JOURNAL OF COMPUTER AND SYSTEM SCIENCES 11, 252–261 (1971)

Formal Grammars and the Regeneration Capability of Biological Systems

ADRIAN WALKER*

Department of Computer Science, State University of New York at Buffalo, Amherst, New York 14226

Received August 19, 1974; revised January 23, 1975

A biological experiment to measure regeneration capability often involves cutting out a part or parts of an organism, and comparing the subsequent development of the damaged organism with that of a normal specimen of the same kind. Outside the laboratory, the natural environment of some organisms during development and adulthood can inflict a sequence of accidents from which, providing the accidents are not unusually severe, the organism is able to recover, at least partially.

A definition is proposed to place such damage and regeneration in the framework of an existing formal model for biological growth and development.

Some results linking the stable adult configurations of such models with the languages of the Chomsky hierarchy are reviewed. Then it is shown that, while the stable adult configurations which can be achieved by models without cellular interactions in the absence of damage correspond exactly to the context-free languages, the analogous sets of configurations obtained when damage is inflicted are regular languages. It is also shown that, in spite of the simplicity of the models without cellular interactions, there is no algorithm which will decide for any such model whether or not it is capable of complete regeneration. Finally, it is shown that cell death and replacement plays an important role in regeneration.

1. INTRODUCTION

We propose a definition which extends an existing biological model to allow us to describe the response of an organism to externally inflicted damage.

The model which we extend is called a Lindenmayer system (L system for short), and it is a formal description of strings of automata in which the individual automata can change state, divide into several new automata, or die out, in a way resembling the behavior of biological cells. L systems were first proposed by Lindenmayer [6], and have stimulated a substantial amount of research, much of which is reported by Herman and Rozenberg [3].

* Present address: Department of Computer Science, Hill Center for Mathematical Sciences, Rutgers University, New Brunswick, New Jersey 08903.

A notation is used in which just the states of the cells are written down for each string of cells. Thus if the states are represented by letters in an alphabet, then the transition from one string of cells to a successor string (which may be of different length) is written as a derivation of one string of letters from another by totally parallel replacement.

Our purpose is to extend the L system framework to cover situations in which an organism is damaged by some external agent, and then regenerates itself. The kind of damage which we model is the excision of some part of the organism. A biological experiment in regeneration often calls for the removal of some part of an organism, and then a comparison of the subsequent behavior of the damaged organism with an undamaged specimen of the same kind. For instance Wolpert, Hicklin and Hornbruch [9] describe a series of experiments of this kind to investigate the regenerative capability of hydra. An organism in its natural environment may, in general, be damaged at any stage of its development up to, and including, adulthood, and such damage may occur several times during the life span of an individual. Provided the damage is not too severe, many organisms show some ability to recover by self-regeneration.

For our purposes an L system has an initial string from which development starts, and a set of rules governing the behavior of the cells. To treat damage and regeneration, we use three concepts. First, we define a stable string to be a string of cells which renews itself dynamically, that is, the component cells continually change state, split to form new cells, or die out, but they do so in such a way as to maintain the string as a whole constant. Second, we are interested in the stable strings which are reachable through development, starting from the initial string, in ideal and damage-free conditions. We call these the adult strings of the system. Third, we deal with the stable strings which are reachable through development, starting from the initial string, if damage is inflicted by removing substrings of the initial string, the intermediate strings, or even of a stable string. At some point damage ceases, and we allow development to continue undisturbed to see if a stable string is reached. We call the strings reached in this way the regenerative adult strings of the system. This model of damage was suggested by Lindenmayer (private communication).

In general only some of the stable strings are reachable from the initial string under damage conditions (i.e., are regenerative adult strings), and only some of these are reachable from the initial string under ideal conditions (i.e., are adult strings). Thus we can say that a system shows regeneration capability if its regenerative adult strings and its adult strings are the same. On the other hand, suppose that every substring of an adult string is a stable string. Then no damage to an adult string is ever repaired, and the system shows less regeneration capability.

In the definitions which follow, an L system consists of an alphabet of cellular states, a finite substitution which indicates how one string is derived from another during development, and an initial string. The stable strings of the L system are

just those strings which map monogenically into themselves under the finite substitution. They form a set which we call the adult language of the L scheme (the L scheme consists of just the alphabet and finite substitution of the L system). Damage to a string is modeled by a sparse subword operator which generates, from a string, all possible substrings. The definition of the adult language of an L system makes use of the initial string, the finite substitution, and the set of stable strings. The definition of the regenerative adult language of an L system makes use of the initial string, the sparse subword operator to represent damage, the finite substitution, and the set of stable strings.

2. Notation and Definitions

If V is a finite nonempty set of symbols we call it an alphabet. If V is an alphabet we say that V^* is the set of all strings of symbols from V, including the string of length zero which we denote by λ. We say that $V^+ = V^* - \{\lambda\}$.

If A is a set, we denote the set of all subsets of A by 2^A.

If f is an operator with domain V^* and $L \subset V^*$, we say that $f(L) = \bigcup_{\alpha \in L} f(\alpha)$. If $f_1, \ldots, f_n$ are operators with domain V^* and $L \subset V^*$ we write $f_1 \cdots f_n(L)$ for $f_1(\cdots f_{n-1}(f_n(L)) \cdots)$. We make free use of regular expression notation for compositions of operators, e.g., $f^*(L)$ stands for $\bigcup_{i=0}^{\infty} f^i(L)$, $(f \cup g)(L)$ stands for $f(L) \cup g(L)$, etc.

We abbreviate context-free grammar as CFG, and we write the class of regular languages as $L(RG)$. We use the notion of a finite substitution as defined by Hopcroft and Ullman [5], and our notation for Chomsky grammars is theirs except when otherwise stated.

If δ is a finite substitution and $a \in V$, $\alpha \in V^*$ are such that $\alpha \in \delta(a)$, we shall write this as $a \to \alpha$, and we shall call $a \to \alpha$ a production. We shall define specific finite substitutions by means of sets of productions.

Definition (proper CFG). We say that a CFG $G = \langle V_N, V_T, P, S \rangle$ is *proper* if

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

Definition ($0L$ scheme). A $0L$ *scheme* is a 2-tuple $F = \langle V, \delta \rangle$, where V is an alphabet and δ is a finite substitution with domain V such that $\delta(\lambda) = \{\lambda\}$.

We define the *adult language of a* $0L$ *scheme* $F = \langle V, \delta \rangle$ as $A(F) = \{\alpha \in V^* \mid \delta(\alpha) = \{\alpha\}\}$. (In this, and other such cases, we shall often write $\delta(\alpha) = \beta$ instead of $\delta(\alpha) = \{\beta\}$).

DEFINITION (sparse subword operator \$). Let V be an alphabet. We define a *sparse subword operator* $\$: V^* \to 2^{V^*}$ as follows. $\$(\lambda) = \{\lambda\}$, for each $a \in V$ $\$(a) = \{\lambda, a\}$, and for each $a\beta$ where $a \in V$ and $\beta \in V^*$ $\$(a\beta) = \$(a)\,\$(\beta)$. \$ is extended to domain 2^{V^*} by $\$(K) = \bigcup_{\alpha \in K} \$(\alpha)$.

DEFINITION ($0L$ system). A $0L$ *system* is a 3-tuple $H = \langle V, \delta, \alpha \rangle$, where $F = \langle V, \delta \rangle$ is a $0L$ scheme and $\alpha \in V^*$ is called the initial string of H.

We define the *adult language of a* $0L$ *system* $H = \langle V, \delta, \alpha \rangle$ as $A(H) = \delta^*(\alpha) \cap A(F)$, where $F = \langle V, \delta \rangle$.

We define the *regenerative adult language of a* $0L$ *system* $H = \langle V, \delta, \alpha \rangle$ as $\tilde{A}(H) = (\delta \cup \$)^*(\alpha) \cap A(F)$, where $F = \langle V, \delta \rangle$.

We write $A(0L)$ for the class of adult languages of $0L$ systems, and $\tilde{A}(0L)$ for the class of regenerative adult languages of $0L$ systems.

3. REGENERATION IN $0L$ SYSTEMS

In this section we shall set out our results about the ability of L systems without interactions to regenerate their adult strings when these strings or some of their predecessors are damaged by removing one or more substrings. First, however, we give some examples to illustrate our definitions, and we review some results about adult languages which we shall need in our treatment of regeneration.

EXAMPLE 1. Let $V_1 = \{a, b, c, d, e\}$ and let δ_1 be a finite substitution defined by the set of productions $P_1 = \{a \to ab, b \to c, c \to \lambda, d \to dc, e \to e\}$. Let $F_1 = \langle V_1, \delta_1 \rangle$. Then F_1 is a $0L$ scheme. The adult language of F_1 is $A(F_1) = \{\alpha \in V_1^* \mid \delta_1(\alpha) = \alpha\}$, and some of the strings in $A(F_1)$ are *abce*, *edc*, and *dcabc*.

EXAMPLE 2. Let $V_2 = V_1 \cup \{s\}$ and let δ_2 be defined by the set of productions $P_1 \cup \{s \to \lambda, s \to as, s \to sd\}$. Let $H_2 = \langle V_2, \delta_2, s \rangle$. Then H_2 is a $0L$ system and $F_2 = \langle V_2, \delta_2 \rangle$ is a $0L$ scheme. The adult language of H_2 is $A(H_2) = \delta_2^*(s) \cap A(F_2) = \{(abc)^i(dc)^j \mid i, j \geqslant 0\}$. The regenerative adult language of H_2 is $\tilde{A}(H_2) = (\delta_2 \cup \$)^*(s) \cap A(F_2)$ and it is also equal to $\{(abc)^i(dc)^j \mid i, j \geqslant 0\}$. Hence for H_2 we have $A(H_2) = \tilde{A}(H_2)$.

EXAMPLE 3. Let $V_3 = V_2 \cup \{t, u\}$ and let δ_3 be defined by the set of productions $P_3 = P_1 \cup \{s \to d, s \to asd, s \to bte, t \to ue, u \to t\}$. Let $H_3 = \langle V_3, \delta_3, s \rangle$. Then H_3 is a $0L$ system. The adult language of H_3 is $A(H_3) = \{(abc)^i(dc)^{i+1} \mid i \geqslant 0\}$. The regenerative adult language of H_3 is $\tilde{A}(H_3) = \{(abc)^i e^j (dc)^k \mid i, j, k \geqslant 0\}$. Hence for H_3 we have $A(H_3) \subsetneqq \tilde{A}(H_3)$.

We note that in the $0L$ systems H_2 and H_3 the initial string is just a single symbol.

PROPOSITION 1. *For each* $0L$ *system* H *we can effectively construct a* $0L$ *system* H', *whose initial string is of length* 1, *such that* $A(H) = A(H')$ *and* $\tilde{A}(H) = \tilde{A}(H')$.

Proof. The proof is easy, and is omitted.

From now on we shall assume, without loss of generality, that each $0L$ system has a single symbol as its initial string.

In our treatment of regeneration we shall make use of the following results about CFG's and adult languages of $0L$ schemes and $0L$ systems.

LEMMA 1. *There exists an algorithm which takes as input any CFG* $G = \langle V_N, V_T, P, S\rangle$ *and produces as output a proper CFG* $G' = \langle V_N', V_T, P', S\rangle$ *such that* $L(G) = L(G')$.

Proof. Immediate from [1, Algorithms 2.8–2.11].

THEOREM 1. *There exists an algorithm which takes as input any* $0L$ *scheme* $F = \langle V, \delta\rangle$ *and produces as output a finite set* $W \subset V^*$ *such that* $A(F) = W^*$.

Proof. See [4].

THEOREM 2. *Let* $G = \langle V_N, V_T, P, S\rangle$ *be a proper CFG and let* $H = \langle V_N \cup V_T, \delta, S\rangle$ *be constructed from* G, *where* $P \cup \{a \to a \mid a \in V_T\}$ *is a set of productions which defines* δ. *Then* $L(G) = A(H)$.

Proof. See [4].

THEOREM 3. *The class of adult languages of* $0L$ *systems is equal to the class of context-free languages. Moreover, there exist algorithms which, given any* $0L$ *system, produce a corresponding CFG, and vice versa.*

Proof. See [4].

We now prove some detailed properties of the effect of damage on strings which represent stages in the development of a biological organism.

PROPOSITION 2. *Let* V *be an alphabet, let* δ *be a finite substitution with domain* V^*, *and let* $L \subset V^*$. *Then* $\delta\$(L) \subset \$\delta(L)$.

Proof. The proof is easy, and is omitted.

LEMMA 2. *Let* V *be an alphabet, let* δ *be a finite substitution with domain* V^*, *and let* $L \subset V^*$. *Then* $(\delta \cup \$)^*(L) \subset \$\delta^*(L)$.

Proof. Clearly it will be sufficient to show that for each $i \geqslant 0$, $(\delta \cup \$)^i(L) \subset \$\delta^*(L)$.

If $i = 0$ this relation holds in the form $L \subset \$(L)$. Suppose the relation holds for $i = 0,\ldots, j$. We have

$$\begin{aligned}
(\delta \cup \$)^{j+1}(L) &= (\delta \cup \$)(\delta \cup \$)^j(L), \\
&\subset (\delta \cup \$)\,\$\delta^*(L), && \text{by the inductive hypothesis,} \\
&= \delta\$\delta^*(L) \cup \$\$\delta^*(L), \\
&= \delta\$\delta^*(L) \cup \$\delta^*(L), && \text{since } \$\$\delta^*(L) = \$\delta^*(L), \\
&\subset \$\delta\delta^*(L) \cup \$\delta^*(L), && \text{by Proposition 2,} \\
&= \$\delta^+(L) \cup \$\delta^*(L), \\
&= \$\delta^*(L), && \text{since } \$\delta^+(L) \subset \$\delta^*(L).
\end{aligned}$$

This completes an inductive proof of the lemma.

LEMMA 3. *If $H = \langle V, \delta, s\rangle$ is a $0L$ system then $\tilde{A}(H) = \$\delta^*(s) \cap A(F)$, where $F = \langle V, \delta\rangle$.*

Proof. By definition we have $\tilde{A}(H) = (\delta \cup \$)^*(s) \cap A(F)$. By Lemma 2 we have $(\delta \cup \$)^*(s) \subset \$\delta^*(s)$, so $(\delta \cup \$)^*(s) \cap A(F) \subset \$\delta^*(s) \cap A(F)$, i.e., $\tilde{A}(H) \subset \$\delta^*(s) \cap A(F)$.

On the other hand it is clear that $\$\delta^* \subset (\delta \cup \$)^*$, hence $\$\delta^*(s) \subset (\delta \cup \$)^*(s)$ and so $\$\delta^*(s) \cap A(F) \subset (\delta \cup \$)^*(s) \cap A(F) = \tilde{A}(H)$, which completes our proof of the lemma.

The above lemma together with the following theorem will allow us to establish that every regenerative adult language of a $0L$ system is a regular language.

THEOREM 4 (Haines). *If V is an alphabet and $L \subset V^*$ then $\$L \in L(RG)$.*

Proof. See [2, Theorem 3].

LEMMA 4. $\tilde{A}(0L) \subset L(RG)$.

Proof. Let $L \in \tilde{A}(0L)$. Then there exists a $0L$ system $H = \langle V, \delta, s\rangle$ such that $L = \tilde{A}(H)$. Let $F = \langle V, \delta\rangle$. Then by Lemma 3 we have $\tilde{A}(H) = \$\delta^*(s) \cap A(F)$. Since $\delta^*(s) \subset V^*$ we have by Theorem 4 that $\$\delta^*(s) \in L(RG)$. By Theorem 1 we have that $A(F) = W^*$ for some finite set $W \subset V^*$. Since every finite set is regular and $L(RG)$ is closed under $*$ we have $A(F) \in L(RG)$ also. Since $L(RG)$ is closed under intersection, we have $\$\delta^*(s) \cap A(F) = \tilde{A}(H) \in L(RG)$, which completes our proof.

We note that the proof of Lemma 4 is nonconstructive, i.e., we do not give an algorithm which for any $0L$ system H constructs a regular grammar G such that $\tilde{A}(H) = L(G)$. This is because the proof of Haines' theorem is itself nonconstructive.

LEMMA 5. *There is a language $L \in L(RG) - \tilde{A}(0L)$.*

Proof. Let $L = \{a, a^3\}$. Clearly $L \in L(RG)$. Suppose $L \in \tilde{A}(0L)$. Then there is a $0L$ system $H = \langle V, \delta, s\rangle$ such that $L = \tilde{A}(H)$. By Lemma 3 we may write $\tilde{A}(H) = \$\delta^*(s) \cap A(F)$, where $F = \langle V, \delta\rangle$. By Theorem 1 $A(F) = W^*$ for some finite set W. Since $a \in L$, $a \in \tilde{A}(H)$, so $a \in W$. But then $a^2 \in W^*$, i.e., $a^2 \in A(F)$, and since $a^3 \in \tilde{A}(H)$ it is easy to see that $a^2 \in \tilde{A}(H)$ also, contradicting $L = \tilde{A}(H)$. So $L \in L(RG) - \tilde{A}(0L)$, which completes our proof.

We are now in a position to characterize the class $\tilde{A}(0L)$ of regenerative adult languages of $0L$ systems.

THEOREM 5. $\tilde{A}(0L) \subsetneqq L(RG)$.

Proof. Immediate from Lemmas 4 and 5.

Thus every regenerative adult language of a $0L$ system is regular. We recall that, by Theorem 3, the class of adult languages of $0L$ systems is exactly the class of context-free languages. The question naturally arises whether we can decide for each $0L$ system whether or not its adult and regenerative adult languages are equal. To settle this question, we shall make use of the following lemma.

LEMMA 6. *There exists an algorithm which takes as input any proper CFG* $G = \langle V_N, V_T, P, S\rangle$ *and produces as output a* $0L$ *system* H *such that* $L(G) = A(H)$ *and* $V_T^* = \tilde{A}(H)$.

Proof. Let $G = \langle V_N, V_T, P, S\rangle$ be a proper CFG. By Theorem 2 we can effectively find a $0L$ system $H' = \langle V', \delta', s'\rangle$ such that $L(G) = A(H')$. Moreover, the fact that G is proper implies that $\tilde{A}(H') \subset V_T^*$. To see this, recall that H is constructed from G by direct use of the productions P of G together with a production $a \to a$ for each $a \in V_T$. Hence if $\alpha \in \tilde{A}(H')$ we have $\delta'(\alpha) = \alpha$ which, since G is proper and H' uses the productions of G, implies that $\alpha \in V_T^*$.

Let Q' be the set of productions of H', and let $H = \langle V' \cup \{s, X\}, \delta, s\rangle$ be a system with productions Q defined by $Q = Q' \cup \{s \to s', s \to X\} \cup R$, where $R = \{X \to Xa \mid a \in V_T\}$.

Obviously H is a $0L$ system, and it is easy to see that $L(G) = A(H)$. It follows from the construction of H and from the fact that $\tilde{A}(H') \subset V_T^*$ that $\tilde{A}(H) \subset V_T^*$. Clearly $\lambda \in \tilde{A}(H)$. Suppose $\alpha \in V_T^+$. Then from R and the production $s \to X$ we have that $X\alpha \in \delta^*(s) \subset (\delta \cup \$)^*(s)$, and from the construction we have $\delta(\alpha) = \alpha$. So $\alpha \in (\delta \cup \$)(X\alpha)$. Thus altogether, $\alpha \in (\delta \cup \$)^*(s)$ and $\delta(\alpha) = \alpha$, so $\alpha \in \tilde{A}(H)$. We have shown that $V_T^* \subset \tilde{A}(H)$.

Thus H is a $0L$ system such that $L(G) = A(H)$ and $V_T^* = \tilde{A}(H)$, which completes our proof of the lemma.

We are now in a position to show that there does not exist an algorithm which

decides for any $0L$ system whether its adult and regenerative adult languages are equal.

THEOREM 6. *There does not exist an algorithm which takes as input any* $0L$ *system* H *and decides whether or not* $A(H) = \tilde{A}(H)$.

Proof. By Lemma 1 there exists an algorithm which takes as input any CFG G and produces as output a proper CFG G' such that $L(G) = L(G')$. Let us call this Algorithm A. We note that if $G' = \langle V_N', V_T, P', S\rangle$ is obtained from $G = \langle V_N, V_T, P, S\rangle$ by Algorithm A, then $L(G') = V_T^*$ iff $L(G) = V_T^*$.

By Lemma 6 there exists an algorithm which takes as input any proper CFG $G' = \langle V_N', V_T, P', S\rangle$ and produces as output a $0L$ system H such that $L(G') = A(H)$ and $V_T^* = \tilde{A}(H)$. Let us call this Algorithm B.

Suppose the contrary of theorem, i.e., that there exists an algorithm which takes as input any $0L$ system H and decides whether or not $A(H) = \tilde{A}(H)$. Let us call this Algorithm C.

If we combine Algorithms A and B with our hypothetical Algorithm C, then we have an algorithm which, for an arbitrary CFG $G = \langle V_N, V_T, P, S\rangle$, decides whether or not $L(G) = V_T^*$. But, by [5, Theorem 14.4], this question is undecidable. Thus we have a contradiction, which completes our proof of the theorem.

Thus the situation is that there exist $0L$ systems H for which $A(H) = \tilde{A}(H)$ and others for which $A(H) \neq \tilde{A}(H)$ (see Examples 2 and 3, respectively), but we cannot decide for an arbitrary $0L$ system whether this equality holds or not. Thus while some $0L$ systems show strong regeneration in the face of damage to their strings, we cannot effectively partition the class of $0L$ systems into those which have strong regeneration and those which do not. Of course this result also holds for the wider class consisting of L systems with cellular interactions.

We can show that cell death and replacement plays an important role in regeneration in the following sense. If there is no cell death in a $0L$ system H, i.e., H is a $P0L$ system, then the regenerative capacity of H is poor in that all damaged versions of strings in the adult language of H are in the regenerative adult language of H. Since any string in $\tilde{A}(H)$ derives only itself, this implies that no damage to an adult string is ever repaired. On the other hand there are $0L$ systems with cell death and replacement in which the adult strings do show some regenerative capacity. We establish this as follows.

THEOREM 7. *If* H *is a* $P0L$ *system then* $\$A(H) \subset \tilde{A}(H)$. *On the other hand there exists a* $0L$ *system* H *for which it is not the case that* $\$A(H) \subset \tilde{A}(H)$.

Proof. Let $H = \langle V, \delta, s\rangle$ be a $P0L$ system. Then $F = \langle V, \delta\rangle$ is a $P0L$ scheme. By Theorem 1 we can find a finite set $W \subset V^*$ such that $A(F) = W^*$. Let us write

$\Sigma = \{b \in V \mid$ there exist $\alpha, \gamma \in V^*$ such that $\alpha b \gamma \in W\}$. It is not difficult to show that for each $b \in \Sigma$, $\delta(b) = b$, and hence that $W = \Sigma$. So $A(H) = \delta^*(s) \cap \Sigma^*$ and $\$A(H) = \$(\delta^*(s) \cap \Sigma^*)$. Let $\beta \in \$A(H)$. Then there is an $\alpha \in (\delta^*(s) \cap \Sigma^*)$ such that $\beta \in \$(\alpha)$. But then $\alpha \in \delta^*(s)$ so $\beta \in \$\delta^*(s)$ and $\beta \in \Sigma^*$. Hence $\$A(H) = \$(\delta^*(s) \cap \Sigma^*) \subset \$\delta^*(s) \cap \Sigma^*$. By Lemma 3 we have $\$\delta^*(s) \cap \Sigma^* = \tilde{A}(H)$, and so $\$A(H) \subset \tilde{A}(H)$.

Let H_2 be the $0L$ system of Example 2. Then $A(H_2) = \tilde{A}(H_2) = \{(abc)^i(dc)^j \mid i, j \geqslant 0\}$. So if we set $\alpha = aacc$ we have $\alpha \in \$A(H_2) - \tilde{A}(H_2)$. Thus it is not the case that $\$A(H_2) \subset \tilde{A}(H_2)$, which completes our proof of the theorem.

4. Conclusions

There are many detailed manifestations of the capability of biological organisms to regenerate themselves when damage is inflicted on them, and this capability is of interest to computer scientists who would like to design damage resistant computing systems.

In this paper we have taken an existing formal framework for the study of biological systems, and added to it definitions which allow us to compare the sets of stable adult configurations which can be reached in ideal conditions with those that can be reached when damage is inflicted. We have concentrated on systems without cellular interactions. Even though these systems appear simple, we have shown that they can achieve nontrivial regenerative behavior through the mechanism of cell death and replacement. We have also demonstrated that while some such systems show complete regeneration in the face of damage and others do not, there is no way to decide for an arbitrary given system whether it will show complete regeneration.

Of particular interest is the fact that the sets of stable adult configurations achieved in ideal conditions by systems without cellular interactions correspond exactly to the context-free languages, whereas if damage is inflicted then the sets of stable configurations reached are regular languages, and not all regular languages are reachable in this way.

Acknowledgments

The author would like to thank Professors Gabor T. Herman, Jan van Leeuwen, and Aristid Lindenmayer for helpful discussions. This research is supported by NSF Grant GJ 998.

References

1. A. V. Aho and J. D. Ullman, "The Theory of Parsing, Translating and Compiling," Vol. 1, Prentice-Hall, Englewood Cliffs, N.J., 1972.

2. L. H. HAINES, On free monoids partially ordered by embedding, *J. Combinatorial Theory* **6** (1969), 94–98.
3. G. T. HERMAN AND G. ROZENBERG, "Developmental Systems and Languages," North-Holland, Amsterdam, 1974.
4. G. T. HERMAN AND A. WALKER, Context free languages in biological systems, *Internat. J. Comput. Math.* **5** (1974).
5. J. E. HOPCROFT AND J. D. ULLMAN, "Formal Languages and Their Relation to Automata," Addison–Wesley, Reading, Mass., 1969.
6. A. LINDENMAYER, Mathematical models for cellular interactions in development, I, II, *J. Theoret. Biol.* 18 (1968), 280–315.
7. A. WALKER, Adult languages of L systems and the Chomsky Hierarchy: Topics in L Systems (G. Rozenberg and A. Salomaa, Eds.), Springer–Verlag, Berlin, 1974.
8. A. WALKER, Formal grammars and the stability of biological organisms, Ph.D. Thesis, State University of New York at Buffalo, 1974.
9. L. WOLPERT, J. HICKLIN, AND A. HORNBRUCH, Positional information and pattern regulation in regeneration of hydra, *in* "Control Mechanisms of Growth and Differentiation," pp. 391–415, Symposia of the Society for Experimental Biology, No. 25, Cambridge University Press, London/New York, 1971.