*$ such that $\langle abc, \beta\rangle \in Q$. $\delta(\lambda) = \{\lambda\}$, and for $\alpha = a_1 \ldots a_n$, $\delta(\alpha) = Q(a_0 a_1 a_2) \ldots Q(a_{j-1} a_j a_{j+1}) \ldots Q(a_{n-1} a_n a_{n+1})$ where a_0 and a_{n+1} stand for g.

---

† Hopcroft, J. E., J. D. Ullman, <u>Formal Languages and their Relation to Automata</u>, Addison-Wesley, Reading, Mass., 1969. From now on we refer to this book simply as H & U.

If for each production $<abc, \beta> \varepsilon Q$ $\beta \neq \lambda$ we say that H is a <u>P2L system</u>.

If H is an L system with mapping δ and initial symbol S, we define the <u>adult language</u> of H as $A(H) = \{\alpha \varepsilon \delta^*(S) \mid \delta(\alpha) = \{\alpha\}\}$.

<u>Phrase Structure Grammars</u>

We now summarize some results about phrase structure grammars which we shall need later.

We follow Aho and Ullman[†] in saying that a CFG $G = <V_N, V_T, P, S>$ is <u>proper</u> if

(i) for each $A \varepsilon V_N$ it is not the case that $A \overset{+}{\Rightarrow} A$,

(ii) either P has no productions of the form $A \to \lambda$, or $S \to \lambda$ is the only such production and S never appears on the right of a production, and

(iii) for each $B \varepsilon V_N$ there exist $\alpha, \beta, \gamma \varepsilon V_T^*$ such that $S \overset{*}{\Rightarrow} \alpha B \gamma \overset{*}{\Rightarrow} \alpha\beta\gamma$.

The following result is obtainable by algorithms 2.8-2.11 of Aho and Ullman[†].

<u>Lemma 1</u> There exists an algorithm which takes as input any CFG G and produces as output a proper CFG G' such that $L(G) = L(G')$.

<u>Lemma 2</u> There exists an algorithm which takes as input any grammar G and produces as output a grammar G' such that

(i) if $\alpha' \to \beta'$ is a production of G' then $|\alpha'| \varepsilon \{1, 2\}$,

(ii) if $\alpha' \to \lambda$ is a production of G', then $|\alpha'| = 1$,

---

† Aho, A. V., J. D. Ullman, <u>The Theory of Parsing, Translating and Compiling</u>, volume 1, Prentice Hall, Englewood Cliffs, 1972.

(iii) if G is a CSG then so is G', and

(iv) $L(G) = L(G')$.

Proof Let $G = \langle V_N, V_T, P, S\rangle$ be a grammar. It is easy to see that we lose no generality by assuming that if $\alpha \underset{P}{\rightarrow} \lambda$ then $|\alpha| = 1$. Then to construct $G' = \langle V_N', V_T, Q, S\rangle$, place each production in P having a left side of length 1 or 2 directly in Q. For each production $A_1 \ldots A_m \underset{P}{\rightarrow} B_1 \ldots B_n$ where $A_i, B_j \in (V_N \cup V_T)$ and $m \geq 3$, Q contains $A_1A_2 \rightarrow B_1C_2$, $Z \rightarrow \lambda$, and in addition

(i) $C_iA_{i+1} \rightarrow B_iC_{i+1}$ $(2 \leq i \leq m-2)$ and
$C_{m-1}A_m \rightarrow B_{m-1} \ldots B_n$, if $m \leq n$;

(ii) $C_iA_{i+1} \rightarrow B_iC_{i+1}$ $(2 \leq i \leq n-1)$ and
$C_nA_m \rightarrow B_nZ$, if $m = n+1$;

(iii) $C_iA_{i+1} \rightarrow B_iC_{i+1}$ $(2 \leq i \leq n)$,
$C_iA_{i+1} \rightarrow ZC_{i+1}$ $(n < i \leq m-2)$, and
$C_{m-1}A_m \rightarrow Z$, if $m \geq n+2$.

In this construction the C_i's are new symbols, and if productions $p_1, p_2 \in P$ give rise to subsets Q_1, Q_2 of Q, then the C_i's in Q_1 and Q_2 are distinct.

It is straightforward to check that our construction has the required properties. ■

Adult Languages of 0L Systems

In order to characterize the adult languages of 0L systems, we first derive a property of the productions which must hold in order for a string to map only into itself. Note that it is not necessarily the case that $a \rightarrow a$ for each letter in such a string, e.g. if $a \rightarrow ab$, $b \rightarrow c$ and $c \rightarrow \lambda$, then $\delta(abc) = \{abc\}$. (When $\delta(\alpha) = \{\beta\}$ we shall write simply $\delta(\alpha) = \beta$.)

Lemma 3 If $H = \langle V, \delta, S\rangle$ is a 0L system, $\Sigma = \text{sym } A(H)$, and $m = \#\Sigma$, then for each $a \in \Sigma$ there is a unique $\beta \in \Sigma^*$ such that $\delta^*\delta^m(a) = \beta$.

Proof

1. For each $a \in \Sigma$, $\#\delta(a) = 1$ and $\delta(a) \in \Sigma^*$: if $a \in \Sigma$ then there exist $\alpha_1, \alpha_2 \in \Sigma^*$ such that $\delta(\alpha_1 a \alpha_2) = \alpha_1 a \alpha_2 = \delta(\alpha_1)\delta(a)\delta(\alpha_2)$.

2. If $a \in \Sigma$ then $\#_a\delta(a) \in \{0, 1\}$: if $a \in \Sigma$ then there exist $\alpha_1, \alpha_2 \in \Sigma^*$ such that $\delta(\alpha_1 a \alpha_2) = \alpha_1 a \alpha_2 = \delta(\alpha_1)\delta(a)\delta(\alpha_2)$. Hence if $\#_a\delta(a) = k$ then $\#_a\delta^i(\alpha_1 a \alpha_2) \geq k^i$ for each $i \geq 0$. Since $\delta^*(\alpha_1 a \alpha_2) = \alpha_1 a \alpha_2$ and $\#_a(\alpha_1 a \alpha_2) \leq |\alpha_1 a \alpha_2|$ it is obvious that we must have $k \leq 1$.

3. If $a \in \Sigma$ and $\#_a\delta(a) = 0$ then $\delta^m(a) = \lambda$: suppose that for all $i \geq 0$, $\delta^i(a) \neq \lambda$. Since $a \in \Sigma$ we have $a \in \text{sym } \gamma$ for some γ such that $\delta(\gamma) = \gamma$. Since $\#_a\delta(a) = 0$, we can write either (i) $\gamma = u\delta(a)\, v\, a\, w$ for some $u, v, w \in \Sigma^*$, or (ii) $\gamma = uav\delta(a)w$ for some $u, v, w \in \Sigma^*$. Hence, since $\delta(\gamma) = \gamma$ we can show that $|\delta^i(\gamma)| \geq |\delta^0(a)\ldots\delta^i(a)|$ for each $i \geq 0$. But then $|\delta^{|\gamma|}(\gamma)| \geq |\delta^0(a)\ldots\delta^{|\gamma|}(a)| > |\gamma|$, a contradiction since $\delta^{|\gamma|}(\gamma) = \gamma$. So it must be the case that for some $i \geq 0$, $\delta^i(a) = \lambda$. From this it is easy to show by path length arguments that $\delta^m(a) = \lambda$.

4. For each $a \in \Sigma$ such that $\#_a\delta(a) = 1$, there is a unique $\beta \in \Sigma^+$ such that $\delta^*\delta^m(a) = \beta$: since $\#_a\delta(a) = 1$ we can write $\delta(a) = \alpha a \bar{\alpha}$ for some $\alpha, \bar{\alpha} \in (\Sigma - \{a\})^*$. If $\delta^i(\alpha\bar{\alpha}) \neq \lambda$ for all $i \geq 0$ then it is easy to see that for any ℓ there exists a j such that $|\delta^j(a)| > \ell$, which is impossible since a occurs in a string γ such that $\delta(\gamma) = \gamma$. So there is an i such that $\delta^i(\alpha\bar{\alpha}) = \lambda$, and hence by 3. we have that $\delta^m(\alpha\bar{\alpha}) = \lambda$. Let r, s be the greatest integers less than or equal m such that $\delta^r(\alpha) \neq \lambda$, $\delta^{r+1}(\alpha) = \lambda$,

$\delta^s(\bar{\alpha}) \neq \lambda$, and $\delta^{s+1}(\bar{\alpha}) = \lambda$. Then it is easy to see that if we write $\beta = \delta^r(\alpha) \ldots \delta^0(\alpha)a\delta^0(\bar{\alpha}) \ldots \delta^s(\bar{\alpha})$ then $\delta^*\delta^m(a) = \beta$. The lemma now follows from 2, 3, and 4. ■

We shall need to know how to find sym A(H) for any 0L system H.

<u>Lemma 4</u> There exists an algorithm which takes as input any 0L system H and produces as output the set sym A(H).

<u>Proof</u> Let $H = \langle V, \delta, S\rangle$ be a 0L system, and let $\Sigma = \text{sym } A(H)$, $m = \#\Sigma$, and $n = \#V$. Let $L = \{\alpha \in \delta^i(S) \mid i \leq 2^n + m, \delta(\alpha) = \alpha\}$. We claim that $\Sigma = \text{sym } L$.

Obviously $\text{sym } L \subset \Sigma$. Suppose $b \in \Sigma$. Then there is an $\alpha \in A(H)$ such that $b \in \text{sym } \alpha$. So $\alpha \in \delta^*(S) \cap \{u \mid \delta(u) = u\}$. Hence there exists an $i \geq 0$ such that $\alpha \in \delta^i(S)$ and $\alpha \in \delta^{i+m}(S)$. But it is easy to check that if $\alpha \in \delta^i(S)$, there exists an $\bar{\alpha} \in \delta^{2^n}(S)$ such that $\text{sym } \alpha = \text{sym } \bar{\alpha}$. Hence by Lemma 3 there exists an $\bar{\bar{\alpha}} \in \delta^m(\bar{\alpha})$ such that $\text{sym } \bar{\bar{\alpha}} = \text{sym } \alpha$ and $\delta(\bar{\bar{\alpha}}) = \bar{\bar{\alpha}}$. So $\bar{\bar{\alpha}} \in L$ and hence $b \in \text{sym } L$. ■

We can use the last two lemmas to put any 0L system in a form in which $\delta(a) = a$ for each letter a which occurs in the adult language.

<u>Lemma 5</u> There exists an algorithm which takes as input any 0L system G and produces as output a 0L system H such that $A(G) = A(H)$ and for each $a \in \text{sym } A(H)$, $\delta_H(a) = a$.

<u>Proof</u> Let $G = \langle V, \delta_G, S\rangle$ be a 0L system. Let $\Sigma_G = \text{sym } A(G)$, and let $m = \#\Sigma_G$.

If $\Sigma_G = \emptyset$ then we are done, so suppose $\Sigma_G \neq \emptyset$. Let $H = \langle V, \delta_H, S\rangle$ be a 0L system constructed from G as follows. Define a mapping $\theta: V \to V^*$ by

$$\theta(a) = \begin{cases} a, & \text{if } a \in V - \Sigma_G \\ \delta_G^m(a), & \text{if } a \in \Sigma_G, \end{cases}$$

extend θ to domain V^* by $\theta(\lambda) = \lambda$ and $\theta(a\alpha) = \theta(a)\theta(\alpha)$, and further to domain 2^{V^*} in the obvious manner. Then define $\delta_H: V \to 2^{V^*}$ by

$$\delta_H(a) = \begin{cases} \theta\delta_G(a), & \text{if } a \in V - \Sigma_G \\ a, & \text{if } a \in \Sigma_G. \end{cases}$$

By lemmas 3 and 4, H is well-defined. We claim that $A(G) = A(H)$.

1. For every $t \geq 0$ and $\beta \in V^*$, $\beta \in \delta_H^t(S)$ iff there exists an $\alpha \in V^*$ such that $\alpha \in \delta_G^t(S)$ and $\theta(\alpha) = \beta$: this is straightforward to prove by induction on t.

2. $A(G) \subset A(H)$: Let $\alpha = a_1 \ldots a_n \in A(G)$. Then $a_j \in \Sigma_G$. Let $\delta_G^m(a_j) = \delta_G^{m+1}(a_j) = \beta_j$. Since $\delta_G^m(\alpha) = \alpha$, we have $\beta_1 \ldots \beta_n = \alpha$, and so $\theta(\alpha) = \beta_1 \ldots \beta_n = \alpha$. Since $\alpha \in \delta_G^*(S)$ we have from 1. that $\theta(\alpha) \in \delta_H^*(S)$. But $\theta(\alpha) = \alpha$, so $\alpha \in \delta_H^*(S)$. Also, since $\alpha \in \Sigma_G^*$ it follows from the construction of δ_H that $\delta_H(\alpha) = \alpha$. Hence $\alpha \in A(H)$.

3. $A(H) \subset A(G)$: Let $\beta \in A(H)$. Then $\beta \in \delta_H^*(S)$. So it follows from 1. that there exists an α such that $\alpha \in \delta_G^*(S)$ and $\theta(\alpha) = \beta$. Let $\alpha = \alpha_0 A_1 \alpha_1 \ldots \alpha_{n-1} A_n \alpha_n$, where $\alpha_j \in \Sigma_G^*$, $A_j \in (V - \Sigma_G)$, and $n \geq 0$. Since $\alpha_j \in \Sigma_G^*$ it follows from Lemma 3 that there is a $\beta_j \in \Sigma_G^*$ such that $\delta_G^m(\alpha_j) = \delta_G^{m+1}(\alpha_j) = \beta_j$. It follows from this and the definition of θ that $\theta(\alpha) = \beta_0 A_1 \beta_1 \ldots \beta_{n-1} A_n \beta_n$. Since $\beta_j \in \Sigma_G^*$, we have from the construction of δ_H that $\delta_H(\beta_j) = \beta_j$. Since $\beta \in A(H)$, we have $\delta_H(\beta) = \beta$. Since $\delta_H(\beta) = \beta$ and $\delta_H(\beta_j) = \beta_j$ it is clear that $\delta_H(A_j) = A_j$. Since $A_j \notin \Sigma_G$, if $\gamma_j \in \delta_G(A_j)$

then $\theta(\gamma_j) \in \delta_H(A_j)$, and so $\theta(\gamma_j) = A_j$. But this is only possible if $\gamma_j = A_j$. Hence $\delta_G(A_j) = A_j$. Now since $\delta_G(\beta_j) = \beta_j$ and $\beta = \theta(\alpha) = \beta_o A_1 \beta_1 \ldots \beta_{n-1} A_n \beta_n$, we have $\delta_G(\beta) = \beta$. Moreover, since $\delta_G^m(\alpha_j) = \beta_j$ and $\delta_G^m(A_j) = A_j$, we have $\delta_G^m(\alpha) = \beta$. Hence $\beta = \delta_G(\beta) \in \delta_G^m(\alpha) \in \delta_G^*(S)$, and so $\beta \in A(G)$.

2. and 3. together establish our claim that $A(H) = A(G)$. ■

We shall use Lemmas 4 and 5 to characterize the class $A(0L)$ of adult languages of 0L systems. First we need the following notation. If $G = \langle V_N, V_T, P, S\rangle$ is a CFG with $V_N \cup V_T = V$ we define a mapping $\psi_G: V \to 2^{V^*}$ by

$$\psi_G(a) = \begin{cases} a, & \text{if } a \in V_T \\ \{\beta \mid a \underset{P}{\to} \beta\} & \text{if } a \in V_N, \end{cases}$$

and we extend ψ_G to domain V^* by $\psi_G(\lambda) = \lambda$ and $\psi_G(a\alpha) = \psi_G(a)\psi_G(\alpha)$. It is easy to check that $L(G) = \psi_G^*(S) \cap V_T^*$.

<u>Lemma 6</u> There exists an algorithm which takes as input any 0L system H and produces as output a CFG G such that $A(H) = L(G)$.

<u>Proof</u> Let $H = \langle V, \delta_H, S\rangle$ be a 0L system, let $\Sigma = \text{sym } A(H)$, and assume without loss of generality that $S \in V - \Sigma$. By Lemma 5 we may also assume that for each $a \in \alpha$, $\delta_H(a) = a$. Let $G = \langle V - \Sigma, \Sigma, P, S\rangle$ be a CFG constructed from H, where $P = \{A \to \alpha \mid A \in V - \Sigma$ and $\alpha \in \delta_H(A)\}$. By Lemma 4 we can compute Σ from H, so our construction is effective.

Now it is easy to check from our construction that for each $i \geq 0$, $\delta_H^i(S) = \psi_G^i(S)$. Hence $\delta_H^*(S) = \psi_G^*(S)$. So $\delta_H^*(S) \cap \Sigma^* = \psi_G^*(S) \cap \Sigma^*$. But since $\delta_H(a) = a$ for each $a \in \Sigma$, it is easy to see that $A(H) = \delta_H^*(S) \cap \Sigma^*$, and it is a property of our notation ψ_G that $L(G) = \psi_G^*(S) \cap \Sigma^*$, hence $A(H) = L(G)$. ■

We can prove the converse of Lemma 6.

Lemma 7 There exists an algorithm which takes as input any CFG G and produces as output a 0L system H such that $L(G) = A(H)$.

Proof Let $G = \langle V_N, V_T, P, S\rangle$ be a CFG. By Lemma 1 we may assume that G is proper. Let $H = \langle V, \delta_H, S\rangle$ be constructed from G by $V = V_N \cup V_T$, and

$$\delta_H(a) = \begin{cases} \{\alpha \mid a \underset{P}{\rightarrow} \alpha\}, & \text{if } a \in V_N \\ a, & \text{if } a \in V_T \end{cases}$$

Clearly the construction is effective, and since G is proper $\delta: V \rightarrow 2^{V^*}$ is everywhere defined, so H is a 0L system.

It follows from our construction that for each $i \geq 0$, $\psi_G^i(S) = \delta_H^i(S)$. Hence $\psi_G^*(S) = \delta_H^*(S)$. Now it follows from the fact that G is proper that $\psi_G(\alpha) = \alpha$ iff $\alpha \in V_T^*$. Hence from our construction, $\delta_H(\alpha) = \alpha$ iff $\alpha \in V_T^*$. So $A(H) = \{\alpha \in \delta_H^*(S) \mid \delta_H(\alpha) = \alpha\} = \delta_H^*(S) \cap V_T^*$. Hence from the property $L(G) = \psi_G^*(S) \cap V_T^*$ of our notation ψ_G, we have $A(H) = L(G)$. ■

We can now characterize the class A(0L) of adult languages of 0L systems in terms of the class L(CF) of context free languages.

Theorem 1 $A(0L) = L(CF)$.

Proof Immediate from Lemmas 6 and 7. ■

Let us say of two classes $\mathcal{L}_1$ and $\mathcal{L}_2$ of languages that $\mathcal{L}_1 \underset{\lambda}{=} \mathcal{L}_2$ if $\{L \cup \{\lambda\} \mid L \in \mathcal{L}_1\} = \{L \cup \{\lambda\} \mid L \in \mathcal{L}_2\}$. Then we have the following result for propagating 0L systems.

<u>Theorem 2</u> $A(P0L) \underset{\lambda}{=} L(CF)$.

<u>Proof</u> By Lemma 6, we have $A(P0L) \subset L(CF)$. Suppose $L \in L(CF)$. Then there is a proper CFG such that $L = L(G)$. It follows from Lemma 7 and the construction in its proof that we can construct a P0L system H such that $(L - \{\lambda\}) = A(H)$, i.e. such that $L = L(G) = A(H) \cup \{\lambda\}$. ■

Thus we have effective constructions which take us from any 0L system to a corresponding CFG, and vice versa. We have also shown that the propagating restriction makes little difference for adult languages of 0L systems, i.e. $A(0L) \underset{\lambda}{=} A(P0L)$. We shall see however that the propagating restriction is very important in 2L systems.

Adult Languages of 2L Systems

We now look at adult languages of 2L systems with and without the propagating restriction, and their relationship to the phrase structure languages of the Chomsky hierarchy.

<u>Lemma 8</u> There exists an algorithm which takes as input any grammar G and produces as output a 2L system H such that $L(G) = A(H)$. Moreover if G is a CSG, then H is a P2L system.

<u>Proof</u> Let $G = \langle V_N, V_T, P, S\rangle$ be a grammar. By Lemma 2 we may assume without loss of generality that if $\alpha \underset{P}{\rightarrow} \beta$ then $|\alpha| \in \{1, 2\}$ and that if $\alpha \underset{P}{\rightarrow} \lambda$ then $|\alpha| = 1$. We shall show how to construct from G a 2L system H such that $A(H) = L(G)$. The idea behind the construction is as follows.

Our construction will be such that if $S \overset{*}{\underset{G}{\Rightarrow}} \gamma$, where $\gamma = C_1C_2...C_n$ and $\gamma \notin L(G)$, then a string $\vec{C}_1C_2...C_n$ is derivable in H. The $\rightarrow$ will then move to the right along the string, allowing local rewriting according to the productions of P which have a single symbol on the left. When $\rightarrow$ reaches the right end of the string, it changes to $\leftarrow$. The $\leftarrow$ then moves to the left along the string, allowing local rewriting according to the productions of P which have two symbols on the left. When $\leftarrow$ reaches the left end of the string, it changes to $\rightarrow$ or to $\Rightarrow$. If the change is to $\rightarrow$, then the above process is repeated. If the change is to $\Rightarrow$ then two things can happen. If the string is in V_T^+, then $\Rightarrow$ moves all the way to the right and vanishes, yielding a string in A(H). If the string contains a symbol from V_N, then $\Rightarrow$ moves as far as that symbol, then changes to $\leftarrow$, and rewriting continues as above.

Formally, our construction of a 2L system H from the grammar $G = \langle V_N, V_T, P, S\rangle$ is as follows.

A) $V = V_N \cup V_T \cup \{X\}$, where X is a symbol not in $V_N \cup V_T$.

$V_g = V \cup \{g\}$, where g is a symbol not in V.

B) $\vec{V}$, $\overleftarrow{V}$, $\overset{\Rightarrow}{V}$ and $\hat{V}$ are mutually disjoint sets, which are individually disjoint from $V \cup \{g\}$, defined by

$$\vec{V} = \{\vec{A} \mid A \in V\}$$

$$\overleftarrow{V} = \{\overleftarrow{A} \mid A \in V\}$$

$$\overset{\Rightarrow}{V} = \{\overset{\Rightarrow}{A} \mid A \in V\}$$

$$\hat{V} = \{[C\gamma] \mid AB \underset{P}{\rightarrow} C\gamma \text{ where } A, B, C \in V \text{ and } \gamma \in V^*\}$$

C) $W = V \cup \vec{V} \cup \overleftarrow{V} \cup \overset{\Rightarrow}{V} \cup \hat{V}$

$W_g = W \cup \{g\}$.

D) $Q_1 = \{L\vec{A}B \rightarrow \gamma \mid A, B \in V, L \in V_g, \gamma \in V^*, \text{ and } A \underset{P}{\rightarrow} \gamma\}$

$Q_2 = \{L\vec{A}g \rightarrow \vec{C}\gamma \mid A, B, C \in V, \gamma \in V^*, \text{ and } A \underset{P}{\rightarrow} C\gamma\}$

$Q_3 = \{L\vec{A}g \rightarrow \overleftarrow{X} \mid L \in V_g, A \in V, A \underset{P}{\rightarrow} \lambda\}$

$Q_4 = \{L\overleftarrow{X}R \rightarrow \lambda \mid L, R \in V_g\}$

$Q_5 = \{LA\overleftarrow{B} \rightarrow [C\gamma] \mid A, B, C \in V, L \in V_g, \gamma \in V^*$ and

$AB \underset{P}{\rightarrow} C\gamma\}$

$Q_6 = \{L[C\gamma]B \rightarrow \overleftarrow{C} \mid B, C, \in V, L \in V_g$, and $[C\gamma] \in \hat{V}\}$

$Q_7 = \{[C\gamma]BR \rightarrow \gamma \mid B, C \in V, R \in V_g$, and $[C\gamma] \in \hat{V}\}$

$Q_8 = \{\overrightarrow{A}BR \rightarrow \overrightarrow{B} \mid A, B \in V$ and $R \in V_g\}$

$Q_9 = \{L\overrightarrow{A}B \rightarrow A \mid L \in V_g$ and $A, B \in V\}$

$Q_{10} = \{L\overrightarrow{B}g \rightarrow \overleftarrow{B} \mid L \in V_g$ and $B \in V_N\}$

$Q_{11} = \{LA\overleftarrow{B} \rightarrow \overleftarrow{A} \mid L \in V_g$ and $A, B \in V\}$

$Q_{12} = \{A\overleftarrow{B}R \rightarrow B \mid A, B \in V$ and $R \in V_g\}$

$Q_{13} = \{g\overleftarrow{A}R \rightarrow \overset{\Rightarrow}{A} \mid A \in V_T$ and $R \in V_g\}$

$Q_{14} = \{g\overleftarrow{A}R \rightarrow \overrightarrow{A} \mid A \in V$ and $R \in V_g\}$

$Q_{15} = \{L\overset{\Rightarrow}{A}R \rightarrow A \mid L, R \in V_g$ and $A \in V\}$

$Q_{16} = \{\overset{\Rightarrow}{A}BR \rightarrow \overset{\Rightarrow}{B} \mid A, B \in V_T$ and $R \in V_g\}$

$Q_{17} = \{\overset{\Rightarrow}{A}BR \rightarrow \overleftarrow{B} \mid A \in V, B \in V_N$ and $R \in V_g\}$

$Q_{18} = \{LAR \rightarrow A \mid L, R \in W_g, A \in W$, and there is no

$\gamma \in W^*$ such that $(LAR \rightarrow \gamma) \in \bigcup_{k=1}^{17} Q_k\}$

E) $Q = \bigcup_{k=1}^{18} Q_k$

F) $H = \langle W, \delta, g, S\rangle$, where δ is defined by Q.

H is a 2L system, since our construction is such that for each $LAR \in W_g W W_g$ there exists a $\gamma \in W^*$ such that $LAR \underset{Q}{\rightarrow} \gamma$.

From the construction it is straightforward to write out a detailed proof that $L(G) = A(H)$. (A full proof is given in Walker[†]).

It remains to be shown that if G is a CSG then H is propagating. Suppose G is a CSG. If Q contains a production of the form $LAR \rightarrow \lambda$, then by inspection this production is in $Q_1 \cup Q_4 \cup Q_7$. But then it follows from the construction that there is a

† Walker, A. D., Formal Grammars and the Stability of Biological Organisms, Ph.D. thesis, Department of Computer Science, State University of New York at Buffalo, 1974.

production $\alpha \underset{P}{\Rightarrow} \beta$ for which $|\alpha| > |\beta|$, a contradiction. ■

In the next lemmas we shall use the following notation. If M is an LBA we denote the language accepted by M as $L(M)$, and if T is a TM we denote the language accepted by T as $L(T)$.

Lemma 9 There exists an algorithm which takes as input any P2L system H and produces as output an LBA M such that $A(H) = L(M)$.

Proof Let $H = \langle V, \delta, g, S\rangle$ be a P2L system. Let M be an LBA constructed from H to operate as follows.

The tape of M has three tracks. If a string α is placed on the top track of the tape, M decides whether or not $\alpha \in L(M)$ in the following way.

(i) M tests whether or not $\delta(\alpha) = \alpha$. If so, M does (ii) below. If not, M rejects α and halts.

(ii) M writes S in the middle track and proceeds, nondeterministically, to see if $\alpha \in \delta^*(S)$, using the lower track as workspace. If M discovers that $\alpha \in \delta^*(S)$, then M accepts α and halts. If, in simulating a derivation $S = \alpha_0, \alpha_1, \ldots, \alpha_k$ where $\alpha_k \in \delta^k(S)$ M finds that $|\alpha_k| > |\alpha|$, M rejects α and halts.

From the above description it is a straightforward task to write down formally an algorithm which constructs M from H, and to show that $L(M) = A(H)$. ■

Lemma 10 There exists an algorithm which takes as input any 2L system H and produces as output a Turing machine T such that $A(H) = L(T)$.

Proof is similar to that of Lemma 9, except that in step (ii) there is no limit on the length of an intermediate string α_k. Hence not every computation by T terminates. However, because of the way in which L(T) is defined for a Turing machine T, it is the case that A(H) = L(T). ■

We can now characterize the classes A(P2L) of adult languages of P2L systems and A(2L) of adult languages of 2L systems in terms of the classes L(CS) of context sensitive languages and L(RE) of recursively enumerable languages.

Theorem 3 A(P2L) = L(CS).

Proof That A(P2L) $\subset$ L(CS) follows from Lemma 9 and the fact that for each LBA M there is a CSG G such that L(M) = L(G); see e.g. H & U, Theorem 8.2. It is immediate from Lemma 8 that L(CS) $\subset$ A(P2L). ■

Theorem 4 A(2L) = L(RE).

Proof That A(2L) $\subset$ L(RE) follows from Lemma 10 and the fact that for each TM T there is a grammar G such that L(T) = L(G); see e.g. H & U, Theorem 7.4. It is immediate from Lemma 8 that L(RE) $\subset$ A(2L). ■

This completes our characterization of 2L systems. We note that while the propagating restriction made little difference for 0L systems, in the sense that A(0L) $=_{\lambda}$ A(P0L), it makes a fundamental difference for 2L systems, since A(P2L) $\subsetneq$ A(2L).

Conclusions

Theorems 1 - 4 give us a satisfactory analysis of L systems from the point of view of the adult languages they generate, for they establish direct correspondences with three of the four main classes of languages in the Chomsky hierarchy. The remaining class is that of the regular languages, and it is an easy exercise to restrict the form of the productions of a 0L system to ensure that its adult language is regular. In Walker† it is shown that the result for 2L systems can be extended to $\langle k, \ell\rangle$L systems (see Herman and Rozenberg [45] for the definition of such systems) with $k + \ell \geq 1$, and that the result for P2L systems can be extended to P$\langle k, \ell\rangle$L systems with $k, \ell \geq 1$.

From the point of view of formal language theory, we have given a new characterization, by totally parallel grammars, of each of the classes of languages in the Chomsky hierarchy. From the point of view of biological model building, we have gained access to many of the established results of formal language theory.

Acknowledgements

The author wishes to thank Professors G. T. Herman, A. Lindenmayer, and G. Rozenberg for their help and encouragement. This work is supported by NSF Grant GJ 998 and NATO Research Grant 574.

---

† Walker, A. D., Formal Grammars and the Stability of Biological Organisms, Ph.D. thesis, Department of Computer Science, State University of New York at Buffalo, 1974.