A SURVEY OF MATHEMATICAL THEORY OF L SYSTEMS

THEORY OF L SYSTEMS:

FROM THE POINT OF VIEW OF FORMAL LANGUAGE THEORY

by

G. Rozenberg

Institute of Mathematics
Utrecht University
Utrecht - De Uithof
Holland

and

Department of Mathematics
University of Antwerp. UIA
Wilrijk
Belgium

CONTENTS

0. INTRODUCTION

The theory of L systems originated from the work of Lindenmayer [59,60]. The original aim of this theory was to provide mathematical models for the development of simple filamentous organisms. At the beginning L systems were defined as linear arrays of finite automata, later however they were reformulated into the more suitable framework of grammar-like constructs. From then on, the theory of L systems was developed essentially as a branch of formal language theory. In fact it constitutes today one of the most vigorously investigated areas of formal language theory.

In this paper we survey the mathematical theory of L systems. As to the biological aspects of the theory we refer the reader to an excellent paper by Lindenmayer ("Developmental systems and languages in their biological context" a contribution to the book Herman and Rozenberg [45]).

This paper is organized in such a way that it discusses several typical problem areas and the results obtained therein. The results quoted here may not always be the most important ones but they are quite representative for the direction of research in this theory. It is rather unfortunate that we have no space here to discuss the basic techniques for solving problems in this theory, but information about these can be found in the listed references. As the most complete source of readings on L systems the book Herman and Rozenberg [45] is recommended to the reader.

In this paper we assume the reader to be familiar with basic formal language theory, e.q. with the scope of the book "Formal Languages and their relation to automata" by J. Hopcroft and J. Ullman, Addison-Wesley, 1969. We shall also freely use standard formal language notation and terminology. (Perhaps the only unusual term used in this paper is "coding" which means a letter-to-letter homomorphism).

We also want to remark that this survey is of informal character, meaning that quite often concepts are introduced in a not entirely rigorous manner, and results are presented in a descriptive way rather than in a form of very precise mathematical statements. This was dictated by both the limited size of the paper and by the profile of its experted reader. We hope that this does not decrease the usefulness of this paper.

Finally, I would like to state that this survey is by no means exhaustive and the selection of topics and results presented reflects my personal point of view.

1. L SCHEMES AND L SYSTEMS

In this section we give definitions and examples of basic objects (the so called L schemes and L systems) to be discussed in this paper. We start with the most general class, the so called TIL schemes and TIL systems. (They were introduced in K.P. Lee and G. Rozenberg "TIL systems and languages" [submitted for publication]). TIL systems are intended to model the development of multicellular filamentous organisms in the case when an interaction can take place among the cells and the environment can be subject to changes.

Definition 1.1. Let $k,l \in N$. An L scheme with tables and with $< k,l >$ interactions (abbreviated $T < k,l >$ L scheme) is a construct $S = < \Sigma, \mathcal{P}, g >$ where Σ is a finite nonempty set (the alphabet of S), g is a symbol which is not in Σ (the masker of S), $\mathcal{P}$ is a finite nonempty set, each dement P of which (called a table of S) is a finite nonempty relation satisfying the following:

$$P \subseteq \bigcup_{\substack{i,j,m,n \geq 0 \\ i+j = k \\ m+n = l}} \{g^i\}\Sigma^j \times \Sigma \times \Sigma^m\{g^n\} \times \Sigma^*$$

and for every $< \alpha,a,\beta >$ in $\bigcup_{\substack{i,j,m,n \geq 0 \\ i+j = k \\ m+n = l}} \{g^i\}\Sigma^j \times \Sigma \times \Sigma^m\{g^n\}$ there exists a γ in Σ^* such that $< \alpha,a,\beta,\gamma > \in P$.

(Each element of P is called a production).

Definition 1.2. Let $S = < \Sigma,\mathcal{P},g >$ be a $T < k,l >$ L scheme. We say that S is:

1) an L scheme with $< k,l >$ interactions (abbreviated $< k,l >$ L scheme) if $\# \mathcal{P} = 1$.

2) an L scheme with tables and without interactions (abbreviated T0L scheme) if $k = l = 0$.

3) an L scheme without interactions (abbreviated 0L scheme) if both $\# \mathcal{P} = 1$ and $k = l = 0$.

Definition 1.3. A construct $S = < \Sigma,\mathcal{P},g >$ is called a TIL scheme (IL scheme) if, for some $k,l \in N$, S is a $T < k,l > L$ scheme ($< k,l > L$ scheme).

Definition 1.4. Let $S = < \Sigma, \mathcal{P}, g >$ be a $T < k,l > L$ scheme. Let $x = a_1 \dots a_n \in \Sigma^*$, with $a_1, \dots, a_n \in \Sigma$, and let $y \in \Sigma^*$. We say that x directly derives y in S (denoted as $x \underset{S}{\Longrightarrow} y$) if $y = \gamma_1 \dots \gamma_n$ for some $\gamma_1, \dots, \gamma_n$ in Σ^* such that, there exists a table P in Σ and for every i in $\{1,\dots,n\}$ P contains a production of the form $< \alpha_i, a_i, \beta_i, \gamma_i >$ where α_i is the prefix of $g^k a_1 \dots a_{i-1}$ of length k and β_i is the suffix of $a_{i+1} \dots a_n g^l$ of length l. The transitive and reflexive closure of the relation $\underset{S}{\Longrightarrow}$ is denoted as $\overset{*}{\underset{S}{\Longrightarrow}}$ (when $x \overset{*}{\underset{S}{\Longrightarrow}} y$ then we say that x derives y in S).

Definition 1.5. A TIL system (IL system) is an ordered pair $G = < S, \omega >$ where S is a TIL scheme (an IL scheme) and ω is a word over the alphabet of S. The scheme S is called the underlying scheme of G and is denoted as S(G). G is called a $T < k,l > L$ system (a $< k,l > L$ system, a TOL system, a OL system) if S(G) is a $T < k,l > L$ scheme (a $< k,l > L$ scheme, a TOL scheme, a OL scheme).

IL systems in restricted form originated from Lindenmayer [59,60]; in the form they are discussed here they were introduced in Rozenberg [86,87]. TOL systems were introduced in Rozenberg [81] and OL systems were introduced in Lindenmayer [61] and Rozenberg and Doucet [91].

Definition 1.6. Let $G = < S, \omega >$ be a TIL system. Let $x,y \in \Sigma^*$. We say that x directly derives y in G, denoted as $x \underset{G}{\Longrightarrow} y$ (x derives y in G denoted as $x \overset{*}{\underset{G}{\Longrightarrow}} y$) if $x \underset{S}{\Longrightarrow} y$ ($x \overset{*}{\underset{S}{\Longrightarrow}} y$).

Notation. It is customary to omit the marker g from the specification of a TOL system. If S is an IL or a OL scheme (system) such that $\# \mathcal{P} = 1$, say $\mathcal{P} = \{P\}$, then in the specification of S we put P rather than $\{P\}$. Also to avoid cumbersome notation in specifying a TIL system G we simply extend the n-tuple specifying S(G) to an (n+1)-tuple where the last element is the axiom of G. (In this sense we write, e.g., $G = < \Sigma, \mathcal{P}, g, \omega >$ rather than $G = \ll \Sigma, \mathcal{P}, g >, \omega >$). In specifying productions in a table of a given TIL systems one often omits those which clearly cannot be used in any rewriting process which starts with the axiom of the system. If $< \alpha, a, \beta, \gamma >$ is a production in a TIL scheme (system) then it is usually written in the form $< \alpha, a, \beta > \to \gamma$ (where $< \alpha, a, \beta >$ is called its left-hand side and γ is called its right-hand side). When the productions of a TOL scheme (system) are being specified, then we write $a \to \gamma$ rather than $< \Lambda,a,\Lambda > \to \gamma$.

Example 1.1. Let $\Sigma = \{a,b\}$, $P_1 = \{< g,a,\Lambda > \to a^3, < a,a,\Lambda > \to a, < a,b,\Lambda > \to b^2, < b,b,\Lambda > \to b^2, < b,a,\Lambda > \to a\}$, $P_2 = \{< g,a,\Lambda > \to a^4, < a,a,\Lambda > \to a, < a,b,\Lambda > \to b^3, < b,b,\Lambda > \to b^3, < b,a,\Lambda > \to a\}$ and $\omega = a^5 b^6 a$. Then $G = < \Sigma,\{P_1,P_2\},g,\omega >$ is a $T < 1,0 > L$ system.

Example 1.2. Let $\Sigma = \{a,b\}$, $P = \{< a,a,\Lambda > \to a^2, < b,a,\Lambda > \to a^2, < g,a,\Lambda > \to a, < a,b,\Lambda > \to b^2, < b,b,\Lambda > \to b^2, < g,b,\Lambda > \to b^2, < g,b,\Lambda > \to ab^2\}$ and $\omega = ba$. Then $G = < \Sigma,P,g,\omega >$ is a $< 1,0 > L$ system.

Example 1.3. Let $\Sigma = \{a,b\}$, $P_1 = \{a \to a^2, b \to b^2\}$, $P_2 = \{a \to a^3, b \to b^3\}$ and $\omega = ab$. Then $G = < \Sigma,\{P_1,P_2\},\omega >$ is a TOL system.

Example 1.4. Let $\Sigma = \{A,\bar{A},a,B,\bar{B},b,C,\bar{C},c,F\}$, $P = \{A \to A\bar{A},\ A \to a,\ B \to B\bar{B},\ B \to b,\ C \to C\bar{C},\ C \to c,\ \bar{A} \to \bar{A},\ \bar{A} \to a,\ \bar{B} \to \bar{B},\ \bar{B} \to b,\ \bar{C} \to \bar{C}, \bar{C} \to c,\ a \to F,\ b \to F,\ c \to F,\ F \to F\}$ and $\omega = ABC$. Then $G = < \Sigma,P,\omega >$ is a OL system.

2. SQUEEZING LANGUAGES OUT OF L SYSTEMS

There are several ways that one can associate the language with a given word-generating device. In this section we shall discuss several ways of defining languages by L systems.

2.1. Exhaustive approach.

Given an L system G (with alphabet Σ and axiom ω) it is most natural to define its language, denoted L(G), as the set of all words (axiom included) that can be derived from ω in G; hence $L(G) = \{x \in \Sigma^* : \omega \overset{*}{\underset{G}{\Rightarrow}} x\}$.

Example 2.1.1. The language of a $T < 1,0 > L$ system G from Example 1.1 is $\{a^{2n+3m} b^{2^n 3^m} a : n,m \geq 1\}$. The language of a TOL system from Example 1.3 is $\{a^{2^n 3^m} b^{2^n 3^m} : n,m \geq 0\}$.

The languages obtained in this way from OL, TOL, TIL and IL systems are called OL, TOL, TIL and IL languages respectively. (Their classes will be denoted by $\mathcal{L}(OL)$, $\mathcal{L}(TOL)$, $\mathcal{L}(TIL)$ and $\mathcal{L}(IL)$ respectively). For $k,l \geq 0$, a $< k,l > L$ language (a $T < k,l > L$ language) is a language generated by a $< k,l > L$ system (a $T < k,l > L$ system).

One may notice here two major differences in generating languages by

0L and IL systems on the one hand and context-free and type 0 grammars on the other. 0L and IL systems do not use nonterminal symbols while context-free and type-0 grammars use them. Rewriting in 0L and IL systems is absolutely parallel (all occurrences of all letters in a word are rewritten in a single derivation step) while rewriting in context-free and type-0 grammars is absolutely sequential (only one occurrence of one symbol is rewritten in a single derivation step).

2.2. Using nonterminals to define languages.

The standard step in formal language theory to define the language of a generating system is to consider not the set of all words generated by it but only those which are over some distinguished (usually called terminal) alphabet. In this way one gets the division of the alphabet of a given system into the set of terminal and nonterminal (sometimes also called auxiliary) symbols. In the case of L systems such an approach gives rise to the following classes of systems.

Definition 2.2.1. An *extended 0L*, (*T0L*, *IL*, *TIL*) *system*, abbreviated E0L (ET0L, EIL, ETIL) system, is a pair $G = \langle H, \Delta \rangle$, where H is a 0L (T0L, IL, TIL) system and Δ is an alphabet (called the *target alphabet of G*).

Definition 2.2.2. The *language* of an E0L (ET0L, EIL, ETIL) system $G = \langle H, \Delta \rangle$, denoted as L(G), is defined by $L(G) = L(H) \cap \Delta^*$.

An E0L (ET0L, EIL, ETIL) system $G = \langle H, \Delta \rangle$ is usually specified as $\langle \Sigma, P, \omega, \Delta \rangle$ ($\langle \Sigma, \mathcal{P}, \omega, \Delta \rangle$, $\langle \Sigma, P, g, \omega, \Delta \rangle$, $\langle \Sigma, \mathcal{P}, g, \omega, \Delta \rangle$) where $\langle \Sigma, P, \omega \rangle$($\langle \Sigma, \mathcal{P}, \omega \rangle$, $\langle \Sigma, P, g, \omega \rangle$, $\langle \Sigma, \mathcal{P}, g, \omega \rangle$) is the specification of H itself.

Example 2.2.1. Let $G = \langle \Sigma, P, \omega, \Delta \rangle$, where Σ, P, ω are specified as in Example 1.4 and $\Delta = \{a,b\}$. Then $L(G) = \{a^n b^n c^n : n \geq 1\}$.

If K is the language of an E0L (ET0L, EIL, ETIL) system, then it is called an *E0L* (*ET0L*, *EIL*, *ETIL*) language. The classes of E0L languages, ET0L languages, EIL languages and ETIL languages are denoted by $\mathcal{L}$(E0L), $\mathcal{L}$(ET0L), $\mathcal{L}$(EIL) and $\mathcal{L}$(ETIL) respectively.

E0L systems and languages are discussed in Herman [35]; ET0L systems and languages were introduced in Rozenberg [89]; EIL systems and languages are discussed e.g. in van Dalen [12] and Rozenberg [86,87]; ETIL systems and languages were introduced in "TIL systems and languages" by

K.P. Lee and G. Rozenberg.

It is very instructive at this point to notice that, as far as generation of languages is concerned, the difference between E0L and EIL systems on one hand and context-free and type-0 grammars on the other hand is the absolutely parallel fashion of rewriting in E0L and EIL systems and the absolutely sequential fashion of rewriting in context-free and type-0 grammars.

2.3. Using codings to define languages.

When we make observations of a particular organism and want to describe it by strings of symbols, we first associate a symbol to each particular cell. This is done by dividing cells into a number of types and associating the same symbol to all the cells of the same type. It is possible that the development of the organism can be described by a developmental system, but the actual system describing it uses a finer subdivision into types that we could observe. This is often experimentally unavoidable. In this case, the set of strings generated by a given developmental system is a coding of the "real" language of the organism which the given developmental system describes. Considering codings for defining languages of L systems gives rise to the following classes of systems.

Definition 2.3.1. A 0L (T0L, IL, TIL) system with coding, abbreviated C0L (CT0L, CIL, CTIL) system, is a pair $G = \langle H,h \rangle$, where H is a 0L (T0L, IL, TIL) system and h is a coding.

Definition 2.3.2. The language of a C0L (CT0L, CIL, CTIL) system $G = \langle H,h \rangle$, denoted as L(G), is defined by $L(G) = h(L(H))$.

Example 2.3.1. Let $H = \langle \{a,b\},\{a \to a^2, b \to b\},ba \rangle$ and h be a coding from $\{a,b\}$ into $\{a,b\}$ such that $h(a) = h(b) = a$. Then $L(\langle H,h \rangle) = \{a^{2^{n+1}} : n \geq 0\}$.

If K is the language of a C0L (CT0L, CIL, CTIL) system, then it is called a C0L (CT0L, CIL, CTIL) language. The classes of C0L, CT0L, CIL and CTIL languages are denoted by $\mathcal{L}$(C0L), $\mathcal{L}$CT0L), $\mathcal{L}$(CIL) and $\mathcal{L}$(CTIL) respectively.

Using codings to define languages of various classes of L systems was considered, e.g., in Culik and Opatrny [10], Ehrenfeucht and

Rozenberg [20,25,27] and Nielsen, Rozenberg, Salomaa, Skyum [71,72].

2.4. Adult languages of L systems.

An interesting way of defining languages by L systems was proposed by A. Walker (see [47] and [118]). Based on biological considerations concerning problems of regulation in organisms, one defines the adult language of an L system G, denoted as A(G), to be the set of all these words from L(G) which derive (in G) themselves and only themselves. Thus we can talk about adult 0L languages, adult T0L languages, adult IL languages and adult TIL languages (their families are denoted by symbols $\mathcal{L}_A(0L)$, $\mathcal{L}_A(T0L)$, $\mathcal{L}_A(IL)$ and $\mathcal{L}_A(TIL)$ respectively).

Example 2.4.1. Let $G = \langle \Sigma, P, \omega \rangle$ be a 0L system such that $\Sigma = \{a,b\}$, $P = \{a \to \Lambda, a \to ab, b \to b\}$ and $\omega = a$. Then $A(G) = \{b^n : n \geq 0\}$.

In the sequel we shall use the term L language to refer to any one of the types of language introduced in this section.

2.5. Comparing the language generating power of various mechanisms for defining L languages.

Once several classes of language generating devices are introduced one is interested in comparing their language generating power. This is one of the most natural and most traditional topics investigated in formal language theory. In the case of L systems we have, for example, the following results.

Theorem 2.5.1. (see, e.g., Herman and Rozenberg [45]).
1) For X in {0L, T0L, IL, TIL}, $\mathcal{L}(X) \subsetneq \mathcal{L}(EX)$.
2) For X in {0L, T0L, IL, TIL}, $\mathcal{L}(X) \subsetneq \mathcal{L}(CX)$.
3) $\mathcal{L}(0L)$ is incomparable but not disjoint with $\mathcal{L}_A(0L)$.

Theorem 2.5.2. (Ehrenfeucht and Rozenberg [20,27], Herman and Walker [(47]).
1) $\mathcal{L}(E0L) = \mathcal{L}(C0L)$ and $\mathcal{L}(ET0L) = \mathcal{L}(CT0L)$.
2) $\mathcal{L}_A(0L) \subsetneq \mathcal{L}(E0L)$.

3. FITTING CLASSES OF L LANGUAGES INTO KNOWN FORMAL LANGUAGE THEORETIC FRAMEWORK

The usual way of understanding the language generating power of a class of generative systems is by comparing them with the now classical Chomsky hierarchy. (One reason for this is that the Chomsky hierarchy is probably the most intensively studied in formal language theory.) In the area of L languages we have, for example, the following result. (In what follows $\mathcal{L}$(RE) denotes the class of recursively enumerable languages, $\mathcal{L}$(CS) denotes the class consisting of every L such that either L or L-{Λ} is a context-sensitive language, and $\mathcal{L}$(CF) denotes the class of context-free languages.)

Theorem 3.1. (van Dalen [12], Rozenberg [89], Herman [35]).
$\mathcal{L}(EIL) = \mathcal{L}(RE)$, $\mathcal{L}(ET0L) \subsetneq \mathcal{L}(CS)$ and $\mathcal{L}(CF) \subsetneq \mathcal{L}(E0L)$.

Note that this theorem compares classes of systems all of which use nonterminals for defining languages. Thus the only real difference (from the language generation point of view) between (the classes of) EIL, ET0L and E0L systems on the one hand and (the classes of) type-0, context-sensitive and context-free grammars respectively on the other hand is the parallel versus sequential way of rewriting strings. In this sense the above results tell us something about the role of parallel rewriting in generating languages by grammar-like devices. In the same direction we have another group of results of which the following two are quite representative.

Theorem 3.2. (Lindenmayer [61], Rozenberg and Doucet [91]).
A language is context-free if and only if it is the language of an E0L system $< \Sigma, P, \omega, \Delta >$ such that, for each a in Δ, the production $a \rightarrow a$ is in P.

Theorem 3.3. (Herman and Walker [47]).
A language is context-free if and only if it is the adult language of a 0L system.

As far as fitting some classes of L languages into the known formal language theoretic framework is concerned, results more detailed than those of Theorem 3.1 are available. For example we have the following results. Let $\mathcal{L}$(IND) denote the class of indexed languages (see A. Aho "Indexed grammars - An extension of context-free grammars" J. of the ACM. 15 (1968), 647-671) and let $\mathcal{L}$(PROG) denote the class of Λ-free programmed languages (see D. Rosenkrantz "Programmed grammars and classes of formal languages" J. of the ACM. 16 (1969), 107-131).

Theorem 3.4. (Culik [7] and Rozenberg [89]).
$\mathcal{L}(ETOL) \subsetneq \mathcal{L}(IND)$ and $\mathcal{L}(ETOL) \subsetneq \mathcal{L}(PROG)$.

Results like these can be helpful for getting either new properties or nice proofs of known properties of some classes of L languages. For example, the family $\mathcal{L}(IND)$ possesses quite strong decidability properties which are then directly applicable to the class of ETOL languages. An example will be considered in section 8.

4. OTHER CHARACTERIZATIONS OF CLASSES OF L LANGUAGES WITHIN THE FRAMEWORK OF FORMAL LANGUAGE THEORY

A classical step toward achieving a mathematical characterization of a class of languages is to investigate its closure properties with respect to a number of operations. There is even a trend in formal language theory, called the AFL theory (see S. Ginsburg, S. Greibach and J. Hopcroft "Studies in Abstract Families of Languages", Memoirs of the AMS, 87, (1969)) which takes this as a basic step towards characterizing classes of languages. The next two results display the behaviour of some of the families of L languages with respect to the basic operations considered in AFL theory. There are essentially two reasons for considering these operations. One reason is that in this way we may better contrast various families of L languages with traditional families of languages. The other reason is that we still know very little about what set of operations would be natural for families of L languages. (In what follows the symbols $\cup, ., *, hom, hom^{-1}, \cap_R$ denote the operations of union, product, Kleene's closure, homomorphism, inverse homomorphism and intersection with a regular language respectively.)

Theorem 4.1. (Rozenberg and Doucet [91], Rozenberg [81], Rozenberg [86], Rozenberg and Lee "TIL systems and languages")
None of the families of 0L, T0L, IL, TIL languages is closed with respect to any of the following operations: $\cup, ., *, hom, hom^{-1}, \cap_R$.

Theorem 4.2. (Rozenberg [89], van Dalen [2], Herman [35])
The families of ETOL and EIL languages are closed with respect to all of the operations $\cup, ., *, hom, hom^{-1}, \cap_R$. The family of E0L languages is closed with respect to the operations $\cup, ., *, hom$ and $\cap_R$ but it is not closed with respect to the hom^{-1} operation.

When we contrast the above two results with each other we see the role of nonterminals in defining languages of L systems. On the other hand contrasting the second result with the corresponding results for the classes of context-free and context-sensitive languages enables us to learn more about the nature of parallel rewriting in language generating systems.

In formal language theory, when a class of generative devices for defining languages is given, one often looks for a class of acceptors (recognition devices) which would yield the same family of languages. Such a step usually provides us with a better insight into the structure of the given family of languages and (sometimes) it provides us with additional tools for proving theorems about the given family of languages.

Several machine models for L systems are already available, see Culik and Opatrny [9], van Leeuwen [55], Rozenberg [90], Savitch [108]. (Of these, the most general models are those presented by Savitch).

As an example, we discuss now the notion of a pre-set pushdown automaton introduced in van Leeuwen [55]. Roughly speaking a pre-set pushdown automaton is like an ordinary pushdown automaton, except that at the very beginning of a computation a certain location on the pushdown store of the automaton is assigned as the maximum location to which the store may grow during the computation. Such a distinguished location is used in such a way that when the automaton has reached it then it switches to a different transition function. When a pre-set pushdown automaton is constructed in such a way that there is a fixed bound on the length of a local computation (meaning a computation that the pointer does not move) then we call it a <u>locally finite pre-set pushdown</u> automaton. We say that a pre-set pushdown automaton has a <u>finite return property</u> if there is a fixed bound on the number of recursions that can occur from a location.

<u>Theorem 4.3.</u> (van Leeuwen [55,56], Christensen [6])
The family of languages accepted by pre-set pushdown automata contains properly the family of E0L languages and is properly contained in the family of ET0L languages.

<u>Theorem 4.4.</u> (van Leeuwen [55])
The family of languages accepted by locally finite pre-set pushdown automata with the finite return property equals the family of E0L languages.

5. SQUEEZING SEQUENCES OUT OF L SYSTEMS.

From a biological point of view the time-order of development is at least as interesting as the unordered set of morphological patterns which may develop. This leads to investigation of sequences of words rather than unordered sets of words (languages), which is a novel point in formal language theory. It turned out that investigation of sequences (of words) gives rise to a non-trivial and interesting mathematical theory (see, e.g., Herman and Rozenberg [45], Paz [74], Paz and Salomaa [75], Rozenberg [82], Szilard [111], Vitanyi [116]).

The most natural way to talk about word sequences in the context of L systems is to consider such L systems which (starting with the axiom) yield the unique next word for a given one. We define now one such class of such L systems.

<u>Definition 5.1.</u> An IL system $G = \langle \Sigma, P, g, \omega \rangle$ is called <u>deterministic</u> (abbreviated DIL system) if whenever $\langle \alpha, a, \beta, \gamma_1 \rangle$ and $\langle \alpha, a, \beta, \gamma_2 \rangle$ are in P then $\gamma_1 = \gamma_2$.

Note that a 0L system is a particular instance of an IL system. Hence we shall talk about D0L systems. The most natural way to define sequences by DIL systems is to take the exhaustive approach, which simply means to include in the sequence of a DIL system the set of all words that the system generates (and in the order that these words are generated).

<u>Definition 5.2.</u> Let $G = \langle \Sigma, P, g, \omega \rangle$ be a DIL system. The <u>sequence of G</u>, denoted as E(G), is defined by $E(G) = \omega_0, \omega_1, \ldots$ where $\omega_0 = \omega$ and for $i \geq 1$, $\omega_{i-1} \underset{G}{\Longrightarrow} \omega_i$.

<u>Example 5.1.</u> Let $G = \langle \Sigma, P, g, \omega \rangle$ be a DIL system such that $\Sigma = \{a,b\}$, $\omega = baba^2$ and $P = \{\langle g,b,\Lambda \rangle \to ba, \langle a,b,\Lambda \rangle \to ba^2, \langle a,a,\Lambda \rangle \to a, \langle b,a,\Lambda \rangle \to a\}$. Then $E(G) = baba^2, ba^2ba^4, \ldots, ba^kba^{2k}, \ldots$.

<u>Definition 5.3.</u> Let s be a sequence of words. It is called a <u>DIL sequence</u> (<u>D0L sequence</u>) if there exists a DIL system (D0L system) G such that $s = E(G)$.

Obviously as in the case of L languages (see section 2) one can apply various mechanisms of squeezing sequences out of DIL systems. Thus, in the obvious sense, we can talk about EDIL and ED0L sequences (when using nonterminals for defining sequences) or about CDIL and CD0L sequences (when using codings for defining sequences). Comparing the sequence generative power of these different mechanisms for sequence definition, we have, for example, the following result.

Theorem 5.1. (Nielsen, Rozenberg, Salomaa, Skyum [71,72])
The family of D0L sequences is strictly included in the family of ED0L sequences, which in turn is strictly included in the family of CD0L sequences.

In the sequel we shall use the term L sequence to refer to any kind of a sequence discussed in this section.

6. GROWTH FUNCTIONS; AN EXAMPLE OF RESEARCH ON (CLASSES OF) L SEQUENCES

As an example of an investigation of properties of L sequences and their classes we will discuss the so called growth functions. It happens quite often (in both mathematical and biological considerations) that one is interested only in the lengths of the words generated by an L system. When the system G under a consideration is deterministic then, in this way, one obtains a function assigning to each positive integer n the length of the n'th word in the sequence of G. This function is called the growth function of G. The theory of growth functions of deterministic L systems is one of the very vigorously (and succesfully) investigated areas of L system theory (see, e.g., Doucet [15], Paz and Salomaa [75], Salomaa [98], Vitanyi [116]). It also lends itself to the application of quite powerful mathematical tools (such as difference equations and formal power series).

Definition 6.1. Let G be a DIL system with $E(G) = \omega_0, \omega_1, \ldots$. The growth function of G, denoted as f_G, is a function from nonnegative integers into nonnegative integers such that $f_G(n) = |\omega_n|$.

Example 6.1. Let $G = \langle \{a,b\}, \{a \to b, b \to ab\}, a \rangle$ be a D0L system. Then $f_G(n)$ is the n'th element of the Fibonacci sequence 1,1,2,3,5,... .

Example 6.2. Let $G = \langle \{a,b,c,d\}, \{a \to abd^6, b \to bcd^{11}, c \to cd^6, d \to d\}, a \rangle$ be a D0L system. Then $f_G(n) = (n+1)^3$.

Directly from the definition of an L system we have the following result.

Theorem 6.1.
The growth function of a DIL system G such that L(G) is infinite is at most exponential and at least logarithmic.

The following are typical examples of problems concerning growth

functions.
Analysis problem: Given a DIL system, determine its growth function.
Synthesis problem: Given a function f from nonnegative integers into nonnegative integers, determine if possible a system G belonging to a given class of systems (say, D0L systems) such that $f = f_G$.
Growth equivalence problem: Given two DIL systems, determine whether their growth functions are the same.

In the following there are some typical results about growth functions.

Theorem 5.2. (Paz and Salomaa [75])
If G is a D0L system then f_G is exponential, polynomial or a combination of these.

Theorem 5.3. (Paz and Salomaa [75])
If f is a function from the nonnegative integers into the nonnegative integers such that
(i) For every n there exists an m such that
$f(m) = f(m+1) = \dots = f(m+n)$, and
(ii) $\lim_{t\to\infty} f(t) = \infty$,
then f is not the growth function of a D0L system.

7. STRUCTURAL CONSTRAINTS ON L SYSTEMS

One of the possible ways of investigating the structure of any language (or sequence) generating device is to put particular restrictions directly on the definition of its various components and then to investigate the effect of these restrictions on the language generating power. Theorem 2.5.1 represents a result in this direction (it says for example that removing nonterminals from ETIL, EIL, ET0L or E0L systems decreases the language generating power of these classes of systems). Now we indicate some other results among the same line.

The first of these results investigates the role of erasing productions in generating languages (sequences) by the class of E0L (ED0L) systems. (A production $<\alpha,a,\beta> \to \gamma$ is called an erasing production if $\gamma = \Lambda$).

Theorem 7.1. (Herman [39])
A language K is an E0L language if and only if there exists an E0L system G which does not contain erasing productions such that $K-\{\Lambda\} = L(G)$.

Theorem 7.2.
There exists an EDOL sequence which does not contain Λ, and which cannot be generated by an EDOL system without erasing productions.

Our next result discusses the need of "two-sided context" (more intuitively: "two-sided communication") in IL systems.
Theorem 7.3. (Rozenberg [86])
There exists a language K such that K is a $< 1,1 >$ L language and for no $m \geqslant 0$ is K an $< m,0 >$ L language or a $< 0,m >$ L language.

Our last sample result in this line says that for the class of IL systems with two-sided context it is the amount of context available and not its distribution that matters as far as the language generating power is concerned.
Theorem 7.4. (Rozenberg [86])
A language is an $< m,n >$ L language for some $m,n \geqslant 1$ if and only if it is a $< 1,m+n-1 >$ L language. For each $m \geqslant 1$ there exists a $< 1,m+1 >$ L Language which is not a $< 1,m >$ L language.

8. DECISION PROBLEMS

Considering decision problems for language generating devices is a customary research topic in formal language theory. It helps to understand the "effectiveness" of various classes of language generating devices, explores the possibilities of changing one way of describing a language into another one, and, in connection with this, it may be a guide line for a choice of one rather than another class of specifications of languages. (For example it is quite often the case that when a membership problem for a given class of language defining devices turns out to be undicidable, one looks for a subclass for which this problem would be decidable). Various decision problems are also considered in the theory of L systems. In addition to more or less traditional problems considered usually in formal language theory new problems concerning sequences are also considered.

Some results concerning decision problems are obtained as direct corollaries of theorems fitting different classes of L languages into known hierarchies of languages. For example, as an application of Theorem 3.4 we have the following result.
Theorem 8.1.
Membership, emptiness and finiteness problems are decidable in the class of ETOL systems.

The following result by Blattner (for different solutions see also Salomaa [99] and Rozenberg [84]) solved a problem which was open for some time.

Theorem 8.2. (Blattner [6])
The language equivalence problem is not decidable in the class of 0L systems.

The corresponding problem for the class of D0L systems is one of the most intriguing and the longest open problems in the theory of L systems. Some results are however available about subclasses of the class of D0L systems.

Theorem 8.3. (Ehrenfeucht and Rozenberg)
If G_1, G_2 are D0L systems such that f_{G_1} and f_{G_2} are bounded by a polynomial (which is decidable) then it is decidable whether they generate the same language (sequence).

The following result points out "undecidability" of various extensions of the class of 0L systems. (In what follows an F0L system denotes a system which is like a 0L system, except that it has a finite number of axioms rather than a single one).

Theorem 8.4. (Rozenberg)
It is undecidable whether an arbitrary IL (C0L, E0L, F0L) system generates a 0L language.

For L sequences we have for example the following results.

Theorem 8.5. (Paz and Salomaa [75], Vitanyi [115])
The growth equivalence problem is decidable in the class of D0L systems but it is not decidable in the class of DIL systems.

Theorem 8.6. (Salomaa [98], Vitanyi [115,116])
Given an arbitrary D0L system G it is decidable whether f_G can be bounded by a polynomial. This problem is not decidable if G is an arbitrary DIL system.

Theorem 8.7. (Nielsen [70])
The language equivalence problem for D0L systems is decidable if and only if the sequence equivalence problem for D0L systems is decidable.

Theorem 8.8. (Ehrenfeucht, Lee and Rozenberg)
If G_1, G_2 are two arbitrary D0L systems and x is a word then it is decidable whether x occurs as a subword the same number of times in the

corresponding words of sequences generated by G_1 and G_2.

9. GLOBAL VERSUS LOCAL BEHAVIOUR OF L SYSTEMS

The topic discussed in this chapter, global versus local behaviour of L systems, is undoubtedly one of the most important in the theory of L systems. Roughly speaking, a global property of an L system is a property which can be expressed independently of the system itself (for example a property expressed in terms of its language or sequence). On the other hand a local property of an L system is a property of its set of productions (for example a property of the "graph" of productions of a given system). In a sense the whole theory of L systems emerged from an effort to explain on the local (cellular) level global properties of development.

As an example of research in this direction we discuss the so called locally catenative L systems and sequences (see Rozenberg and Lindenmayer [95]). Locally catenative L sequences are examples of L sequences in which the words themselves carry in some sense the history of their development.

Definition 9.1. An infinite sequence of words $\tau_0, \tau_1, \dots$ is called locally catentative if there exist positive integers $m, n, i_1, \dots, i_n$ with $n \geqslant 2$ such that for each $j \geqslant m$ we have $\tau_j = \tau_{j-i_1}\tau_{j-i_2}\dots\tau_{j-i_n}$.

Definition 9.2. A DIL (or a DOL) system G is called locally catenative if E(G) is locally catenative.

Very little is known about locally catenative DIL sequences. For locally catenative DOL sequences some interesting results are available. Our first result presents a property of a DOL sequence which is equivalent to the locally catenative property.

Let G be a DOL system such that $E(G) = \omega_0, \omega_1, \dots$ is a doubly infinite sequence, meaning that the set of different words occurring in E(G) is infinite. We say that E(G) is covered by one of its words if there exist $k \geqslant 0$ and $j \geqslant k+2$ and a sequence s of occurrences of ω_k in (some of the) strings $\omega_{k+1}, \omega_{k+2}, \dots, \omega_{j-1}$ such that ω_j is the catenation of the sequence of its subwords derived from respective elements of s.

Theorem 9.1. (Rozenberg and Lindenmayer [95])
A DOL system G is locally catenative if and only if E(G) is covered by one of its words.

Our next theorem presents the result of an attempt to find a "structural" property of the set of productions of a DOL system such that its sequence is locally catenative. First we need some more notation and terminology.

If $G = < \Sigma, P, \omega >$ is a DOL system then the graph of G is the directed graph whose nodes are elements of Σ and for which a directed edge leads from the node a to the node b if and only if $a \to \alpha b \beta$ is in P for some words α, β over Σ.

Theorem 9.2. (Rozenberg and Lindenmayer [95])
Let $G = < \Sigma, P, \omega >$ be a DOL system without erasing productions such that both E(G) and L(G) are infinite, ω is in Σ and each letter from Σ occurs in a word in E(G). If there exists σ in Σ such that $\omega \overset{*}{\underset{G}{\Rightarrow}} \sigma$ and each cycle in the graph of G goes through the node σ then E(G) is locally catenative.

We may note that neither of the above results is true in the case of DIL sequences (systems).

10. DETERMINISTIC VERSUS NONDETERMINISTIC BEHAVIOUR OF L SYSTEMS

An L system is called deterministic if, roughly speaking, after one of its tables has been chosen, each word can be rewritten in exactly one way. Investigation of the role the deterministic restriction plays in L systems is an important and quite extensively studied topic in the theory of L systems (see, e.g., Doucet [14], Ehrenfeucht and Rozenberg [17], Lee and Rozenberg [52], Nielsen [70], Paz and Salomaa [75], Rozenberg [82], Salomaa [98], Szilard [111]). First of all, some biologists claim that only deterministic behaviour should be studied. Secondly, studying deterministic L systems, especially when opposed to general (undeterministic) L systems, allows us to better understand the structure of L systems. Finally, the notion of determinism studied in this theory differs from the usual one studied in formal language theory. One may say that they are dual to each other: "deterministic" in L systems means a deterministic process of generating strings, "deterministic" in the sense used in formal language theory means a deterministic process of parsing. Contrasting these notions may help us to understand some of the basic phenomena of formal language theory.

Definition 10.1. A TIL system $G = < \Sigma, \mathcal{P}, g, \omega >$ is called deterministic (abbreviated DTIL system) if for each table P of $\mathcal{P}$ if $< \alpha, a, \beta > \to \gamma_1$ and $< \alpha, a, \beta > \to \gamma_2$ are in P then $\gamma_1 = \gamma_2$.

A TOL system is a special instance of a TIL system hence we talk about DTOL systems.

As an example of a research towards understanding deterministic restriction in L systems we shall discuss deterministic TOL systems.

It is not difficult to construct examples of languages which can be generated by a TOL system but cannot be generated by a DTOL system. One would like however to find a nontrivial (and hopefully interesting) property which would be inherent to the class of deterministic TOL languages. It turns out that observing the sets of all subwords generated by DTOL systems provides us with such a property. In fact the ability to generate an arbitrary number of subwords of an arbitrary length is a property of a TOL system which disappears when the deterministic restriction is introduced. More precisely, we have the following result. (In what follows $\pi_k(L)$ denotes the number of subwords of length k that occur in the words of L).

Theorem 10.1. (Ehrenfeucht and Rozenberg [17])
Let Σ be a finite alphabet such that $\#\Sigma = n \geqslant 2$. If L is a language generated by a DTOL system, $L \subseteq \Sigma^*$, then $\lim_{k\to\infty} \frac{\pi_k(L)}{n^k} = 0$.

Various ramifications of this result are discussed in Ehrenfeucht, Lee and Rozenberg [18].

11. L TRANSFORMATIONS

An L system consists of an L scheme and of a fixed word (the axiom). An L scheme by itself represents a transformation (a mapping) from Σ^+ into Σ^* (where Σ is the alphabet of the L scheme). From the mathematical point of view, it is the most natural to consider such transformations. This obviously may help to understand the nature of L systems. Although not much is known in this direction yet, some results about TOL transformations are already available (see Ginsburg and Rozenberg [31]).

Let a TOL scheme $G = \langle \Sigma, \mathcal{P} \rangle$ be given. (Note that each table P of $\mathcal{P}$ is in fact a finite substitution, or a homomorphism in the case that $\mathcal{P}$ satisfies a deterministic restriction). The basic situation under examination consists of being given two of the following three sets: a set L_1 of (start) words over Σ, a set L_2 of (target) words over Σ, and a (control) set $\mathcal{C}$ of finite sequences of applications of tables from $\mathcal{P}$. The problem is to ascertain information about the remaining set. (Note that we can consider a sequence of elements from $\mathcal{P}$ either as a word over $\mathcal{P}^*$, called a control word, or a mapping from Σ^* into Σ^*. We shall do both in the sequel but this should not lead to confusion). The following are examples of known results concerning this problem.

Theorem 11.1. (Ginsburg and Rozenberg [31])
If L_2 is a regular language, and L_1 an arbitrary language then the set $\mathcal{C}$ of control words leading from L_1 to L_2 is regular.

Theorem 11.2. (Ginsburg and Rozenberg [31])
If L_2 is a regular language and $\mathcal{C}$ is an arbitrary set of control words then the set of all words mapped into L_2 by $\mathcal{C}$ is regular.

Theorem 11.3. (Ginsburg and Rozenberg [31])
If L_1 is a regular language and $\mathcal{C}$ is a regular set of control words then the set of all words obtained from the words of L_1 by applying mappings from $\mathcal{C}$ is an ETOL language. Moreover each ETOL language can be obtained in this fashion.

Also the following is quite an interesting result.
Theorem 11.4. (Ginsburg and Rozenberg [31])
There is no TOL scheme $S = \langle \Sigma, \mathcal{P} \rangle$ such that $\mathcal{P}^*$ is the set of all finite nonempty substitutions on Σ^*. There is no TOL scheme $S = \langle \Sigma, \mathcal{P} \rangle$ such that $\mathcal{P}^*$ is the set of all homomorphisms on Σ^*.

We may also mention here the following result concerning "adult L transformations". Roughly speaking the adult language of an IL scheme G with an alphabet Σ is the set of all those strings over Σ^* which are transformed by G into themselves and only themselves.
Theorem 11.5.
There are regular languages which are not adult languages of IL schemes.

12. GETTING DOWN TO PROPERTIES OF SINGLE L LANGUAGES OR SINGLE L SEQUENCES

Undoubtedly, one of the aims of the theory of L systems is to understand the structure of a single L language or a single L sequence. Although some results in this direction are already available (see, e.g., Ehrenfeucht and Rozenberg [19,22,24] and Rozenberg [82]), in my personal opinion, there is not enough work done on this (rather difficult) topic.

Here are two samples of already available results.

Let Σ be a finite alphabet and B a non-empty subset of Σ. If x is a word over Σ then $\#_B(x)$ denotes the number of occurrences of elements from B in x. Let K be a language over Σ and let $I_{K,B} = \{n : \#_B(w) = n$ for some w in K$\}$. We say that B is numerically dispersed in K if $I_{K,B}$

is infinite and for every positive integer k there exists a positive integer n_k such that for every u_1, u_2 in $I_{K,B}$ such that $u_1 > u_2 > n_k$ we have $u_1 - u_2 > k$. We say that <u>B is clustered in K</u> if $I_{K,B}$ is infinite and there exist positive integers k_1, k_2, both larger than 1, such that for every w in L if $\#_B(w) \geq k_1$ then w contains at least two occurrences of symbols from B which are distant less than k_2.

<u>Theorem 12.1.</u> (Ehrenfeucht and Rozenberg [24])
Let K be an EOL language over an alphabet Σ and let B be a nonempty subset of Σ. If B is numerically dispersed in K, then B is clustered in K.

Results like the above one are very useful for proving that some languages are not in a particular class. This is often a difficult task. For example as a direct corollary of Theorem 12.1 we have that the language $\{w \text{ in } \{0,1\}^* : \#_{\{0\}}(w) \text{ is a power of } 2\}$ is not an EOL language. (The direct combinatorial proof of this fact in Herman [35] is very tedious.)

For DOL sequences we have the following result. (In what follows if x is a word and k a positive integer then $\mathrm{Pref}_k(x)$ denotes either x itself if $k \geq |x|$ or the word consisting of the first k letters of x if $k < |x|$. Similarly $\mathrm{Suf}_k(x)$ denotes either x itself if $k \geq |x|$ or the word consisting of the last k letters of x if $k < |x|$).

<u>Theorem 12.2.</u> (Rozenberg [82])
For every DOL system G such that $E(G) = \omega_0, \omega_1, \ldots$ is infinite there exists a constant C_G such that for every integer k the sequence $\mathrm{Pref}_k(\omega_0), \mathrm{Pref}_k(\omega_1), \ldots$ (respectively $\mathrm{Suf}_k(\omega_0), \mathrm{Suf}_k(\omega_1), \ldots$) is ultimately periodic with period C_G.

The above result is not true for DIL sequences; the corresponding sequences of prefixes (or suffixes) are not necessarily ultimately periodic.

It should be clear that Theorem 12.2 can provide elegant proofs that some sequences are not DOL sequences.

13. GENERALIZING L SYSTEMS IDEAS; TOWARDS A UNIFORM FRAMEWORK

As in every mathematical theory, also in the (mathematical) theory of L systems one hopes to generalize various particular results and concepts and get a unifying framework for the theory. One still has to wait for such a uniform framework for the theory of L systems, however partial results are already available.

It was already noticed in early papers on L systems (see, e.g., Rozenberg [81]) that the underlying operation is that of the iterated substitution. This operation was quite intensively studied in formal language theory, however in the theory of L systems it occurs in somewhat modified way (one has a finite number of finite substitutions, tables, and then performs all their possible "iterative" compositions). This point of view was taken by J. van Leeuwen and A. Salomaa and (as a rather straightforward generalization of the notion of an ETOL system) they introduced the so called K-iteration grammars (van Leeuwen [57], Salomaa [103]).

For a language family K, a K-substitution is a mapping σ from some alphabet V into K. The mapping is extended to languages in the usual way. A <u>K-iteration grammar</u> is a construct $G = \langle V_N, V_T, S, U \rangle$ where V_N, V_T are disjoint alphabet (of nonterminals and terminals), $S \in (V_N \cup V_T)^*$ (the axiom) and $U = \{\sigma_1, \ldots, \sigma_n\}$ is a finite set of K-substitutions defined on $(V_N \cup V_T)$ with the property that, for each i and each a in $(V_N \cup V_T)$, $\sigma_i(a)$ is a language over $(V_N \cup V_T)$. The language generated by such a grammar is defined by

$$L(G) = \bigcup \sigma_{i_k} \ldots \sigma_{i_1}(S) \cap V_T^*,$$

where the union is taken over all integers $k \geqslant 1$ and over all k-tuples $(i_1, \ldots, i_k)$ with $1 \leqslant i_j \leqslant n$. The family of languages generated by K-iteration grammars is denoted by $\underline{K_{iter}}$. For $t \geqslant 1$, we denote by $K_{iter}^{(t)}$ the subfamily of K_{iter}, consisting of languages generated by such grammars where U consists of at most t elements.

<u>Example 13.1.</u> If we denote the family of all finite languages by F, then it is clear that $F_{iter}^{(1)} = \mathcal{L}(EOL)$ and $F_{iter} = \mathcal{L}(ETOL)$.

The families of K-iterated languages can be related to Abstract Families of Languages (AFL's) as follows.

<u>Theorem 13.2.</u> (van Leeuwen [57], Salomaa [103])
If the family K contains all regular languages and is closed under finite substitution and intersection with regular languages then both K_{iter} and $K_{iter}^{(t)}$ are full AFL's.

The notion of a K-iteration grammar was extended to the case of context-sensitive substitutions by D. Wood in "A note on Lindenmayer systems, spectra and equivalence" McMaster University, Comp. Sc. Techn. Dep. No. 74/1. Some results are also available about possibly extending a few basic properties of the families of L systems (or L languages) to the case of families of K-iteration grammars (or languages) (see, e.g., Salomaa [103] and the above mentioned paper by Wood).

14. CONCLUSIONS

We would like to conclude this paper with two remarks.

(1) In the first five years of its existence the mathematical theory of L systems has become each year fruitful and popular. This is exemplified by exponential growth of the number of papers produced (per year), and a linear (with a decent coefficient) growth of both the number of results and the number of people joining the area.

(2) It may have already occurred to the reader (and it is certainly clear to the author of this paper) that both formal language theory and the theory of L systems have benefited by the existence of the other.

ACKNOWLEDGEMENT

The author is indebted to A. Lindenmayer and A. Salomaa for their useful comments on the first version of this paper.