stichting
mathematisch
centrum

MC

AFDELING INFORMATICA IW 28/74 DECEMBER

P.M.B. VITANYI

DETERMINISTIC LINDENMAYER LANGUAGES, NONTERMINALS AND HOMOMORPHISMS

Prepublication

2e boerhaavestraat 49 amsterdam

Printed at the Mathematical Centre, 49, 2e Boerhaavestraat, Amsterdam.

The Mathematical Centre, founded the 11-th of February 1946, is a non-profit institution aiming at the promotion of pure mathematics and its applications. It is sponsored by the Netherlands Government through the Netherlands Organization for the Advancement of Pure Research (Z.W.O), by the Municipality of Amsterdam, by the University of Amsterdam, by the Free University at Amsterdam, and by industries.

AMS (MOS) subject classification scheme (1970): 68A30 68A25 94A30 92A05

ACM -Computer Review- category: 5.22 5.23

DETERMINISTIC LINDENMAYER LANGUAGES, NONTERMINALS AND HOMOMORPHISMS

by

Paul M.B. Vitányi.

ABSTRACT

Lindenmayer systems are a class of parallel rewriting systems originally introduced to model the growth and development of filamentous organisms. Families of languages generated by deterministic Lindenmayer systems (each string has a unique successor) are investigated. In particular, the use of nonterminals, homomorphisms and both together are studied for deterministic Lindenmayer systems using one sided context (D1Ls) and two sided context (D2Ls). Languages obtained from Lindenmayer systems by the use of nonterminals are called extensions. Typical results are: the closure under letter to letter homomorphism of the family of extensions of D1L languages is equal to the family of recursively enumerable languages, although the family of extensions of D1L languages does not even contain all regular languages. Let P denote the restriction that the system does not rewrite a letter as the empty word. The family of extensions of PD2L languages is equal to the family of languages accepted by deterministic linear bounded automata. The closure under nonerasing homomorphism of the family of extensions of PD1L languages does not even contain languages like $\{a_1,a_2,\ldots,a_n\}^*\backslash\{\lambda\}$, $n \geq 2$. The closure of the family of PD1L languages under homomorphisms, which map a letter either to itself or to the empty word, is equal to the family of recursively enumerable languages. Strict inclusion results follow from necessary conditions for a language to be in one of the considered families. By stating the results in their strongest form, the paper contains a systematic classification of the effect of nonterminals, letter to letter homomorphisms, nonerasing homomorphisms and homomorphisms for all the basic types of deterministic Lindenmayer systems

using context. The relevance of the used concepts in the biological setting is discussed.

KEYWORDS & PHRASES: formal languages, Lindenmayer systems, monogenic rewriting, nonterminals, homomorphisms, Chomsky hierarchy.

1. INTRODUCTION

The study of Lindenmayer languages (also called L languages or developmental languages) has been one of the major trends in automata and formal language theory during the past few years. L languages are generated by highly parallel rewriting systems introduced by Lindenmayer [12] to model the growth and development of filamentous biological organisms. These Lindenmayer systems, or L systems for short, have been investigated in a large number of papers both from the language theory and theoretical biology point of view. (See e.g. [9] and the references contained therein.) A Lindenmayer system is called deterministic if each string has exactly one successor under the rewriting rules. The purpose of this paper is to make a systematic study of languages generated by deterministic L systems and the effect of two essentially different defining mechanisms: the use of nonterminals and the use of homomorphic mappings of different kinds. Both mechanisms are frequently used in formal language theory [20], [19].

An L system consists of an initial string of letters, symbolizing an initial linear array of cells (a filament); and the subsequent strings (stages of development) are obtained by rewriting all letters of a string simultaneously at each time step. When the rewriting of a letter may depend on the m letters to its left and the n letters to its right we talk

about an (m, n) L system. If $m = n = 0$ the L system is said to be context independent or without interactions, if $m + n > 0$ the L system is said to be context dependent or with interactions. Most of the literature on L systems is concerned with 0L systems ($m = n = 0$); 1L systems ($m + n = 1$); and 2L systems ($m = n = 1$).

From the point of view of developmental biology the language consisting of the set of all strings generated by the system is of primary interest. Such an L language is taken to correspond with the set of all developmental stages the organism might attain in its development. Here also homomorphic mappings (especially those in which a letter is mapped to a letter) are of considerable importance. The reasons for this are as follows. When we make observations of a particular organism, and wish to describe it by strings of symbols, we first associate a symbol to each particular cell. We divide the cells into a number of types and associate the same symbol to each cell of the same type. It is possible that the development of the organism can be described by an L system, but the actual system describing it uses a finer subdivision into types than we could observe. This is often experimentally unavoidable. In this case, the set of strings generated by a given L system is a coding of the "real" language of the organism which the given L system is supposed to describe. More formal language theory oriented investigators,

however, divide the set of letters the L system uses into a set of terminals and nonterminals. The language obtained from the L system by this mechanism consists of all strings over terminals generated by the system. Such languages are called extensions of L languages. (They are obtained by taking the intersection of the "ordinary" L language and the set of all strings over the terminals, which operation extends considerably the generating power of the type of L system under consideration.) Families of extensions of L languages usually have nice mathematical properties like closure under certain operations etc. The distinction between terminals and nonterminals is well motivated from the linguistic point of view because nonterminals correspond to the syntactic classes of the language. This distinction is not so well motivated with respect to theories of development where we are interested in the set of all generated strings. One of the facts which have made the use of nonterminals interesting within the theory of developmental languages is that it was established in [4] and [5] that, for basic families of 0L systems the use of nonterminals and the use of letter to letter homomorphisms is equivalent as far as the generating capacity is concerned. Thus, the trade-off between the two language defining mechanisms (i.e. nonterminals versus homomorphisms) has become a very interesting and well motivated problem for L systems. Continueing this train of thought, trade offs between combinations of one or two sided context, restrictions where no letter is rewritten as the empty word, use of nonterminals and various kinds of

homomorphisms are interesting. The present paper is concerned with this topic where we restrict our attention to the deterministic L systems.

A biological motivation for the use of a nonterminal mechanism may be given as follows. Suppose that under certain conditions an organism consisting solely of cells in certain "terminal" states stabilizes i.e. it reaches an "adult" stage. Such a condition might be the presence or absence of chemicals, enzymes etc. either induced by external agents or by internal agents. The presence of some cells in "nonterminal" states then could, by changing the internal condition, prevent the organism to stabilize. The extension of the L language then corresponds to the set of all adult stages the modeled organism might reach in its development. Alternatively, we could select from all strings generated by an L system those strings that can only be rewritten as themselves. These languages have been called stable string languages (or, with a biological connotation,adult languages) and were introduced by Walker (c.f. [21] and the references contained therein). Vitányi and Walker [21] established that for many classes of L systems, especially those using interactions, the families of extensions and stable string languages are equal. Hence the use of nonterminals for developmental systems is of interest in its own right and because of various trade offs which are possible.

The notion of generating languages by monogenic (deterministic) rewriting systems, i.e. each string has a unique successor, is more or less foreign to the usual generative grammar approach since there such a language would either be empty or consist of one string only. Accepting languages by deterministic automata on the other hand, is a well investigated subject and has important applications in parsing problems for formal languages, [20]. (Considering all strings generated by a grammar rather than only strings over terminals is of central interest for both theories of parsing and development.)

Nondeterminism as it appears in formal language theory has no counterpart in nature on the macroscopic level in which we are dealing with the modeling of biological development. The closest we can approach it in the physical reality of development (morphogenesis) is by a probabilism where the probabilities are contingent with influences of unknown factors internal and external to the organism. Therefore, deterministic L systems are particularly relevant in the biological setting as would also appear from the fact that most attempts to provide L systems modeling the development of actual biological organisms use deterministic systems [1], [6], [7], [8], [9], [10]. The study of the change in pattern, size and weight of a growing organism as a function of time constitutes a considerable portion of the literature on developmental biology. Usually, genetically identical specimens of a specific

organism are investigated in a controlled environment and their changes in time are described. The scientific presupposition is that identical genetical material and identical environment will result in an approximately identical developmental history, i.e. that the experiment is repeateable. This assumes a deterministic (causal) underlying structure, and makes a good case for the biological importance of the study of deterministic L systems.

The paper falls apart in roughly three main themes. In section 2 we formally define L systems and relate them to Turing machines, as in [3]. Sections 3 and 4 are concerned with "ordinary" deterministic L languages, i.e. languages consisting of all strings generated by the systems; in sections 5 and 6 we deal with extensions of deterministic L languages, i.e. languages consisting of all strings over some terminals generated by the systems.

In section 3 we are interested in Lindenmayer languages which are not recursive. The existence of such languages is a known fact [3]. We provide a more detailed construction for the deterministic case and develop a simulation technique which will prove useful in the sequel of the paper. In section 4 we compare families of deterministic L languages with the Chomsky hierarchy. Here our results refine those in [3], [17] and [18]. In section 5 we compare families of extensions of deterministic L languages with the Chomsky hierarchy. Typical results are : the amount of context needed for rewriting

makes no difference for families of extensions; the only differences lie in no context, context on one side and context on both sides. Let the capital D denote the deterministic property. The family of extensions of D2L languages is equal to the family of recursively enumerable languages as is also the closure under letter to letter homomorphism of the family of extensions of D1L languages. On the other hand, the family of extensions of D1L languages does not even contain all regular languages.

In section 6 we consider extensions and homomorphisms of languages generated by deterministic L systems with the propagating property: no letter can be rewritten as the empty word. As is well known such a restriction usually limits drastically the generating capacity of a rewriting system. We show that the family of extensions of PD2L languages (the capital P stands for propagating) is equal to the family of languages accepted by deterministic linear bounded automata. The closure under nonerasing homomorphism of the family of extensions of PD1L languages is strictly included in the family of extensions of PD2L languages. Indeed, this closure does not even contain languages like $\{a_1, a_2, \ldots, a_n\}^* \setminus \{\lambda\}$, $n \geq 2$. (Contrast this with the result for the nonpropagating case in section 5.) On the other hand, the closure of the family of PD1L languages under homomorphisms which map a letter either to itself or to the empty word is again equal to the family of recursively enumerable languages.

Our strict inclusion results follow from necessary properties of the considered language families rather than by an exhaustive analysis of a particular example.

Essentially, the paper analyzes the trade offs which are possible between combinations of one or two sided context, the property that no letter is rewritten as the empty word, use of nonterminals and various kinds of homomorphisms. By stating results in their strongest form, the paper contains a systematic classification about the effect of these mechanisms on the generating capacity of deterministic L systems using context.

For a treatment of the effect of nonterminals, homomorphisms and letter to letter homomorphisms in different variations of 0L systems the reader is referred to [14].

2. LINDENMAYER SYSTEMS AND TURING MACHINES

We assume that the reader is familiar with the usual terminology of formal language theory as e.g. presented in [11] or [19]. Except when indicated otherwise we shall customarily use, with or without indices, i, j, k, ℓ, m, n, p, q, r, s, t to range over the set of natural numbers $N = \{0, 1, 2, \ldots\}$; a, b, c, d, e, to range over an alphabet W; u, v, w, z to range over W^* i.e. the set of all words (strings) over W including the <u>empty word</u> λ. #Z denotes the <u>cardinality</u> of a set Z; lg(z) denotes the <u>length</u> of a word z and $lg(\lambda) = 0$.

A <u>deterministic (m, n)L system (D(m, n)L)</u> is a triple $G = \langle W, \delta, w \rangle$ where W is a finite nonempty <u>alphabet</u>; δ is a total mapping from $\bigcup_{i=0}^{m} W^i \times W \times \bigcup_{j=0}^{n} W^j$ into W^*; $w \in WW^*$ is called the <u>axiom</u>. δ induces a total mapping $\overline{\delta}$ from W^* into W^* as follows: $\overline{\delta}(\lambda) = \lambda$ and for $k > 0$ holds that $\overline{\delta}(v) = v'$ iff $v = a_1a_2\ldots a_k$, $v' = \alpha_1\alpha_2\ldots\alpha_k$ and for all i, $i = 1, 2, \ldots, k$,

$$\delta(a_{i-m}a_{i-m+1}\ldots a_{i-1}, a_i, a_{i+1}a_{i+2}\ldots a_{i+n}) = \alpha_i,$$

where we take $a_j = \lambda$ for all j such that $j < 1$ or $j > k$. The composition of i copies of $\overline{\delta}$ is inductively defined by $\overline{\delta}^0(v) = v$ and $\overline{\delta}^i(v) = \overline{\delta}(\overline{\delta}^{i-1}(v))$ for $i > 0$. When no confusion can result we shall write δ for $\overline{\delta}$. The <u>L language</u> produced or generated by G is defined as $L(G) = \{\delta^i(w) \mid i \geq 0\}$.

At this stage we would like to point out that although our definition of an L system varies from the usual one, see e.g. [9], in that it dispenses with the environmental letter g, it

is exactly equivalent to the previous definitions. It has the additional advantages that proofs get shorter and the notation more transparent. With regard to the amount of context used the following terminology is standard throughout the literature: a D(0, 0)L is called a D0L; a D(0, 1)L or D(1, 0)L is called a D1L (one sided context); a D(1, 1)L is called a D2L (two sided context); a D(m, n)L such that $m + n > 0$ is called a DIL.

It was shown by van Dalen [3] that for a suitable standard definition of Turing machines (e.g. the quintuple version), for every Turing machine T with symbol set S and state set ψ we can effectively construct a D2L $G = \langle W, \delta, w \rangle$, $W = \psi \cup S$, which simulates it in real time, that is, the t^{th} instantaneous description of T is equal to $\delta^t(w)$.[1] If we do away with the excess blank symbols on the ends of the Turing machine tape, by letting the letters corresponding to the blank symbols derive the empty word λ in the L system simulation of T, then the following statement clearly holds. Let $G = \langle W, \delta, w \rangle$ be a D2L, S and ψ be disjoint subsets of W, and let h_1 be a homomorphism from $S^*\psi S^*$ into S^* defined by $h_1(a) = \lambda$ for all $a \in \psi$ and $h_1(a) = a$ for all $a \in S$. The set of languages of the form $h_1(L(G) \cap S^*\psi S^*)$ is the family of recursively enumerable languages. Since the family of recursive languages is closed under intersection with a regular set and k-limited erasing and since there exist recursively enumerable languages that are not recursive there

exist D2L languages which are not recursive.[2] ($S^*\psi S^*$ is regular and h_1 is 1-limited on $S^*\psi S^*$.) That all L languages considered in this paper are recursively enumerable follows by the usual Turing machine simulation argument.

3. NONRECURSIVE L LANGUAGES

At the end of the last section we gave the usual proof that there are non recursive D2L languages. By an application of a result due to Rabin and Wang [15] we can be slightly more specific and at the same time develop a simulation technique which will be of use in the sequel. Let the word at any moment t in the history of a Turing machine be the string consisting of the contents of the minimum block on the tape at t that includes all the marked squares and the square scanned at the initial moment (the origin).

Theorem 1. (Rabin and Wang). For any fixed (finite) word at the initial moment we can find a Turing machine T such that the set of words P in its subsequent history is not recursive.

Proof. Take a nonrecursive set $A \subseteq \{1\}^*$ enumerated by a one-one recursive function $f: N \xrightarrow{1:1} A$; we can recover n from $f(n)$ by f^{-1}. That every infinite recursively enumerable set can be enumerated by a one-one recursive function follows from Rogers [16, exercise 5.2]. We can now construct a Turing machine T with symbol set $S = \{b, 1, a\}$, where b is the blank symbol,such that T first erases the finitely many marks on the initial tape and returns to the origin, puts down the representation of 0 on the tape and calculates the value of $f(0)$. Subsequently, T erases everything else

except the representation of $f(0)$, retrieves the representation of 0 from $f(0)$ by f^{-1} adds one to this representation and computes $f(1)$, and so on. In particular we can do it in such a way that the specific symbol a is used, after the initial tape contents is erased, only to mark $f(0)$, $f(1)$, ...; it is erased before we calculate $f(n+1)$ from $f(n)$. Moreover, the string consisting of a followed by the representation of $f(n)$ always begins at the origin. Let h_2 be a homomorphism from $\{a\}\{1\}^*$ into $\{1\}^*$ defined by $h_2(a) = \lambda$ and $h_2(1) = 1$. Now $h_2(P \cap \{a\}\{1\}^*) = A$ where h_2 is 1-limited on $\{a\}\{1\}^*$ and $\{a\}\{1\}^*$ is regular. Since A is nonrecursive P must be nonrecursive by the closure of the recursive languages under k-limited erasing and intersections with regular sets. ■

Theorem 2. Let G_T be a D2L which simulates a Turing machine T satisfying the statement of Theorem 1 (in the sense explained in section 2). Then $L(G_T)$ is nonrecursive.

Proof. Let h_3 be a homomorphism on $L(G_T)$ defined by $h_3(s) = s$ and $h_3(q) = \lambda$ for all $s \in S$ and all $q \in \psi$, where S and ψ are the symbol set and the state set of T, respectively. Since $L(G_T) \subseteq S^*\psi S^*$ h_3 is 1-limited on $L(G_T)$. $h_3(L(G_T)) = P$ and since P is nonrecursive, $L(G_T)$ is nonrecursive. ■

We use G_T to construct a nonrecursive D(0, 1)L language.

Lemma 1. Let $G = \langle W, \delta, w\rangle$ be any D2L. There is an algorithm which, given G, produces a D(0, 1)L $G' = \langle W', \delta', w'\rangle$ such that for all t, $\delta'^{2t}(\text{¢}w) = \text{¢}\delta^t(w)$ and $\delta'^{2t+1}(\text{¢}w) = \text{¢}'(a_1, a_2)(a_2, a_3)\ldots(a_k, \lambda)$ if $\delta^t(w) = a_1a_2\ldots a_k$, where ¢ and ¢' are letters not in W.

Proof. Construct $G' = \langle W', \delta', w'\rangle$ as follows.

$$W' = W \cup (W \times (W \cup \{\lambda\})) \cup \{\text{¢}, \text{¢}'\},$$

where ¢ and ¢' are letters not in W.

$$w' = \text{¢}w.$$

$$\delta'(\lambda, a, c) = (a, c),$$
$$\delta'(\lambda, \text{¢}, c) = \text{¢}',$$
$$\delta'(\lambda, \text{¢}', \lambda) = \text{¢},$$
$$\delta'(\lambda, (a, b), (b, c)) = \delta(a, b, c),$$
$$\delta'(\lambda, \text{¢}', (a, c)) = \text{¢}\delta(\lambda, a, c),$$
$$\delta'(\lambda, (a, \lambda), \lambda) = \lambda,$$

for all $a, b \in W$ and all $c \in W \cup \{\lambda\}$. (The arguments for which δ' is not defined shall not occur in our operation of G'.)

For all words $v = a_1a_2\ldots a_k \in W^*$ holds:

$k > 1$: $\bar{\delta}'^2(\text{¢}a_1a_2\ldots a_k) = \bar{\delta}'(\text{¢}'(a_1, a_2)(a_2, a_3)\ldots(a_k, \lambda))$

$= \text{¢}\delta(\lambda, a_1, a_2)\delta(a_1, a_2, a_3)\ldots\delta(a_{k-1}, a_k, \lambda$

$= \text{¢}\bar{\delta}(a_1a_2\ldots a_k).$

$k = 1$: $\bar{\delta}'^2(\cent a_1) = \bar{\delta}'(\cent'(a_1, \lambda)) = \cent\delta(\lambda, a_1, \lambda) = \cent\bar{\delta}(a_1)$;

$k = 0$: $\bar{\delta}'^2(\cent) = \bar{\delta}'(\cent') = \cent = \cent\bar{\delta}(\lambda)$.

Therefore, for all t, $\bar{\delta}'^{2t}(\cent w) = \cent\bar{\delta}^t(w)$ and

$$\bar{\delta}'^{2t+1}(\cent w) = \cent'(a_1, a_2)(a_2, a_3)\ldots(a_k, \lambda)$$

if

$$\bar{\delta}^t(w) = a_1a_2\ldots a_k. \blacksquare$$

The following two corollaries illustrate the relation between D1L and D2L languages.

Corollary 1. Let $G = \langle W, \delta, w\rangle$ be a D2L. There is an algorithm which, given G, produces a D(0, 1)L G' (resp. a D(1, 0)L G'') and a letter to letter homomorphism h_4 such that $h_4(L(G')) = \{\cent\}L(G)$ (resp. $h_4(L(G'')) = L(G)\{\cent\}$). (Hint: Let h_4 be a letter to letter homomorphism defined by $h_4(a) = a$ for all $a \in W \cup \{\cent\}$, $h_4(\cent') = \cent$, and $h_4((a, b)) = a$ for all $(a, b) \in W \times (W \cup \{\lambda\})$.)

Corollary 2. Let $G = \langle W, \delta, w\rangle$ be any D2L. There is an algorithm which, given G, produces a D(0, 1)L G' (resp. D(1, 0)L G'') and a homomorphism h_5, which maps a letter either to itself or to λ, such that

$$h_5(L(G') \cap \{\cent\}W^*) = h_5(L(G'') \cap W^*\{\cent\}) = L(G).$$

(Hint: h_5 is defined by $h_5(a) = a$ for all $a \in W$ and $h_5(¢) = \lambda$. h_5 is 1-limited on $\{¢\}W^*$ and $W^*\{¢\}$.)

Theorem 3. We can construct D1LLs whose associated languages are not recursive.

Proof. Let $G_T = \langle W_T, \delta_T, w_T \rangle$ be a D2L as in Theorem 2. By Corollary 2 we can construct a D(0, 1)L G' such that $h_5(L(G') \cap \{¢\}W_T^*) = L(G_T)$. Since $\{¢\}W_T^*$ is regular, h_5 is a 1-limited homomorphism on $\{¢\}W_T^*$, and $L(G_T)$ is not recursive, it follows that L(G') is not recursive. ∎

4. DETERMINISTIC L LANGUAGES AND THE CHOMSKY HIERARCHY

A natural subclass of the L systems is formed by the propagating L systems. A deterministic L system $G = \langle W, \delta, w \rangle$ is propagating if for all arguments the value of δ is not equal to λ. We indicate this property by prefixing the capital P to the type of L system, e.g. PD(m, n)Ls, PD0Ls, PDILs. From the work of van Dalen [3], Rozenberg [17] and Rozenberg and Lee [18] on nondeterministic L systems we can readily deduce several facts about the place in the Chomsky hierarchy of the deterministic L languages: e.g. the PDIL languages are strictly included in the context sensitive languages, the DIL languages are strictly included in the recursively enumerable languages. By the use of direct arguments concerning the deterministic nature of the systems under consideration we shall refine these results implicit in the above references and fix completely the place of the D(m, n)L and PD(m, n)L languages with respect to the four main classes of the Chomsky hierarchy.

Lemma 2. There are regular languages over a one letter alphabet which are not DIL languages.

Proof. $L = (aaa)^*(a \cup aa)$ is such a language. To prove this we make use of the following:

Claim. If $G = \langle W, \delta, w\rangle$ is a unary $D(m, n)L$ (i.e. $\#W = 1$) which generates an infinite language then there exist positive integers t_0, p and x such that for all $t \geq t_0$ the following equation holds:

(1) $\lg(\bar{\delta}^{t+1}(w)) = p(\lg(\bar{\delta}^{t}(w)) - m - n) + x.$

Proof of Claim. Let $\delta(a^m, a, a^n) = a^p$ and let

$$x = \sum_{i=0}^{m-1} \lg(\delta(a^i, a, a^n)) + \sum_{j=0}^{n-1} \lg(\delta(a^m, a, a^j)).$$

If $L(G)$ is infinite then there exists a t_0 such that $\lg(\bar{\delta}^{t_0}(w)) \geq 2(m + n) + x + 1$.

Case 1. $p = 0$. $\lg(\bar{\delta}^{t}(w)) \leq y$ for all $t > 0$ where $y = \max\{\lg(\bar{\delta}(a^k)) \mid k \leq m + n\}$: contrary to the assumption.

Case 2. $p > 0$. Clearly (1) holds.

By observing that $L = \{a^i \mid i \not\equiv 0 \bmod 3\}$ we see that for every positive integer k such that $k \equiv 0 \bmod 3$ holds that $a^{k-1}, a^{k+1}, a^{k+2} \in L$ and $a^k \notin L$. Hence, if $L(G) = L$ it follows that $p = 1$ in (1). But then the lengths of the subsequent words in $L(G)$, ordered by increasing length, differ by a constant amount $x - m - n$ and hence $L(G) \neq L$. ∎

Let X be any of the restrictions on L systems discussed above. Then $\mathcal{L}(XL)$ denotes the family of XL languages, e.g. $\mathcal{L}(D(m, n)L)$, $\mathcal{L}(DIL)$, $\mathcal{L}(D0L)$. Let $\mathcal{L}(REG)$, $\mathcal{L}(CF)$, $\mathcal{L}(CS)$ and $\mathcal{L}(RE)$ denote the families of the regular, context free, context sensitive and recursively enumerable languages, respectively.

Theorem 4. (i) For all $m, n \geq 0$ the intersection of $L(PD(m, n)L)$ with $L(REG)$, $L(CF) - L(REG)$ and $L(CS) - L(CF)$ are nonempty; there are languages in $L(REG)$, $L(CF) - L(REG)$ and $L(CS) - L(CF)$ which are not in $L(PDIL)$; $L(PDIL) \subsetneq L(CS)$. (Fig. 1).

(ii) For all $m, n \geq 0$, such that $m + n > 0$, the intersections of $L(D(m, n)L)$ with $L(REG)$, $L(CF) - L(REG)$, $L(CS) - L(CF)$ and $L(RE) - L(CS)$ are nonempty; there are languages in $L(REG)$, $L(CF) - L(REG)$, $L(CS) - L(CF)$ and $L(RE) - L(CS)$ which are not in $L(DIL)$; $L(DIL) \subsetneq L(RE)$. (Fig. 2).

(iii) The intersections of $L(D0L)$ with $L(REG)$, $L(CF) - L(REG)$ and $L(CS) - L(CF)$ are nonempty; there are languages in $L(REG)$, $L(CF) - L(REG)$ and $L(CS) - L(CF)$ which are not in $L(D0L)$; $L(D0L) \subsetneq L(CS)$. (Fig. 3).

(iv). For all $m, n \geq 0$, $L(PD(m, n)L) \subsetneq L(D(m, n)L)$; $L(PDIL) \subsetneq L(DIL)$.

Proof. (i) and (ii). Let G_1, G_2 and G_3 be PD0Ls defined by:

$$G_1 = \langle \{a\}, \{\delta(\lambda, a, \lambda) = a\}, a \rangle,$$

$$G_2 = \langle \{a, b, c\}, \{\delta(\lambda, a, \lambda) = a, \delta(\lambda, b, \lambda) = b, \delta(\lambda, c, \lambda) = acb\}, c \rangle$$

$$G_3 = \langle \{a\}, \{\delta(\lambda, a, \lambda) = aa\}, a \rangle.$$

$L(G_1) = \{a\}$, $L(G_2) = \{a^n c b^n \mid n \geq 0\}$ and $L(G_3) = \{a^{2^n} \mid n \geq 0\}$. $L(G_1) \in L(REG)$; it is well known that $L(G_2) \in L(CF) - L(REG)$; $L(G_3) \in L(CS)$ by the working space theorem or the usual linear

bounded automaton argument and $L(G_3) \notin \mathcal{L}(CF)$ by the uvwxy lemma.[3] This proves that all considered families of languages have nonempty intersections with $\mathcal{L}(REG)$, $\mathcal{L}(CF) - \mathcal{L}(REG)$ and $\mathcal{L}(CS) - \mathcal{L}(CF)$. By Theorem 3 there is a DlL language $L(G)$ such that $L(G) \in \mathcal{L}(RE) - \mathcal{L}(CS)$. The language L of Lemma 2 belongs to $L(REG)$ but not to $\mathcal{L}(DIL)$. $L \cup L(G_2) \in \mathcal{L}(CF) - \mathcal{L}(REG)$ and it is easy to show that $L \cup L(G_2) \notin \mathcal{L}(DIL)$. $L' = \{a^{2^{(2^t)}} \mid t \geq 0\}$ does not belong to $\mathcal{L}(DIL)$ because of equation (1) but $L' \in \mathcal{L}(CS) - \mathcal{L}(CF)$ by the working space theorem and the uvwxy lemma. The language $A \subseteq \{1\}^*$ of Theorem 1 belongs to $\mathcal{L}(RE) - \mathcal{L}(CS)$ and $A \notin \mathcal{L}(DIL)$ by equation (1). Hence there are languages in $\mathcal{L}(REG)$, $\mathcal{L}(CF) - \mathcal{L}(REG)$, $\mathcal{L}(CS) - \mathcal{L}(CF)$ and $\mathcal{L}(RE) - \mathcal{L}(CS)$ which are not in $\mathcal{L}(DIL)$. From this it follows that the inclusions of $\mathcal{L}(PDIL)$ in $\mathcal{L}(CS)$ and of $\mathcal{L}(DIL)$ in $\mathcal{L}(RE)$ are strict.

(iii) Follows from the proof of (i) and (ii) and the observation that $\mathcal{L}(D0L) \subseteq \mathcal{L}(CS)$, Salomaa [19, p. 245].

(iv) $\mathcal{L}(PD(m, n)L) \subseteq \mathcal{L}(D(m, n),)$ holds by definition. Strict inclusion follows from the fact that if $\lambda \in L$ and $L \in \mathcal{L}(D(m, n)L)$ then $L \notin \mathcal{L}(PD(m, n)L)$. (It is easy to give nontrivial counterexamples of D0L languages which are not PD0L languages; for $m + n > 0$ there are nonrecursive $D(m, n)L$ languages by Theorem 3 and all $PD(m, n)L$ languages are context sensitive by (i)). Similarly we prove $\mathcal{L}(PDIL) \subsetneq \mathcal{L}(DIL)$. ∎

From equation (1) it follows immediately that $L(D(m, n)L) \subsetneq L(D(m', n')L)$ for $m < m'$ and $n = n'$ or $m = m'$ and $n < n'$. In particular $L(D0L) \subsetneq L(D1L) \subsetneq L(D2L)$. Analogously this holds with the propagating restriction added. For a further discussion of the inclusion relations between families of L languages using different amounts of context see [17] and [18].

From Lemma 1 we also have the following useful information concerning the difference between $L(D2L)$ and $L(D1L)$.

If $L \in L(D2L)$ then there is an $L' \in L(D(0, 1)L$ (resp.$L'' \in L(D(1, 0)L)$) such that $\{w \mid \¢w \in L'\} = L$ (resp. $\{w \mid w\¢ \in L''\} = L$).

5. EXTENSIONS OF DETERMINISTIC L LANGUAGES

The usual device in formal language theory for extracting languages from rewriting systems is the use of nonterminals, i.e. by selecting from the set of produced words all words over a terminal alphabet. This operation is called intersection with a terminal alphabet. Such an operation considerably contributes to the generating power and therefore a language $E(G, V_T) = L(G) \cap V_T^*$ is called an extension of an L language, where G is an L system and V_T is some alphabet. We denote the family of extensions of XL languages by $E(XL)$ where X is one of our usual restrictions. Considering nondeterministic L systems, van Dalen [3] proved that $E(1L) = L(RE)$, and $E(P2L) = L(CS)$. Furthermore, $E(0L) \subsetneq L(CS)$, see e.g. Herman and Rozenberg [9]. For deterministic L systems it therefore follows that $E(D1L) \subseteq E(D2L) \subseteq L(RE)$; $E(PD1L) \subseteq E(PD2L) \subseteq L(CS)$ (and in general by the working space theorem $E(PDIL) \subseteq L(CS)$); and $E(D0L) \not\subseteq L(CS)$. From the definitions it is immediate that $L(XL) \subseteq E(XL)$ for all classes of XL systems.

Theorem 5. $E(D2L) = L(RE)$.

Proof. Let A be any recursively enumerable language over some alphabet V_T which is enumerated by a 1:1 recursive function $f: N \xrightarrow{1:1} A$; n is recovered from $f(n)$ by f^{-1}. That every infinite recursively enumerable language can be enumerated by a one-one recursive function follows from

Rogers [16, exercise 5.2]; for finite languages clearly an appropriate version of our proof suffices. Let T be a Turing machine with symbol set $S = V_T \cup \{a, b\}$ where $a, b \notin V_T$ and b is the blank symbol. At time $t = 0$ T is presented with a finitely inscribed tape of which the origin contains a. We assume that the tape is halfway infinite, i.e. the reading head of T never scans a square left of the origin. That this is no restriction on the power of a Turing machine is well known. T starts with erasing the finitely many marks on its tape except the symbol a at the origin, returns to the origin, writes the representation of 0 on the tape and calculates the value of $f(0)$. Subsequently, T erases everything else except the representation of $f(0)$, retrieves the representation of 0 from $f(0)$ by f^{-1}, adds one to this representation and computes $f(1)$, and so on. In particular we can do this in such a way that the specific symbol a is used only to mark the origin and is erased only to indicate $f(0), f(1), \ldots$; it is printed again before we calculate $f(n + 1)$ from $f(n)$. If P is the set of all words in the history of T then $P \cap \{b\}V_T^* = \{b\}A$. Let $G_T = \langle W_T, \delta_T, w_T \rangle$ be a D2L which simulates T in the sense of Theorem 2. Since T uses a halfway infinite tape the strings of G_T always have a letter a at the left end except when $f(n)$ has been computed for some n in which case the string has a letter q_{i_n} (indicating the state of the simulated Turing machine) at the left end. That is, for each $n \in N$ there is a $t_n \in N$ and a state $q_{i_n} \in \psi$ (where ψ is the state set of T)

such that $\delta_T^{t_n}(w_T) = q_{i_n} af(n)$. We can construct T with two distinguished states q', q" in ψ such that for all n: $\delta_T^{t_n+1}(w_T) = q'f(n)$, $\delta_T^{t_n+2}(w_T) = aq"f(n)$, and q', q" never occur in $\delta_T^t(w_T)$ for $t_n + 2 < t \leq t_{n+1}$, $n \in N$. Now we modify G_T to $G = \langle W_T, \delta, w_T \rangle$ where δ is exactly like δ_T but for the productions $\delta(\lambda, q, a) = \lambda$ if $\delta_T(\lambda, q, a) = q'$ and $\delta(\lambda, c, d) = aq"$ for all letters $c \in V_T$ and $d \in V_T \cup \{\lambda\}$. It is easily seen that $\delta^{t_n+1}(w_T) = f(n)$ for all n and $\delta^t(w_T) = \delta_T^t(w_T) \in W_T^* \psi W_T^*$ for all t such that $t \neq t_n + 1$, $n \in N$. Hence $L(G) \cap V_T^* = A$. (To capture the case where $\lambda \in A$ we could define $\overline{\delta}(\lambda) = aq"$.) ■

Theorem 6. The closure of $E(D(0, 1)L)$ (or $E(D(1, 0)L)$) under letter to letter homomorphism is equal to $L(RE)$.

Proof. We prove the theorem for D(0, 1)Ls. The case for D(1, 0)Ls is completely analogous. Let $G = \langle W, \delta, w \rangle$ be a D2L constructed as in Theorem 5. Let $G' = \langle W', \delta', w' \rangle$ be a D(0, 1)L defined as follows.

$$W' = W \cup (W \times (W \cup \{0, 1, \lambda\})) \cup \{¢\}$$

where 0, 1, ¢ are letters not in W.

$w' = (b_1, 1)(b_2, 0)\ldots(b_n, 0)$ if $w = b_1 b_2 \ldots b_n$.

$\delta'(\lambda, a, b) = (b, 0)$,

$\delta'(\lambda, ¢, a) = (a, 1)$,

$\delta'(\lambda, ¢, \lambda) = \delta'(\lambda, a, \lambda) = \delta'(\lambda, (a, \lambda), \lambda) = \lambda$,

$\delta'(\lambda, (a, 0), (b, 0)) = (a, b)$,

$\delta'(\lambda, (a, 1), (b, 0)) = \cent(a, b)$,

$\delta'(\lambda, (a, 0), \lambda) = (a, \lambda)$,

$\delta'(\lambda, (a, 1), \lambda) = \cent(a, \lambda)$,

$\delta'(\lambda, (a, b), (b, c)) = \delta(a, b, c)$,

$\delta'(\lambda, \cent, (a, c)) = \cent\delta(\lambda, a, c)$,

for all $a, b \in W$ and all $c \in W \cup \{\lambda\}$. (The arguments for which δ' is not defined shall not occur in our operation of G'.) Assume that $\lambda \notin L(G)$.

We see that for all t holds that $h_6(\delta'^{3t}(w')) = \delta^t(w)$ where h_6 is a letter to letter homomorphism from $(W \times \{1, 0\})^*$ onto W^* defined by $h_6((a, 0)) = h_6((a, 1)) = a$ for all $a \in W$. Since by the synchronicity of the productions $\delta'^t(w') \in \{\cent\}W'^*$ for all $t \not\equiv 0 \bmod 3$ we have $h_6(L(G') \cap (W \times \{0, 1\})^*) = L(G)$ and therefore $h_6(L(G') \cap (V_T \times \{0, 1\})^*) = L(G) \cap V_T^*$. (To capture the case where $\lambda \in L(G)$ we could define $\bar{\delta}'(\lambda) = \cent\bar{\delta}(\lambda)$, and the proof proceeds analogously.) ■

Theorem 7. If $L \in E(D2L)$, or equivalently $L \in L(RE)$, then $\{\cent\}L \in E(D(0, 1)L$ (similarly $L\{\cent\} \in E(D(1, 0)L)$ where $\cent$ is a letter not occuring in a word in L.

Proof. Follows immediately from Lemma 1.

We shall now prove some properties of D0L and D1L languages which give us criteria to show that certain languages

cannot be D0L or D1L languages or their intersections with a terminal alphabet.

We call a language permutation free if no word in the language is a permutation of any other word in the language.

Lemma 3. Let $G = \langle W, \delta, w \rangle$ be a D0L. If $L(G)$ is infinite then $L(G)$ is permutation free.

Proof. Suppose $L(G)$ is infinite, $v, v' \in L(G)$, $v \neq v'$, and v' is a permutation of v. Let $\delta^k(v) = v'$ for some $k > 0$. Since v' is a permutation of v we have for each $n > 0$: $\delta^{nk}(v)$ is a permutation of v. There are only a finite number of words in W^* which are a permutation of v and therefore there exist $n_2 > n_1 > 0$ such that $\delta^{n_1 k}(v) = \delta^{n_2 k}(v)$. But $v = \delta^{t_0}(w)$ for some t_0 and therefore $\delta^{t_0+n_1 k}(w) = \delta^{t_0+n_2 k}(w)$ so $L(G)$ is finite: contradicting the assumption. ∎

The converse of the lemma holds in the following sense. Let $G = \langle W, \delta, w \rangle$ be a D0L. $L(G)$ is infinite iff for no integers i and j, $i \neq j$, holds that $\delta^i(w)$ is a permutation of $\delta^j(w)$. (We consider λ to be a permutation of λ.).

Corollary 3. Let $G = \langle W, \delta, w \rangle$ be a D0L and V_T a subset of W. If $E(G, V_T)$ is infinite then $E(G, V_T)$ is permutation free, i.e. all infinite languages in E(D0L) are permutation free.

We call a word v' a prefix (postfix) of a word v if $v = v'z$ ($v = zv'$) for some word z. We call v' a proper prefix (proper postfix) of a word v if v' is a prefix (postfix) of v and $v' \neq v$.

Lemma 4. Let $G = \langle W, \delta, w \rangle$ be a D(1, 0)L (D(0, 1)L).
(i) $L(G)$ is finite iff $\delta^t(w) = \delta^{t'}(w)$ for some t, t' such that $t \neq t'$.
(ii) Let $L(G)$ be infinite. If $v, v' \in L(G)$ and v' is a prefix (postfix) of v then, with finitely many exceptions, for each word u in $L(G)$ there is a word u' in $L(G)$ such that u' is a proper prefix (postfix) of u.

Proof. (i) Obvious by the deterministic property of G.
(ii) We prove (ii) only for D(1, 0)Ls and prefixes. The proof is completely analogous for D(0, 1)Ls and postfixes. Since $L(G)$ is infinite $v \neq v'$ by (i).

Case 1. $\delta^t(w) = v'$ and $\delta^k(v') = v = v'z$ for some $t \geq 0$ and some $k > 0$. For each $j \geq 0$ there is a $z' \in W^*$ such that $\delta^{t+k+j}(w) = \delta^j(v) = \delta^j(v'z) = \delta^j(v')z' = \delta^{t+j}(w)z'$, and by (i) $z' \neq \lambda$.

Case 2. $\delta^t(w) = v = v'z$ and $\delta^k(v'z) = v'$ for some $t \geq 0$ and some $k > 0$. $\delta^k(v'z) = \delta^k(v')z' = v'$ for some $z' \in W^*$ and by (i) $z' \neq \lambda$. Therefore, $lg(\delta^k(v')) < lg(v')$. By iterating this argument $lg(v') + 1$ times we obtain either $lg(\delta^{k(lg(v')+1)}(v')) < lg(v') - lg(v')$ which is impossible or $\delta^{klg(v')}(v') = \delta^{k(lg(v')+1)}(v')$. In the latter case $L(G)$ is finite: contradictory to the assumption. ∎

(If we allow $\overline{\delta}(\lambda) \neq \lambda$ then Lemma 4(ii) holds under the additional restriction: not both $\lambda \in L(G)$ and $\overline{\delta}(\lambda) \neq \lambda$.)

Corollary 4. Let $G = \langle W, \delta, w \rangle$ be a D(1, 0)L (D(0, 1)L) such that $E(G, V_T)$ is infinite for some V_T (and not both $\lambda \in L(G)$ and $\overline{\delta}(\lambda) \neq \lambda$). If $v, v' \in E(G, V_T)$ such that v' is a prefix of v (v' is a postfix of v) then, with finitely many exceptions, for each word u in $E(G, V_T)$ there is a word u' on $E(G, V_T)$ such that $u = u'z$ ($u = zu'$) for some $z \in V_T V_T^*$.

Clearly, Lemma 4 and Corollary 4 hold for D(m, 0)Ls with respect to prefixes and for D(0, m)Ls with respect to postfixes, $m \geq 0$.

Theorem 8. (i) The intersection of E(PDlL) with L(REG), L(CF) - L(REG) and L(CS) - L(CF) are nonempty. There are languages in L(REG), L(CF) - L(REG) and L(CS) - L(CF) which are not in E(PDlL). E(PDlL) $\subsetneq L$(CS).

(ii) The intersections of E(DlL) with L(REG), L(CF) - L(REG), L(CS) - L(CF) and L(RE) - L(CS) are nonempty. There are languages in L(REG), L(CF) - L(REG), L(CS) - L(CF) and L(RE) - L(CS) which are not in E(DlL). E(DlL) $\subsetneq L$(RE).

(iii) The intersections of $E(D0L)$ with $L(REG)$, $L(CF) - L(REG)$ and $L(CS) - L(CF)$ are nonempty. There are languages in $L(REG)$, $L(CF) - L(REG)$ and $L(CS) - L(CF)$ which are not in $E(D0L)$. $E(D0L) \subsetneq L(CS)$.

Proof. Since $L(DXL) \subseteq E(DXL)$ the first sentences of the statements (i) - (iii) are correct by Theorem 4. Let $L_1 = \{a, aa\} \cup \{b\}\{c\}^*\{b\}$, $L_2 = \{a, aa\} \cup \{a^n b c^n \mid n > 0\}$, $L_3 = \{a, aa\} \cup \{b^n c^n d^n \mid n > 0\}$ and $L_4 = \{a, aa\} \cup \{a\}A\{a\}$ where $A \subseteq \{1\}^*$ is the nonrecursive language from Theorem 1. By Corollary 4 L_1, L_2, L_3 and L_4 do not belong to $E(D1L)$, but $L_1 \in L(REG)$, $L_2 \in L(CF) - L(REG)$ as is well known, $L_3 \in L(CS) - L(CF)$ as is well known and $L_4 \in L(RE) - L(CS)$. The inclusion in the last sentences of the statements of (i) and (iii) follows by the usual working space theorem and strict inclusion by the foregoing. The inclusion in the last sentence of the statement of (ii) is true by the usual Turing machine simulation argument and strict inclusion follows by the foregoing. ■

We might note that the existence of languages in $L(REG)$, $L(CF) - L(REG)$ and $L(CS) - L(CF)$ which are not in $E(D0L)$ could also have been proven using Corollary 3.

That with respect to families of extensions of L languages differences can only lie in no context, one directional context

and two directional context, but not in the amount of context, is shown by the next theorem.

Theorem 9.

(i) $E(D2L) = E(DIL)$.

(ii) $E(PD2L) = E(PDIL)$.

(iii) $E(D1L) = \bigcup_{i\in N} (E(D(i, 0)L) \cup E(D(0, i)L))$.

(iv) $E(PD1L) = \bigcup_{i\in N} (E(PD(i, 0)L) \cup E(PD(0, i)L))$.

Proof. We give the outline of a simulation technique to prove (i). (ii) - (iv) are completely analogous. ((i) follows also from Theorem 5 but the present proof is direct.)

Let $G = \langle W, \delta, w\rangle$ be a $D(m, n)L$ and let r be the greatest one of m and n. We construct a $D2L$ $G' = \langle W', \delta', w'\rangle$ as follows. $W' = W \cup (\bigcup_{i=0}^{m} W^i \times W \times \bigcup_{j=0}^{n} W^j)$ and $w' = w$. The production rules δ' are defined in such a way that, for each production of G, G' executes r productions. The first $r - 1$ of these r productions serve to gather the necessary context for each letter in the string and the r^{th} production produces the string produced by G.

E. g. If $\delta(a_1a_2\ldots a_k) = \alpha_1\alpha_2\ldots\alpha_k$ then

$$\delta'^{r}(a_1a_2\ldots a_k) = \delta'^{r-1}((\lambda, a_1, a_2)(a_1, a_2, a_3)\ldots(a_{k-1}, a_k, \lambda))$$

$$= \delta'^{r-2}((\lambda, a_1, a_2a_3)(a_1, a_2, a_3a_4)\ldots (a_{k-2}a_{k-1}, a_k, \lambda))$$

$$\ldots$$

$$= \delta'((\lambda, a_1, a_2a_3\cdots a_n)(a_1, a_2, a_3a_4\cdots a_{n+1})$$
$$\cdots(a_{k-m+1}a_{k-m+2}\cdots a_{k-1}, a_k, \lambda))$$
$$= \alpha_1\alpha_2\cdots\alpha_k.$$

Therefore, $\delta'^{tr}(w') = \delta^t(w)$ for all t, and $\delta'^t(w) \notin W^*$ for all $t \not\equiv 0 \bmod r$. Hence, for each subset V_T of W, $L(G') \cap V_T^* = L(G) \cap V_T^*$. ■

Similarly we can prove the analog of Theorem 9 for the general case of nondeterministic L systems.

In the next section we study $\mathcal{E}$(PD2L) and show, among other things, that the closure of $\mathcal{E}$(PD1L) under nonerasing homomorphism is strictly contained in $\mathcal{E}$(PD2L).

6. EXTENSIONS OF PROPAGATING DETERMINISTIC L LANGUAGES

A linear bounded automaton M is a Turing machine with, say, symbol set S, state set ψ and start state $q_0 \in \psi$ such that M accepts a word v over a subset V_T of S using at most c lg(v) tapesquares during its computation, where c is a fixed constant. It is well known that the family of languages accepted by linear bounded automata is equal to $\mathcal{L}$(CS), [11] or [19]. A deterministic linear bounded automaton (DLBA) is a linear bounded automaton such that each instantaneous description has exactly one successor.

We shall show that $\mathcal{E}$(PD2L) equals the family of languages accepted by DLBA's, i.e. $\mathcal{L}$(DLBA). Thus the question of whether or not the inclusion of $\mathcal{E}$(PD2L) in $\mathcal{E}$(P2L) is strict is shown to be equivalent with one of the more famous open problems in formal language theory, i.e. whether or not the inclusion of $\mathcal{L}$(DLBA) in $\mathcal{L}$(CS) is strict, [11] or [19]. That $\mathcal{E}$(PD1L) $\subsetneq$ $\mathcal{E}$(PD2L) follows already from the fact that it is easy to construct a PD2L G such that L(G) = {a, aa} $\cup$ {b}{c}*{b} which language is not in $\mathcal{E}$(PD1L) by Corollary 4. However, we shall prove the much stronger result that the closure of $\mathcal{E}$(PD1L) under nonerasing homomorphisms is strictly contained in $\mathcal{E}$(PD2L).

Theorem 10. $E(PD2L) = L(DLBA)$.

Proof. We give an outline since the details would be tedious. Let $G = \langle W, \delta, w \rangle$ be a PD2L and V_T a subset of W. Construct a deterministic linear bounded automaton M as follows. M uses an amount of tape equal to 4 times the length of its input word plus 1, divided in 4 sections I, II, III, IV of equal length. The input word v is written on I; section II contains the axiom w, section III is blank and section IV contains the representation of 0 in the $\#W$-ary number system. M compares $\delta^i(w)$ with v, $i \geq 0$, and accepts v if $\delta^i(w) = v$. Otherwise, scuttling back and forth between sections II and III, M produces $\delta^{i+1}(w)$ from $\delta^i(w)$ such that $\delta^{i+1}(w)$ is written on III if $\delta^i(w)$ is written on II and vice versa. (If $lg(\delta^{i+1}(w)) \geq lg(v) + 1$ then M rejects v.) Subsequently, M increments the number written on IV by 1. If IV contains a number equal to $\#W^{lg(v)+1} - 1$ then M rejects v. Otherwise, M compares $\delta^{i+1}(w)$ with v, and so on. Since $v \in L(G)$ iff $v = \delta^i(w)$ for some $i < \#W^{lg(v)+1} - 1$ we see that $L(M) = L(G)$, where $L(M)$ is the language accepted by M. Now construct M' from M where M' is exactly like M except that M' first ascertains that $v \in V_T^*$ and rejects v if $v \notin V_T^*$. Then $L(M') = L(G) \cap V_T^*$,

Let M be a DLBA, which accepts $L(M)$ over S, using no more than cn tapesquares for an input word of length n. Now construct a DLBA M' such that M' generates all words $v_0, v_1, \ldots$ over S in lexicographical order and accepts or rejects them by simulating M. In particular we can do it

such that M', started in state q_0' on a word v_i, $i \geq 0$, written from left to right from the origin with the remaining $(c - 1)lg(v_i)$ tape squares containing blank symbols, computes the next word v_{i+1} written from left to right from the origin with the remaining tapesquares containing blank symbols. Subsequently, M' proceeds to the origin, enters the start state q_0 of M and simulates M. After rejection or acceptance M erases everything but v_{i+1} from the tape and starts in q_0' at the origin, i.e. scanning the left most letter of v_{i+1}, and so on.

Let V be the set of symbols of M', b the blank symbol, and ψ the state set of M'. Construct $G = \langle W, \delta, w \rangle$ as follows.

$$W = V \cup (V^c \times (\psi \cup \{\lambda\}) \times \{0, 1, 2, \ldots, c\}),$$

$$w = (a, b, b, \ldots, b, q_0, 1),$$

where a is the first word of SS* in the lexicographical order. G simulates M' as follows: if $\delta^t(w) = \underset{\sim}{a}_1\underset{\sim}{a}_2\ldots\underset{\sim}{a}_n$,

$$\underset{\sim}{a}_1\underset{\sim}{a}_2\ldots\underset{\sim}{a}_n \in (V^c \times \{\lambda\} \times \{0\})^*(V^c \times \psi \times \{1,2,\ldots,c\})(V^c \times \{\lambda\} \times \{0\})^*$$

then the j^{th} element of a_i, $1 \leq j \leq c$ and $1 \leq i \leq n$, corresponds with the $i + (j - 1)n^{th}$ tape square of M', the $c + 1^{th}$ element of $\underset{\sim}{a}_i$ indicates the present state of M' if one of the tapesquares coded in $\underset{\sim}{a}_i$ is under scan (and is λ otherwise) and the $c + 2^{th}$ element tells which (and is 0 otherwise). In particular we can construct G such that if M' enters an accepting state the accepted word

v_i over S is "read out" from right to left, and subsequently is restored (from left to right) to the form $(a_1,b,b,\ldots,b,q_0',1)$ $(a_2,b,b,\ldots,b,\lambda,0)\ldots(a_n,b,b,\ldots,b,\lambda,0)$ for $v_i = a_1a_2\ldots a_n$. Hence $L(G) \cap S^* = L(M)$. ∎

We now proceed to show that the closure of $\mathcal{E}$(PDlL) under nonerasing homomorphism does not contain $\mathcal{L}$(REG).

Lemma 5. Let $G = \langle W, \delta, w\rangle$ be a PD(1, 0)L such that L(G) is infinite. Let $r = \#W$. For each $t \geq r$ there is a prefix v of $\delta^t(w)$, $lg(v) \geq \lfloor \log_r((r-1)t + r) \rfloor$, and a constant k, $0 < k \leq r^{lg(v)}$, such that v is a prefix of $\delta^{t+nk}(w)$ for all n. For PD(0, 1)Ls this holds with respect to postfixes.

Proof. Denote the i^{th} letter of a string $\delta^j(w)$, $i, j \in N$, by a_{ij}. Since L(G) is infinite, the slowest rate of growth G can achieve is by generating all words over W in lexicographical order, i.e. $lg(\delta^t(w)) \geq \lfloor \log_r((r-1)t + r) \rfloor$. Therefore, a_{ij} is indeed a letter in W for all j such that $j \geq \sum_{\ell=1}^{i-1} r^\ell$. Since there are only r different letters in W, there are natural numbers j_1 and k_1, $j_1, k_1 \leq r$ and $k_1 > 0$ such that $a_{1j_1} = a_{1j_1+k_1}$. Since G is a

PD(1, 0)L, $a_{1j_1+nk_1} = a_{1j_1}$, for all n. Therefore, a letter

in the second position has a_{1j_1} as its left neighbor at all

times, $j_1 + nk_1$, $n \in N$. There is surely a letter in the second position for all times $t \geq r$. Therefore, there are positive natural numbers j_2 and k_2, $j_2 \geq r$, $k_2 \leq r^2$ and $j_2 + k_2 \leq r + r^2$, such that $j_2 = j_1 + n_1k_1$ $j_2 + k_2 = j_1 + n_2k_1$ for some $n_1, n_2 \in N$ and $a_{2j_2} = a_{2j_2+k_2}$. By iteration of this argument, for each $s = 1, 2, \ldots$ there are positive natural numbers j_s and k_s, $j_s \geq \sum_{i=1}^{s-1} r^i$, $k_s \leq r^s$ and $j_s + k_s \leq \sum_{i=1}^{s} r^i$, such that

$$a_{1j_1}a_{2j_2}\cdots a_{sj_s} = a_{1j_s+nk_s}\, a_{2j_s+nk_s} \cdots a_{sj_s+nk_s},$$

for all n. Since G is a PD(1, 0)L,

$$a_{1j_s+t}\, a_{2j_s+t} \cdots a_{sj_s+t} = a_{1j_s+t+nk_s} a_{2j_s+t+nk_s} \cdots a_{sj_s+t+nk_s}$$

for all t and n. Therefore, for all s and all t such that $\sum_{i=1}^{s} r^i > t \geq j_s \geq \sum_{i=1}^{s-1} r^i$, there is a prefix v of $\delta^t(w)$, $\lg(v) \geq \lfloor \log_r((r-1)t + r) \rfloor = s$, and a positive constant $k_s \leq r^s$ such that v is a prefix of $\delta^{t+nk_s}(w)$ for all n. Hence the lemma. ∎

Contrasting Lemma 5 with Lemma 4 gives a nice insight in the influence of the propagating restriction with respect to

the necessary behavior of pre- and postfixes of the sequence of words generated by DlLs.

Theorem 11. Let V be any alphabet containing at least two letters. No language containing VV^* belongs to the closure under nonerasing homomorphism of $\mathcal{E}(\text{PDlL})$.

Proof. Assume that $\{a, b\} \subseteq V$, and consider the subset $L = \{(a^n b^n)^{n^n} \mid n \geq 1\}$ of V^*. Suppose that $L \subseteq h(L(G) \cap V_T^*)$ for some PD(1, 0)L $G = \langle W, \delta, w\rangle$, a set V_T and a nonerasing homomorphism h from V_T^* into V^*. Define t_n by

$$t_n = \min\{i \in N \mid \delta^i(w) \in V_T^* \text{ and } h(\delta^i(w)) = (a^n b^n)^{n^n}, n \in N\}.$$

As is easily seen, $\lg(\delta^t(w)) \leq m^t \lg(w)$ where m is the maximum length of a value of δ. Therefore, $(2n)^{n^n} \leq m^{t_n} \lg(w) c$ where $c = \max\{\lg(h(a)) \mid a \in V_T\}$. Or, $t_n \geq n^n \log_m(2n/(\lg(w)c)) > n^n$ for all $n \geq n_0$ where n_0 is some fixed natural number. For each $n \geq n_0$ $\delta^{t_n}(w)$ has a prefix v_n such that $\lg(v_n) \geq \lfloor \log_r(t_n(r-1) + r) \rfloor \geq n \log_r n$, $r = \#W$, and v_n occurs infinitely often with a constant period k_n by Lemma 5. Since for each n the prefix v_n of $\delta^{t_n}(w)$ is mapped under h to $a^n bz$, $z \in \{a, b\}^*$, v_n cannot be a prefix of $\delta^{t_{n'}}(w)$ for $n \neq n'$ and $n, n' \geq n_0$. We now derive a contradiction by showing that the $k_n = k_{n_0}$ for all $n \geq n_0$. Since G is propagating and the prefix v_n $(n \geq n_0)$ occurs with a constant period k_n there is a j_n such that $\delta^{j_n}(v_{n_0}) = v_n z$ for some $z \in W^*$. But then

$$\delta^{t_{n_0}+pk_{n_0}+j_n}(w) = \delta^{j_n}(v_{n_0}z_p) = \delta^{j_n}(v_{n_0})z_p' = v_n z z_p' \quad \text{for}$$

all p and some $z, z_p, z_p' \in W^*$. I.e. from time $t_{n_0} + j_n$ the prefix v_n occurs with period k_{n_0} and $k_n = k_{n_0}$ (or k_n divides k_{n_0}) for all $n \geq n_0$. Hence

$$h(L(G) \cap V_T^*) \cap \{(a^n b^n)^{n^n} \mid n \geq 1\} \leq k_{n_0}$$

and

$$VV^* \not\subseteq h(L(G) \cap V_T^*).$$

(Since $VV^* = (VV^*)^R$, i.e. the language consisting of all words from VV^* reversed, the above proof holds also for $PD(0, 1)Ls$.). ∎

We see that any language which contains a language like $\{(a^n b^n)^{n^n} \mid n \geq 1\}$ cannot be the image under nonerasing homomorphism of a language in $\mathcal{E}(PDlL)$. Hence also e.g. $(\{a\}\{a\}^*\{b\}\{b\}^*)^*$. The idea behind the proof of Theorem 11 is roughly the following. If a language L contains a large enough subset L' where each pair of words in L', say v and v', are distinguishable by their resp. prefixes (postfixes) u and u' such that $lg(u) = O(\log \log(lg(v)))$

and $lg(u') = O(\log \log(lg(v')))$ then L cannot be in the closure under nonerasing homomorphism of $E(PD(1, 0)L)$ ($E(PD(0, 1)L)$. For example $\{b\}\{b\}^*\{a\}^*\{b\}\{b\}^*$ contains $\{b^n(a^n)^{n^n}b^n \mid n \geq 1\}$ and therefore is not contained in a nonerasing homomorphic image of a language in $E(PD1L)$.

Let us denote the closure of a language family X under nonerasing homomorphism by $h_\lambda X$ and under letter to letter homomorphism by $h_{1:1}X$.

Theorem 12. (i) $E(PD1L) \subsetneq h_{1:1}E(PD1L) \subseteq h_\lambda E(PD1L) \subsetneq E(PD2L) = L(DLBA) = h_\lambda E(PD2L)$.

(ii) For each $x \in \{\lambda, h_{1:1}, h_\lambda\}$ the language family $xE(PD1L)$ has nonempty intersection with $L(REG)$, $L(CF) - L(REG)$ and $L(CS) - L(CF)$; there are languages in $L(REG)$, $L(CF) - L(REG)$ and $L(CS) - L(CF)$ which are not in $xE(PD1L)$; $h_\lambda E(PD1L) \subsetneq L(DLBA)$.

Proof. (i) Let $G = \langle\{a_1, a_2, a_3, b, c\}, \{\delta(\lambda, a_1, \lambda) = a_2a_3, \delta(\lambda, a_2, \lambda) = \delta(a_2, a_3, \lambda) = \delta(\lambda, b, \lambda) = b, \delta(b, b, \lambda) = \delta(c, b, \lambda) = cb, \delta(b, c, \lambda) = c\}, a_1\rangle$ be a $PD(1, 0)L$. Let h be a letter to letter homomorphism defined by $h(a_i) = a$ for $i = 1, 2, 3$ and $h(b) = b$, $h(c) = c$. $h(L(G)) = \{a, aa\} \cup \{b\}\{c\}^*\{b\}$ and by Corollary 4 $h(L(G)) \notin E(PD1L)$. Therefore, $E(PD1L) \subsetneq h_{1:1}E(PD1L)$. $h_{1:1}E(PD1L) \subseteq h_\lambda E(PD1L)$ holds by definition. It is easy to show that $L(DLBA) = h_\lambda L(DLBA)$; together with Theorem 10 this gives $E(PD2L) = h_\lambda E(PD2L) = L(DLBA)$. Since $E(PD1L) \subseteq E(PD2L)$, we have $h_\lambda E(PD1L) \subseteq E(PD2L)$. $L(CF) \subsetneq L(DLBA)$ [11, exercise 3.3], and therefore $\{a, b\}\{a, b\}^* \in E(PD2L)$ and by Theorem 11 $\{a, b\}\{a, b\}^* \notin h_\lambda E(PD1L)$. Hence $h_\lambda E(PD1L) \subsetneq E(PD2L)$.

(ii). Since $L(\text{PD1L}) \subseteq xE(\text{PD1L})$ the first sentence follows from Theorem 4. The second sentence follows by taking languages from $L(\text{REG})$, $L(\text{CF}) - L(\text{REG})$, $L(\text{CS}) - L(\text{CF})$ forming their union with $\{a, b\}\{a, b\}^*$ and applying Theorem 11. The last sentence follows from (i). ∎

In the foregoing we have seen that with deterministic propagating one directional L systems, together with nonterminal mechanisms and nonerasing homomorphisms, we stay within the range of the DLBA languages and cannot even obtain all regular languages. We conclude by proving that the closure of $L(\text{PD1L})$ under homomorphisms, which map a letter either to itself or to λ, is equal to the family of recursively enumerable languages.

The proof method was suggested by a proof of Ehrenfeucht and Rozenberg [5] for the equality of $L(\text{RE})$ and the closure of $L(\text{D2L})$ under weak coding. The difficulty lies in the fact that we have to "read out" the whole word in the language in one production since otherwise also subwords of the desired words appear under the homomorphism. The solution makes essential use of the parallelism in L systems by a firing squad simulation. The firing squad synchronization problem, see e.g. Minsky [13], can be stated as follows. Suppose we want to synchronize an arbitrary long finite chain of interacting identical finite state automata. All finite state automata are initially in the same state m and stay in that state if both neighbors are in state m. The automata on the ends of the chain are allowed to be different since they sense that they lack one

neighbor. Synchronization is achieved if all automata enter the firing state f at the same time and no automaton in the chain is in state f before that time. In the terminology of L systems a firing squad is a PD2L $F = \langle W_F, \delta_F, m^k \rangle$ such that $\delta_F(m, m, m) = \delta_F(m, m, \lambda) = m$. F satisfies the following requirement: there is a function $t: N \to N$ such that for each $k \in N$ holds that $\delta_F^{t(k)}(m^k) = f^k$ and $\delta_F^i(m^k) \notin W_F^*\{f\}W_F^*$ for all i, $0 \leq i < t(k)$. Balzer [2] proved that there is such an F with $\#W_F = 8$ and $t(k) = 2k - 2$. After these preliminaries we state the theorem.

Theorem 13. The closure of L(PD1L) under homomorphisms, which map a letter either to itself or to λ, is equal to L(RE).

Proof. Since by now these kinds of proofs are familiar we give only an outline. Let A be any recursively enumerable language enumerated by a 1:1 recursive function $f: N \xrightarrow{1:1} A$; n is recovered from $f(n)$ by f^{-1}. (The case where A is finite follows by a similar method.) Let T be a Turing machine which starts with the representation of 0 on its tape, say $a_1a_2\ldots a_{n_0}$, computes $f(0)$, replaces everything except $f(0)$ on its tape by the blank symbol b and returns to the left most symbol of $f(0)$. Subsequently T retrieves 0 from $f(0)$ by f^{-1}, increments 0 with 1, and computes $f(1)$, and so on. In particular we can do this in such a way that after the computation of $f(n)$ the instantaneous description of T is $b^\ell q' f(n) b^r$ for some $\ell, r \in N$ and a distinguished state q' of T. The next instantaneous

description of T is $b^{\ell}q"f(n)b^{r}$ for another distinguished state q" of T. Scanning the leftmost symbol of f(n), T starts retrieving n from f(n) by f^{-1} in state q". We simulate T by a PD2L $G = \langle W, \delta, w \rangle$; hence the blank symbols will not disappear. G is defined as follows:

$$W = (\psi \times S \cup (S - \{b\})) \times W_F \cup S,$$

where ψ is the state set of T, S is the symbol set of T and b is the blank symbol, and W_F is the alphabet of the firing squad F.

$$w = (q_0, a_1, m)(a_2, m)\ldots(a_{n_0}, m),$$

where q_0 is the start state of T, $a_1a_2\ldots a_{n_0}$ is the representation of 0 and m is the initial state of the firing squad F. G simulates T until the situation $\delta^{t_0}(w) = b^{\ell}(q', c_1, m)(c_2, m)\ldots(c_{\ell_0}, m)b^{r}$ occurs where $c_1c_2\ldots c_{\ell_0}$ is f(0). Subsequently, the substring between the b's executes a firing squad and, when the squad fires maps itself to f(0). I.e.

$$\delta^{t_0+2\ell_0-2}(w) = b^{\ell}(q', c_1, f)(c_2, f)\ldots(c_{\ell_0}, f)b^{r},$$

$$\delta^{t_0+2\ell_0-1}(w) = b^{\ell}c_1c_2\ldots c_{\ell_0}b^{r} = b^{\ell}f(0)b^{r}.$$

δ is constructed such that a letter $c \in S - \{b\}$ is rewritten as (c, m), except when it has b or λ as left neighbor in which case it is rewritten as (q", c, m). Therefore, $\delta^{t_0+2\ell_0}(w) = b^{\ell}(q", c_1, m)(c_2, m)\ldots(c_{\ell_0}, m)b^{r}$, and G continues simulating T, retrieves 0 adds one and computes the representation of f(1), and so on. Hence h(L(G)) = A where h is a homomorphism defined by h(a) = a if $a \in S - \{b\}$ and $h(a) = \lambda$ otherwise.

We now simulate G by a PDlL $G' = \langle W', \delta', w' \rangle$ which is defined exactly as the $D(0, 1)L$ in Lemma 1 except that $\delta'(\lambda, (a, \lambda), \lambda) = b$ for all $a \in W$. Then $h'(L(G')) = A$ where h' is a homomorphism defined by $h'(a) = a$ if $a \in S - \{b\}$ and $h'(a) = \lambda$ otherwise. ■

We see that the most simple form of erasing homomorphism, i.e. all letters which are not mapped to λ are mapped to themselves, adds tremendously to the generating power of PDlL systems.

We summarize the more important results of sections 5 and 6 in Fig. 4. Connection by a solid line means that the upper language family strictly contains the lower one; connection by a dotted line means that the upper language family contains the lower one and it is not known yet whether the inclusion is strict; if two language families are not connected at all this means that their intersection is nonempty but neither contains the other, i.e. they are <u>incomparable.</u>

We denote the closure of L(PDlL) under homomorphisms which map a letter either to itself or to λ, by $h_w L$(PDlL). (These homomorphisms are a restricted type of weak codings.)

1. $L(RE) = E(D2L) = h_{1:1}E(DlL) = h_w(PDlL)$: Theorems 5, 6, 13.
2. $E(DlL) \subsetneq L(RE)$ and $E(DlL)$ incomparable with $L(CS)$, $L(DLBA)$, $L(CF)$ and $L(REG)$: Theorem 8.
3. $E(DlL)$ incomparable with $h_{1:1}E(PDlL)$ and $h_\lambda E(PDlL)$.

   This needs a brief explanation. Let $L = \{a, aa\} \cup \{b\}\{c\}^*\{b\}$.

$L \in h_{1:1}E(PD1L) \subseteq h_\lambda E(PD1L)$ by the proof of Theorem 12 (i), and $L \notin E(D1L)$ by the proof of Theorem 8. Therefore

(a) $h_{1:1}E(PD1L) \not\subseteq E(D1L)$ and $h_\lambda E(PD1L) \not\subseteq E(D1L)$.

Since $E(D1L)$ contains languages in $L(RE) - L(CS)$ by Theorem 8 (ii) and $h_{1:1}E(PD1L) \subseteq h_\lambda E(PD1L) \subsetneq L(CS)$ by Theorem 12 (i) we have

(b) $E(D1L) \not\subseteq h_{1:1}E(PD1L)$ and $E(D1L) \not\subseteq h_\lambda E(PD1L)$.

Furthermore, by definition:

(c) $E(PD1L) \subseteq E(D1L)$ and $E(PD1L) \subseteq h_{1:1}E(PD1L) \subseteq h_\lambda E(PD1L)$.

From (a) (b) and (c) it follows that $E(D1L)$ is incomparable with both $h_{1:1}E(PD1L)$ and $h_\lambda E(PD1L)$.

4. $L(CS) = E(P2L)$: van Dalen [3].

5. $L(DLBA) = E(PD2L) = h_\lambda E(PDIL)$: Theorems 10 and 9.

6. $E(PD1L) \subsetneq h_{1:1}E(PD1L) \subseteq h_\lambda E(PD1L) \subsetneq L(DLBA)$: Theorem 12(i).

7. $E(PD1L) \subseteq E(D1L)$ by definition. Strict inclusion since

$E(PD1L) \subsetneq L(CS)$ by Theorem 12 (i) and

$E(D1L) \cap (L(RE) - L(CS)) \neq \emptyset$ by Theorem 8 (ii).

8. $E(PD1L)$ is incomparable with both $L(CF)$ and $L(REG)$ by Theorem 8 (i).

9. (a) $L(REG) \notin h_\lambda E(PD1L)$ by Theorem 11.

(b) $E(PD1L) \subsetneq h_{1:1}E(PD1L) \subseteq h_\lambda E(PD1L)$ by Theorem 12 (i)

(c) $E(PD1L)$ is incomparable with $L(REG)$ and $L(CF)$ by Theorem 8 (i).

From (a), (b) and (c) follows that both $h_{1:1}E(PD1L)$ and $h_\lambda E(PD1L)$ are incomparable with $L(REG)$ and $L(CF)$ respectively.

Acknowledgement. I thank Professor A. Salomaa for constructive criticism on an early draft of this paper.

REFERENCES

1. R. Baker and G. T. Herman, Simulation of organisms using a developmental model, I: Basic description; II: The heterocyst formation problem in blue-green algae, Int. J. Bio-Med. Comput., 3 (1972), 201 and 251.

2. R. Balzer,An 8 state minimal solution to the firing squad synchronization problem, Inf. Contr.10 (1967), 22-42.

3. D. van Dalen, A note on some Systems of Lindenmayer, Math Systems Theory 5 (1971), 128-140.

4. A. Ehrenfeucht and G. Rozenberg, The equality of E0L languages and codings of 0L languages, Int. J. of Comp. Math., (to appear).

5. A. Ehrenfeucht and G. Rozenberg, Trade off between the use of nonterminals. codings and homomorphisms in defining languages for some classes of rewriting systems, In: Automata, Languages and Programming,(J. Loeckx ed.), Lecture Notes in Comp. Sci. Vol. 14, Springer, (1974), 473-480.

6. D. Frijters and A. Lindenmayer, A model for the growth and flowering of Aster-Angliae on the basis of table (1, 0) L systems. In: L Systems (G. Rozenberg and A. Salomaa, eds.), Lecture Notes in Comp. Sci. Vol. 15, Springer, (1974).

7. G. T. Herman and W. H. Liu, The daughter of CELIA, the french flag and the firing squad, Simulation 21 (1973), 33.

8. G. T. Herman, W. H. Liu, S. Rowland and A. Walker, Synchronization of growing cellular arrays, Inf. Contr. (to appear).

9. G. T. Herman and G. Rozenberg, Developmental Systems and Languages, North Holland, Amsterdam (1974), to appear.

10. G. T. Herman and G. L. Schiff, Simulation of organisms based on L systems, SUNY at Buffalo, Dept. Comp. Sci. Tech. Rept. #75 (1974).

11. J. E. Hopcroft and J. D. Ullman, Formal Languages and their Relation to Automata, Addison-Wesley (1969).

12. A. Lindenmayer, Mathematical models for cellular interactions in development I and II, J. Theoret. Biol. 18 (1968), 280-315.

13. M. Minsky, Computation: Finite and Infinite Machines, Prentice Hall (1967).

14. M. Nielsen, G. Rozenberg, A. Salomaa and S. Skyum, Nonterminals, homomorphisms and codings in different variations of 0L systems, Univ. of Aarhus, Dept. Comp. Sci. Tech. Rept. PB-21 (1974).

15. M. O. Rabin and H. Wang, Words in the history of a Turing machine with fixed input, J. ACM 10 (1963), 526-527.

16. H. Rodgers, Theory of Recursive Functions and Effective Computability, McGraw-Hill (1967).

17. G. Rozenberg, L systems with interactions: the hierarchy, SUNY at Buffalo, Dept. Comp. Sci. Tech. Rept. # 28 (1972).

18. G. Rozenberg and K. P. Lee, Some properties of the class of L languages with interactions, SUNY at Buffalo, Dept. Comp. Sci. Tech. Rept. #43 (1972).

19. A. Salomaa, Formal Languages, Academic Press (1973).

20. A. Salomaa, On sentential forms of context free grammars, Acta Informatica 2 (1973), 40-49.

21. P. Vitányi and A. Walker, Stable string languages of Lindenmayer systems, (1974) (submitted to a technical journal.)

FOOTNOTES

1) See e.g. Minsky [13] for terminology and results on Turing machines.

2) A family of languages is said to be closed under k-limited erasing if, for any language L of the class and any homomorphism h with the property that h never maps more than k consecutive symbols of any sentence x in L to λ, h(L) is in the class. We shall furthermore be concerned with nonerasing homomorphisms, i.e. homomorphisms which map no letter to the empty word λ; letter to letter homomorphisms (also called codings), i.e. homomorphisms which map letters to letters, and homomorphisms which map a letter either to itself or to the empty word λ. (These homomorphisms are a subclass of the weak codings where a letter is mapped either to a letter or to λ.) For further details concerning homomorphisms and other operations on languages and closure under these operations see [11] or [19].

3) For the working space theorem see [19, p. 93]. The working space theorem is a variant of the linear bounded automaton theorem which tells that the family of languages accepted by linear bounded automata is equal to $\mathcal{L}$(CS). For a definition of linear bounded automata see section 6, [11] or [19]. For the uvwxy lemma (or Bar Hillel's lemma) see [19, p. 56].

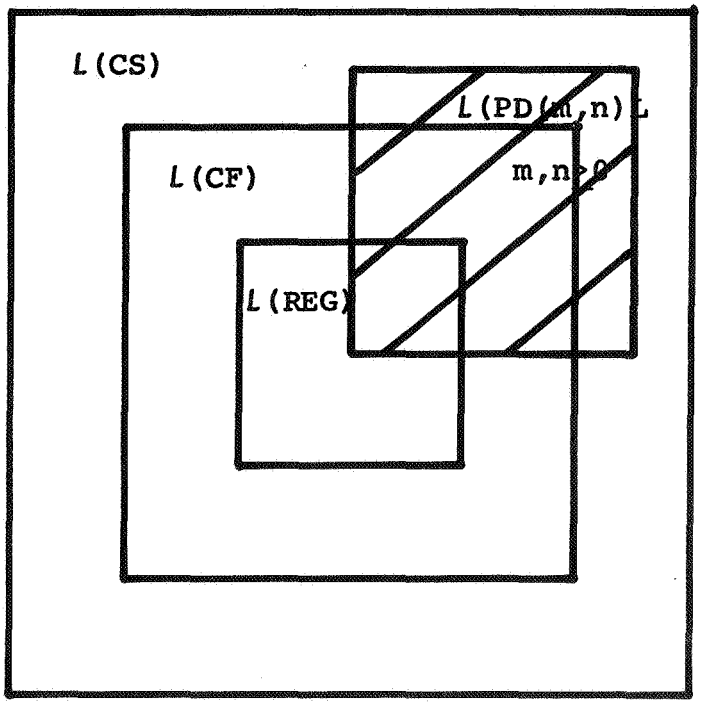

Figure 1

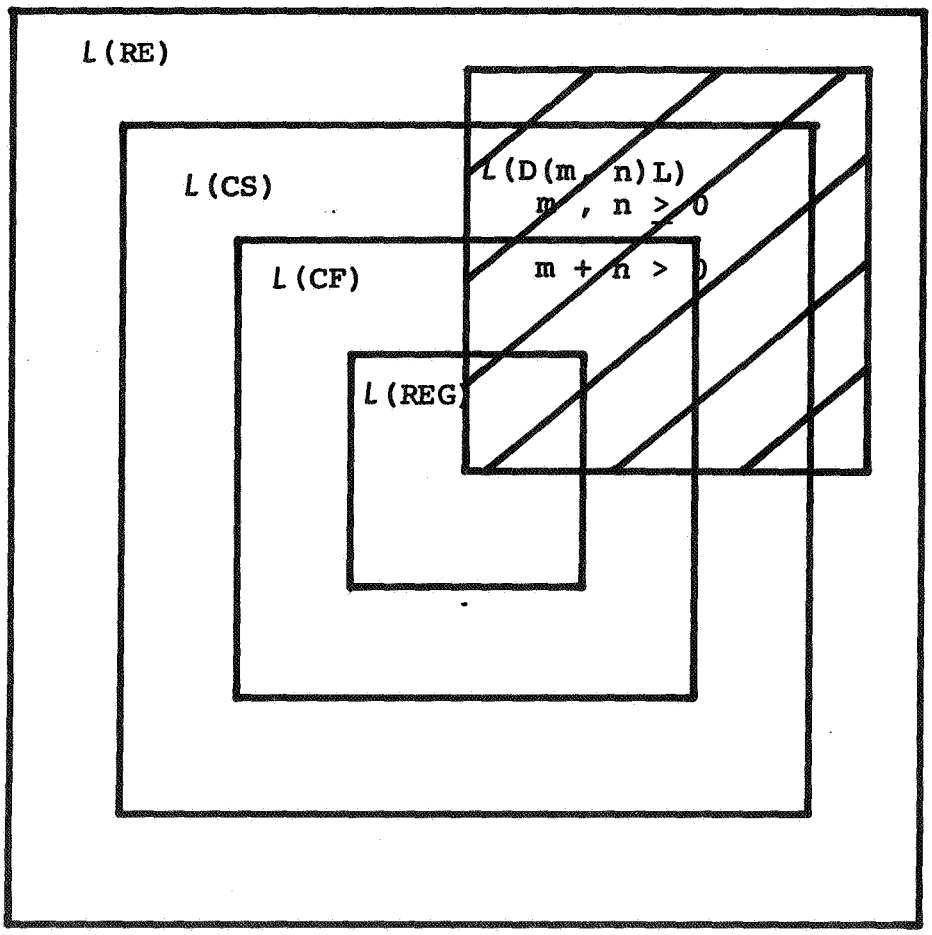

Figure 2

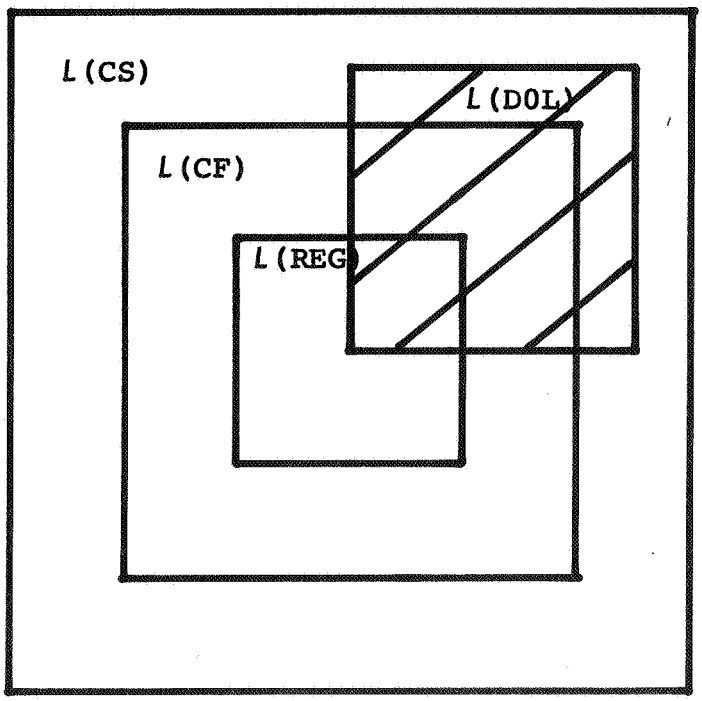

Figure 3

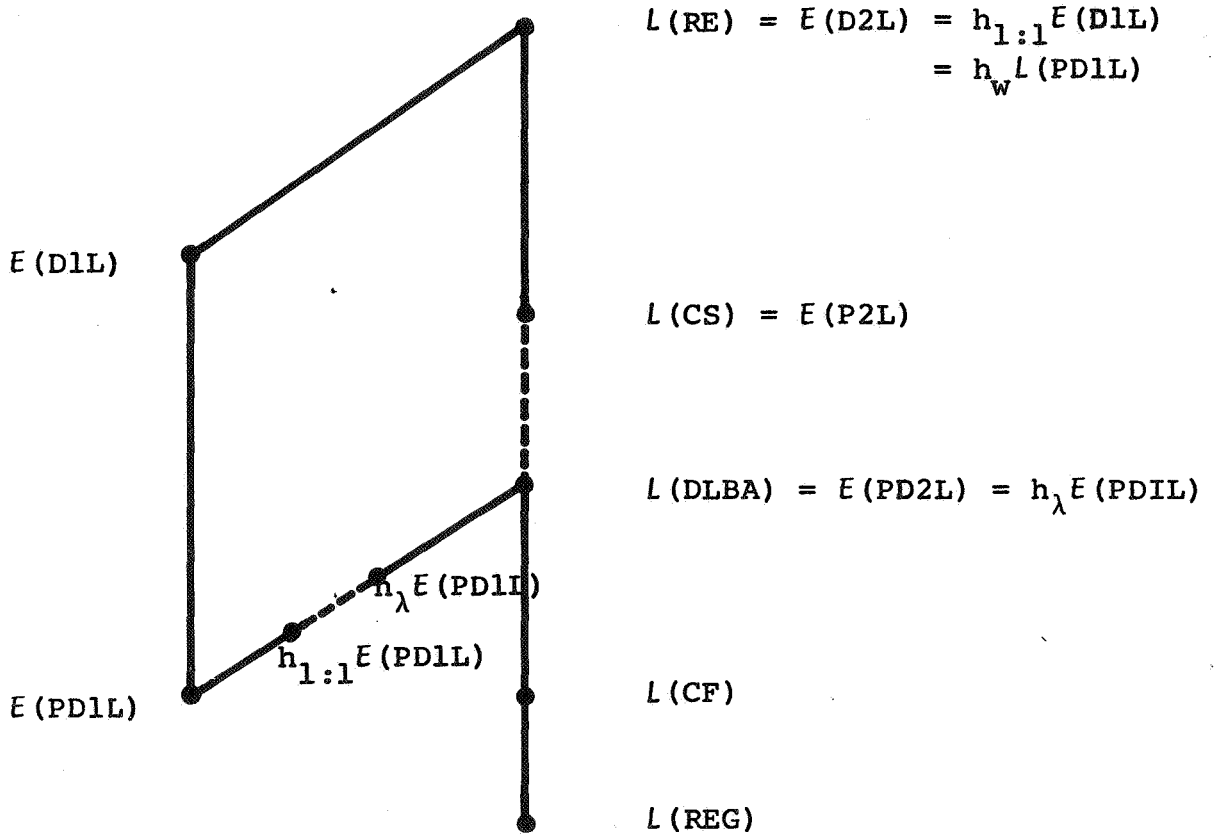
L(RE) = E(D2L) = $h_{1:1}$E(D1L)
= h_wL(PD1L)
E(D1L)
L(CS) = E(P2L)
L(DLBA) = E(PD2L) = h_λE(PDIL)
h_λE(PD1L)
$h_{1:1}$E(PD1L)
E(PD1L)
L(CF)
L(REG)

Figure 4