GROWTH FUNCTIONS

GROWTH OF STRINGS IN CONTEXT DEPENDENT LINDENMAYER SYSTEMS *)

PAUL M.B. VITÁNYI

Mathematisch Centrum, Amsterdam, The Netherlands

ABSTRACT

Growth functions of context dependent Lindenmayer systems are investigated. Bounds on the fastest and slowest growth in such systems are derived, and a method to obtain (P)D1L growth functions from (P)D2L growth functions is given. Closure of context dependent growth functions under several operations is studied with special emphasis on an application of the firing squad synchronization problem. It is shown that, although all growth functions of DILs using a one letter alphabet are DOL growth functions, there are growth functions of PDILs using a two letter alphabet which are not. Several open problems concerning the decidability of growth equivalence, growth type classification etc. of context dependent growth are shown to be undecidable. As a byproduct we obtain that the language equivalence of PDILs is undecidable and that a problem proposed by Varshavsky has a negative solution.

1. INTRODUCTION

Lindenmayer systems, L systems for short, are a class of parallel rewriting systems. They were introduced by Lindenmayer [59,60] as a model for the developmental growth in filamentous organisms. These systems have been extensively studied, see e.g. Herman & Rozenberg [45], and, from the formal language point of view, form an alternative to the usual generative grammar approach. A particularly interesting topic in this field, both from the viewpoint of the biological origins and in its own right, is the study of the growth of the length of a filament as a function of time. An L system consists of an initial string of letters, symbolizing an initial one di-

*) This paper is registered at the Mathematical Center as IW 19/74.

mensional 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.* When each letter can be rewritten in exactly one way in each context of m letters to its left and n letters to its right we talk about a *deterministic (m,n)L system.* All L systems considered in this paper are deterministic (i.e. essentially monogenic rewriting systems) since this allows a cleaner theory of growth to be developed. However, most of the results concerning growth types and decidability we shall derive hold under appropriate interpretation also for nondeterministic L systems.

The general family of deterministic L systems is called the *family of deterministic context dependent L systems* or *DIL systems.* The best investigated subfamily is that of the D(0,0)L (i.e. DOL) or deterministic context independent L systems. Growth of the length of strings in this latter class has been extensively studied, cf. section 2, and almost all questions posed have been proved to be decidable by algebraic means [111,75,98] and some by combinatorial arguments [116]. The study of the growth of length of strings in the general case of context dependent L systems has been more or less restricted to the observation that the corresponding problems here are still open, cf. [45, chapter 15], [75] and [102]. We shall investigate the growth of length of strings in context dependent L systems and we shall solve some of the open problems by quite elementary means. By a reduction to the printing problem for Turing machines we are able to show that e.g. the growth type of a context dependent L system is undecidable, even if no production is allowed to derive the empty word; that the growth equivalence problem for these systems is unsolvable; and that the corresponding questions for the growth ranges have similar answers. (As a byproduct we obtain the results that the language equivalence for PD1Ls is undecidable and that a problem proposed by Varshavsky has a negative solution.)

Furthermore, we derive bounds on the fastest and slowest growth in such systems; we give a method for obtaining growth functions of systems with a smaller context from systems with a larger context; it is shown that all bounded growth functions of context dependent L systems are within the realm of the context independent growth functions whereas for each type of unbounded context dependent growth functions there are growth functions which are not; similarly, all growth functions of context dependent L systems using a one letter alphabet are growth functions of context independent L systems whereas this is not the case for growth functions of the simplest context dependent L systems using a two letter alphabet; we give an application of the firing squad synchronization problem, etc.

The paper is divided in three parts. In section 2 we prepare the ground by giving a cursory review of some results on growth functions of context independent L systems. In sections 3.1-3.3 we develop outlines for a theory of context dependent growth functions and give some theorems and illuminating examples. In section 3.4

we prove the undecidability of several open problems in this area.

2. GROWTH FUNCTIONS OF CONTEXT INDEPENDENT L SYSTEMS

We assume that the usual terminology of formal language theory is familiar. Except where defined otherwise we shall customarily use, with or without indices, i,j, k,m,n,p,r,t to range over the set of natural numbers $\mathbb{N} = \{0,1,2,...\}$; a,b,c,d,e to range over an alphabet W; v,w,z to range over W^* i.e. the set of all words over W including the *empty word* λ. #Z denotes the *cardinality* of a set Z; lg(z) the *length* of a word z and $lg(\lambda) = 0$.

An L system is called deterministic context independent (DOL system) if the rewriting rules are deterministic and the rewriting of a letter is independent of the context in which it occurs. With each DOL system G we can associate a growth function f_G where $f_G(t)$ is the length of the generated string at time t. Growth functions of DOL systems were studied first by Szilard [111], later by Doucet [15], Paz & Salomaa [75], Salomaa [98] and Vitányi [116,115].

A *semi DOL system (semi DOL)* is an ordered pair $S = \langle W,\delta\rangle$ where W is a finite nonempty *alphabet* and δ is a total mapping from W into W^* called the *set of production rules*. A pair $(a,\delta(a))$ is also written as $a \to \delta(a)$. We extend δ to a homomorphism on W^* by defining $\delta(\lambda) = \lambda$ and $\delta(a_1a_2...a_n) = \delta(a_1)\delta(a_2)...\delta(a_n)$, $n > 0$. δ^i is the composition of i copies of δ and is inductively defined by $\delta^0(v) = v$ and $\delta^i(v) = \delta(\delta^{i-1}(v))$ for $i > 0$. A *DOL system (DOL)* is a triple $G = \langle W,\delta,w\rangle$ where W and δ are as above and $w \in WW^*$ is the *axiom*. The *DOL language* generated by G is $L(G) = \{\delta^i(w) \mid i \geq 0\}$. The *growth function* of G is defined by $f_G(t) = lg(\delta^t(w))$. Clearly, for each DOL $G = \langle W,\delta,w\rangle$, if $m = \max\{lg(\delta(a)) \mid a \in W\}$ then $f_G(t) \leq lg(w)\, m^t$. Hence the fastest growth possible is exponentially bounded. We classify the growth of DOLs as follows [116]:

A growth function f_G is *exponential (type 3)* if $\lim_{t\to\infty} f_G(t)/x^t \geq 1$ for some $x > 1$; f_G is *polynomial (type 2)* if $\lim_{t\to\infty} f_G(t)/p(t) \geq 1$ and $\lim_{t\to\infty} f_G(t)/q(t) \leq 1$ for some unbounded polynomials[1] p and q; f_G is *limited (type 1)* if $0 < f_G(t) \leq m$ for some constant m and all t; f_G is *terminating (type 0)* if $f_G(t) = 0$ but for a finite number of initial arguments.

By an application of the theory of homogeneous linear difference equations with constant coefficients, Salomaa [98] gave an algorithm to derive an explicit formula of the following form for the growth function of an arbitrary DOL G:

$$(1) \qquad f_G(t) = \sum_{i=1}^{n} p_i(t)\, c_i^t,$$

[1] A function f(t) is said to be *unbounded* if for each n_0 there is a t_0 such that $f(t) > n_0$ for all $t > t_0$.

where p_i is an r_i-th degree polynomial with complex algebraic coefficients and c_i a complex algebraic constant, $1 \le i \le n$, $\Sigma_{i=1}^{n}(r_i+1) = \#W$. From this it follows that the above classification is exhaustive in the DOL case; that the growth type of a DOL can be determined and that the growth equivalence for two DOLs is decidable (two DOLs G, G' are said to be growth equivalent iff $f_G(t) = f_{G'}(t)$ for all t).

The approach of [98] becomes too complicated for large alphabets and does not tell us anything about the *structure* of growth, viz. the local properties of production rules which are responsible for types of growth [116]. By considering DOLs with one letter axioms we can talk about growth types of letters, and clearly the growth type of a DOL is the highest numbered growth type of the letters in its axiom. Given a semi DOL, different types of growth may result from different choices of the axioms; therefore the growth type of a semi DOL is a combination of the growth types possible for different choices of the axiom. (Written from left to right according to decreasing digits, e.g. 3210, 321, 21.)

Example 1.

$G = \langle\{a\},\{a\to a^2\},a\rangle$: growth type 3.

$G = \langle\{a,b\},\{a\to b, b\to ab\},a\rangle \quad f_G(t) = \frac{\sqrt{5}+1}{2\sqrt{5}}\left(\frac{1+\sqrt{5}}{2}\right)^t + \frac{\sqrt{5}-1}{2\sqrt{5}}\left(\frac{1-\sqrt{5}}{2}\right)^t$: growth type 3.

($f_G(t)$ is the t-th term of the Fibonacci sequence)

$G = \langle\{a,b\},\{a\to ab, b\to b\},a\rangle \quad f_G(t) = t+1$: growth type 2.

$S = \langle\{a,b,c,d\},\{a\to a^2b, b\to bc, c\to cd, d\to\lambda\}\rangle$: growth type 3210.

The following theorem, [116], tells us which combinations may occur in the DOL case.

Theorem 1. Type 2 never occurs without type 1. All other combinations are possible. (I.e. there are no semi DOLs of growth type 320, 32, 20 or 2).

It is, however, easy to show that growth type 2 may occur without growth type 1 for the simplest context dependent L systems, i.e. the one letter alphabet PD1Ls. (cf. example 2, section 3).

Furthermore, in the DOL case, necessary and sufficient conditions for the growth type of a letter $a \in W$ under a set of production rules are obtained from the emptyness of the intersection of the set of letters, derivable from a, with three disjoint classes of recursive letters, where a letter $b \in W$ is *recursive* if $\delta^i(b) = v_1bv_2$ for some $i > 0$ and some $v_1,v_2 \in W^*$, [116].

3. GROWTH FUNCTIONS OF CONTEXT DEPENDENT L SYSTEMS

The general form of a context dependent L system was introduced by Rozenberg [86]. We define a *deterministic (m,n)L system (D(m,n)L)* as a triple $G = \langle W,\delta,w\rangle$, where W is a finite nonempty *alphabet*, the *set of production rules* δ is a total mapping from $\bigcup_{i=0}^{m} W^i \times W \times \bigcup_{j=0}^{n} W^j$ into W^* and $w \in WW^*$ is the *axiom*. δ induces a total mapping $\bar{\delta}$ from W^* into W^* as follows: $\bar{\delta}(\lambda) = \lambda$ and $\bar{\delta}(a_1a_2\ldots a_k) = \alpha_1\alpha_2\ldots\alpha_k$ if for each i such that $1 \le i \le k$ we have

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

where we take $a_j = \lambda$ for $j < 1$ and $j > k$. The composition of i copies of $\bar{\delta}$ is inductively defined by $\bar{\delta}^0(v) = v$ and $\bar{\delta}^i(v) = \bar{\delta}(\bar{\delta}^{i-1}(v))$, $i > 0$. When no confusion can result we shall write δ for $\bar{\delta}$. The *D(m,n)L language* generated by G is $L(G) = \{\delta^i(w) \mid i \ge 0\}$, and the *growth function* of G is $f_G(t) = \lg(\delta^t(w))$.

A *semi D(m,n)L* is a D(m,n)L without the axiom. A *propagating D(m,n)L (PD(m,n)L)* is a D(m,n)L $G = \langle W,\delta,w\rangle$ such that $\delta(v) \neq \lambda$ for all $v \in WW^*$. In the literature a D(0,0)L is usually called a DOL, a D(1,0)L or D(0,1)L is usually called a D1L, a D(1,1)L is usually called a D2L and a D(m,n)L $(m,n\ge 0)$ a DIL. The corresponding semi L systems are named accordingly.

Example 2. $S = \langle W,\delta\rangle$ is a semi PD(0,1)L where $W = \{a\}$ and $\delta(\lambda,a,\lambda) = a^2$, $\delta(\lambda,a,a) = a$. It is easily verified that for every axiom a^k, $k > 0$, S yields the growth function $f(t) = k+t$. (At each time step the letter on the right end of the string generates aa while the remaining letters generate a.) Therefore, even for PD1Ls using a one letter alphabet growth type 2 can occur without growth type 1 and all combinations of growth types 0,1,2,3 are possible. (Contrast this with the situation for DOLs in theorem 1.)

In section 2 we defined growth types 3,2,1,0 which were exhaustive for the DOL case. However, as will appear in the sequel, this is not so for DILs. Therefore we define two additional growth types to fill the gaps between types 1 and 2, and types 2 and 3. We call the growth in a DIL G *subexponential (type 2½)* iff the growth is not exponential and there is no unbounded polynomial p such that $f_G(t) \le p(t)$ for all t; *subpolynomial (type 1½)* iff f_G is unbounded and for each unbounded polynomial p holds that $\lim_{t\to\infty} f_G(t)/p(t) = 0$.

For DOLs the following types of problems have been considered and solved effectively (cf. section 2 and the references contained therein).

(i) Analysis problem. Given a DIL, describe its growth function in some fixed predetermined formalism.

(ii) Synthesis problem. Given a function f in some fixed predetermined formalism and some restriction x on the family of DILs. Find a DIL which satisfies x and whose growth function is f. Related to this is the problem of which growth functions can be growth functions of DILs satisfying restriction x.

(iii) Growth equivalence problem. Given two DILs, decide whether or not they have the same growth function.

(iv) Classification problems. Given a DIL or a semi DIL, decide what is its growth type.

(v) Structural problems. What properties of production rules induce what types of growth?

Furthermore we have the hierarchy problem. Is the set of DOL growth functions a proper subset of the D1L growth functions and similar problems?

In section 3.4 we shall show that even for PD1Ls the problems (i)-(v) are recursively unsolvable.

3.1. Bounds on unbounded growth

Since it is difficult to derive explicit formulas for growth functions of the more involved examples of DILs, and according to section 3.4 impossible in general, we avail ourselves of the following notational devices.

$\lfloor f(t) \rfloor$ is the *lower entier* of f(t), i.e. for each t, $\lfloor f(t) \rfloor$ is the largest integer not greater than f(t).

$f(t) \sim g(t)$: f(t) is *asymptotic* to g(t), i.e. $\lim_{t\to\infty} f(t)/g(t) = 1$.

$f(t) \approx\!> g(t)$: f(t) *slides onto* g(t) (terminology provided by G. Rozenberg) iff for each maximum argument interval [t',t"] on which g(t) has a constant value holds that $f(t) = g(t)$ for all t and some t''' such that $t' \le t''' \le t \le t''$.

As in the DOL case, for each DIL $G = \langle W,\delta,w\rangle$ holds that $f_G(t) \le lg(w)\, m^t$ where $m = \max\{lg(\delta(v_1,a,v_2)) \mid v_1,v_2 \in W^* \text{ and } a \in W\}$. Hence the fastest growth is exponential, and for each DIL there is a DOL which grows faster. We shall now investigate what is the slowest unbounded growth which can occur. Remember that a function f is unbounded if for each n_0 there is a t_0 such that $f(t) > n_0$ for $t > t_0$.

Theorem 2.

(i) For any PDIL $G = \langle W,\delta,w\rangle$ such that f_G is unbounded holds:

$$\lim_{t\to\infty} f_G(t)/\log_r t \ge 1 \qquad \text{where } r = \#W > 1.$$

(ii) For any DIL $G = \langle W,\delta,w\rangle$ such that f_G is unbounded holds:

$$\lim_{t\to\infty} \sum_{i=0}^{t} f(t) / \sum_{i=0}^{t} \lfloor \log_r((r-1)i+r) \rfloor \ge 1 \qquad \text{where } r = \#W > 1.$$

Proof.

(i) Order all strings in WW^* according to increasing length. The number of strings of length less than k is given by $t = \Sigma_{i=1}^{k-1} r^i$, $i = \#W$. Hence $t = \frac{r^k - r}{r-1}$ and therefore $k = \log_r((r-1)t+r)$. If we define f(t) as the length of the t-th string in WW^* then, clearly, $f(t) = \lfloor \log_r((r-1)t+r) \rfloor$ and $\lim_{t\to\infty} f(t)/\log_r t = 1$. The most any PDIL system with an unbounded growth function can do is to generate all strings of WW^* in order of increasing length and without repetitions. Therefore $\lim_{t\to\infty} f_G(t)/\log_r t \geq 1$.

(ii) The most any DIL with an unbounded growth function can do is to generate all strings of WW^* in some order and without repetitions. Therefore, $\lim_{t\to\infty} \Sigma_{i=0}^{t} f_G(t)/\Sigma_{i=0}^{t} f(i) \geq 1$. $\square$

In the sequel of this section we shall show that theorem 2 is optimal.

Example 3. Let $G_1 = \langle W,\delta,w \rangle$ be a PD(0,1)L such that $W = \{0,1,2,\ldots,r-1,¢,s\}$ $(r>1)$; $\delta(\lambda,¢,i) = ¢$ for $0 \leq i \leq r-1$, $\delta(\lambda,¢,s) = ¢0$, $\delta(\lambda,i,\lambda) = \delta(\lambda,i,s) = i+1$ for $0 \leq i < r-1$, $\delta(\lambda,s,\lambda) = 1$, $\delta(\lambda,s,0) = \delta(\lambda,s,1) = 0$, $\delta(\lambda,r-1,\lambda) = \delta(\lambda,r-1,s) = s$, $\delta(\lambda,i,j) = i$ for $0 \leq i,j \leq r-1$; $w = ¢0$.

The starting sequence is: $¢0$, $¢1$, ..., $¢\,r-1$, $¢s$, $¢01$, ..., $¢0\,r-1$, $¢0s$, $¢11$, ..., $¢\underbrace{r-1\ldots r-1}_{k\ \times}$, $¢\underbrace{r-1\ldots r-1}_{k-1\ \times}s$, $¢\underbrace{r-1\ldots r-1}_{k-2\ \times}s1$, ..., $¢s00\ldots$, $¢0000\ldots$, ...

Observe that G counts all strings over an alphabet of r letters. When an increment of the length k is due on the left side it needs k extra steps. Furthermore, there is an additional letter ¢ on the left. Therefore,

$$f_{G_1}(t) = \lfloor \log_r((r-1)t+r - \lfloor \log_r((r-1)t/r+1) \rfloor) \rfloor + 1$$
$$\approx \lfloor \log_r((r-1)t+r) \rfloor + 1.$$

Hence $f_{G_1}(t) \sim \log_r t$. Hence, with a PD1l using r+2 letters we can reach the slowest unbounded growth of a PDIL using r letters.

Some variations of Example 3 are the following:

Example 4. Let G_2 be a PD(0,1)L defined as G_1 but with $\delta(\lambda,¢,s) = ¢1$. Then, essentially, G_2 counts on a number base r and

$$f_{G_2}(t) = 2, \qquad 0 \leq t < r$$
$$f_{G_2}(t) = \lfloor \log_r(t - \lfloor \log_r t/r \rfloor) \rfloor + 2$$
$$\approx \lfloor \log_r t \rfloor + 2, \qquad t \geq r.$$

Example 5. Let $G_3 = <\{0,1,2,\ldots,r-1\} \times \{0,¢,s\}, \delta_3, (0,¢)>$ be such that the action is as in G_1 but with ¢ and s coded in the appropriate letters. Then,

$$f_{G_3}(t) = \lfloor \log_r((r-1)t+r - \lfloor \log_r((r-1)t/r+1)\rfloor)\rfloor$$
$$\approx> \lfloor \log_r((r-1)t+r)\rfloor$$

Example 6. Let G_4 be as G_2 with the modifications of G_3. Then

$$f_{G_4}(t) = 1, \qquad 0 \le t < r$$
$$f_{G_4}(t) = \lfloor \log_r(t-\lfloor \log_r t/r\rfloor)\rfloor + 1$$
$$\approx> \lfloor \log_r t\rfloor + 1, \qquad t \ge r.$$

Examples 3-6 all corroborate the fact that for any PDIL with an unbounded growth function there is a PD1L with an unbounded growth function which grows slower, although not slower than logarithmic. That theorem 2 (ii) cannot be improved upon follows from the following lemma, implicit in van Dalen [12] and Herman [33].

Lemma 1. For a suitable standard formulation of Turing machines [2], e.g. the quintuple version, holds that for any deterministic Turing machine T with symbol set S and state set Q we can effectively construct a D2L $G_5 = <W_5,\delta_5,w_5>$ which simulates it in real time. I.e. the t-th instantaneous description of T is equal to $\delta_5^t(w_5)$. There is a required G_5 with $W_5 = S \cup Q$ and a required propagating G_5 with $W_5 = Q \cup (S \times Q)$.

Since T can expand its tape with at most one tape square per move we see that $f_{G_5}(t+1) \le f_{G_5}(t)+1$.

It is well known that a Turing machine can compute every recursively enumerable set $A = \{1^{f(t)} \mid f(t)$ is a 1:1 total recursive function$\}$. We can do this in such a way that for each t when f(t) has been computed the Turing machine erases everything else on its tape. Subsequently, it recovers t from f(t) by f^{-1}, adds 1 and computes f(t+1). In particular, the simulating D2L G_5 can, instead of replacing all symbols except the representation of f(t) by blank symbols, replace all the superfluous blank letters by the empty word λ. Suppose that A is nonrecursive. Then, clearly, it is not the case that for each n_0 we can find a t_0 such that $f_{G_5}(t) > n_0$ for $t > t_0$, although such a t_0 exists for each n_0. Hence theorem 2 (ii) is optimal for D2Ls, and as will appear from the next lemma also for D1Ls.

[2] For results and terminology concerning these devices see e.g. M. Minsky, *Computation: finite and infinite machines*. Prentice-Hall, London (1967).

Lemma 2.

(i) Let $G = \langle W,\delta,w\rangle$ be any D2L. We can effectively find a D1L $G' = \langle W',\delta',w'\rangle$ such that for all t holds: $\delta'^{2t}(w') = ¢\delta^t(w)$ for some $¢ \notin W$.

(ii) Let $G = \langle W,\delta,w\rangle$ be any PD2L. We can effectively find a PD1L $G'' = \langle W'',\delta'',w''\rangle$ such that for all t holds: $\delta''^{2t}(w'') = ¢\delta^t(w)\$^t$ for some $¢,\$ \notin W$.

Proof.

(i) Let $G = \langle W,\delta,w\rangle$ be any D2L. Define a D(0,1)L $G' = \langle W',\delta',w'\rangle$ as follows:

$$W' = W \cup (W \times (W\cup\{\lambda\})) \cup \{¢\}, \quad ¢ \notin W; \quad w' = ¢w;$$
$$\delta'(\lambda,a,c) = (a,c), \qquad \delta'(\lambda,¢,(a,c)) = ¢\delta(\lambda,a,c),$$
$$\delta'(\lambda,¢,c) = ¢, \qquad \delta'(\lambda,(a,\lambda),\lambda) = \lambda,$$
$$\delta'(\lambda,(a,b),(b,c)) = \delta(a,b,c),$$

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'.)

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

$$\begin{aligned}\delta'^2(¢a_1a_2\ldots a_k) &= \delta'(¢(a_1,a_2)(a_2,a_3)\ldots(a_k,\lambda)) = \\ &= ¢\delta(\lambda,a_1,a_2)\delta(a_1,a_2,a_3)\ldots\delta(a_{k-1},a_k,\lambda) = \\ &= ¢\delta(a_1a_2\ldots a_k).\end{aligned}$$

Since, furthermore, $\delta'^2(¢) = ¢$ we have therefore $\delta'^{2t}(w') = ¢\delta^t(w)$ for all t.

(ii) Let $G = \langle W,\delta,w\rangle$ be any PD2L. Define a PD1L $G'' = \langle W'',\delta'',w''\rangle$ as follows:

$$W'' = W \cup (W \times (W\cup\{\lambda\})) \cup \{¢,\$\}, \quad ¢,\$ \notin W; \quad w'' = ¢w;$$
$$\delta''(\lambda,a,c) = (a,c), \qquad \delta''(\lambda,(a,b),(b,c)) = \delta(a,b,c),$$
$$\delta''(\lambda,a,\$) = (a,\lambda), \qquad \delta''(\lambda,¢,(a,c)) = ¢\delta(\lambda,a,c),$$
$$\delta''(\lambda,\$,\$) = \$, \qquad \delta''(\lambda,(a,\lambda),\lambda) = \delta''(\lambda,(a,\lambda),\$) = \$,$$
$$\delta''(\lambda,¢,d) = ¢, \qquad \delta''(\lambda,\$,\lambda) = \$,$$

for all $a,b, \in W$, all $c \in W\cup\{\lambda\}$ and all $d \in W\cup\{\lambda,\$\}$. Analogous with the above we prove that if $\delta^t(w) \neq \lambda$ for all t then $\delta''^{2t}(w'') = ¢\delta^t(w)\$^t$. □

Theorem 3.

(i) If f(t) is a D2L growth function then $g(t) = f(\lfloor t/2\rfloor)+1$ is a D1L growth function.

(ii) If f(t) is a PD2L growth function then $g(t) = f(\lfloor t/2\rfloor)+\lfloor t/2\rfloor+1$ is a PD1L growth function.

(iii) If f(t) is a PD2L growth function then $g(t) = f(\lfloor t/2\rfloor)$ is a D1L growth function.

(iv) If f(t) is a PD2L growth function then $g(t) = f(\lfloor t/2\rfloor)+\lfloor t/2\rfloor$ is a PD1L growth function.

Proof. (i) and (ii) follow from lemma 2 and its proof. (iii) and (iv) follow from lemma 2 and its proof by the observation that we can encode the left end marker ¢ in the leftmost letter of a string and keep it there in the propagating case. □

Note that by lemma 2 the transition in theorem 3 is effective, i.e. given a D2L G, of which f is the growth function, we can construct a required D1L G' such that $f_{G'} = g$.

3.2. Synthesis of growth functions

In the last section we saw that if f(t) is the growth function of a D2L G then $g(t) = f(\lfloor t/2 \rfloor)+1$ is the growth function of a D1L G' and there is a uniform method to construct G' given G. In this sense we shall treat some methods for obtaining growth functions. We consider operations under which families of growth functions are closed. An important tool here is an application of the *Firing Squad Synchronization Problem*[3]. Stated in the terminology of L systems it is the following. Let $S = \langle W_S, \delta_S \rangle$ be a semi PD2L such that $lg(\delta_S(a,b,c)) = 1$ for all $b \in W_S$ and all $a,c \in W_S \cup \{\lambda\}$, and there is a letter m in W_S such that $\delta_S(m,m,\lambda) = \delta_S(m,m,m) = m$. The problem is to design an S satisfying the restrictions above such that $\delta^{k(n)}(m^n) = f^n$, $f \in W_S$, for all natural numbers n and a minimal function k of n, while $\delta^t(m^n) \in (W_S - \{f\})^n$ for all t, $0 \leq t < k(n)$. Balzer[4] proved that there is a minimal time solution $k(n) = 2n-2$. In the PD2L case we can achieve a solution in e.g. $k(n) = n-1$ by dropping the restriction $\delta_S(m,m,\lambda) = m$ and having both letters m on the ends of an initial string act like "soldiers receiving the firing command from a general" in the firing squad terminology. Assume that $S = \langle W_S, \delta_S \rangle$ is such a semi PD2L simulating a firing squad with $k(n) = n-1$. Let $G = \langle W, \delta, w \rangle$ be any (P)D2L. We define the (P)D2L $G' = \langle W', \delta', w' \rangle$ as follows:

$$W' = W \times W_S;$$

$$w' = (a_1,m)(a_2,m)\dots(a_k,m) \qquad \text{for } w = a_1a_2\dots a_k,$$

$$\delta'((a,a'),(b,b'),(c,c')) = (b,b'') \qquad \text{for } \delta_S(a',b',c') = b'' \text{ and } a'b'c' \neq fff,$$

$$\delta'((a,f),(b,f),(c,f)) = \begin{cases} (b_1,m)(b_2,m)\dots(b_h,m) & \text{for } \delta(a,b,c) = b_1b_2\dots b_h, \\ \lambda & \text{for } \delta(a,b,c) = \lambda. \end{cases}$$

We easily see that if $\delta(v) = v'$ for $v,v' \in W^*$ then

---

[3] See e.g. Minsky, Op. cit., 28-29.

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

$$\delta'^{\lg(v)}((a_1,m)(a_2,m)\dots(a_k,m)) = (b_1,m)(b_2,m)\dots(b_h,m)$$

where $v = a_1a_2\dots a_k$ and $v' = b_1b_2\dots b_h$; and $\delta'^{\lg(v)}((a_1,m)(a_2,m)\dots(a_k,m)) = \lambda$ for $v' = \lambda$. Therefore we have:

Lemma 3. Let G be any (P)D2L. We can effectively find a (P)D2L G' such that

$$(2) \qquad f_{G'}(t) = \begin{cases} f_G(0) & \text{for all } t \text{ such that } 0 \le t < f_G(0), \\ f_G(\tau+1) & \text{for all } t \text{ such that } \sum_{i=0}^{\tau} f_G(i) \le t < \sum_{i=0}^{\tau+1} f_G(i). \end{cases}$$

Since we can simulate an arbitrary (but fixed) number of r firing squads in sequence plus a number j of production steps of G' for each production step of G, we can effectively find a (P)D2L G' for each (P)D2L G such that:

$$f_{G'}(t) = \begin{cases} f_G(0) & \text{for all } t \text{ such that } 0 \le t < r\, f_G(0) + j \\ f_G(\tau+1) & \text{for all } t \text{ such that } r\sum_{i=0}^{\tau} f_G(i)+(\tau+1)j \le t < \\ & \qquad r\sum_{i=0}^{\tau+1} f_G(i)+(\tau+2)j. \end{cases}$$

Let us call the operation to obtain a growth function $f_{G'}$ from f_G as defined in (2) *FSS*. Then $f_{G'} = FSS(f_G)$.

A cascade of r firing squads working inside each other, such that one production step of a (P)D2L G is simulated if the outermost squad fires, gives us a (P)D2L G' such that $f_{G'} = FSS^r(f_G)$, i.e.

$$(3) \qquad f_{G'}(t) = \begin{cases} f_G(0) & \text{for all } t \text{ such that } 0 \le t < f_G(0)^r \\ f_G(\tau+1) & \text{for all } t \text{ such that } \sum_{i=0}^{\tau} f_G(i)^r \le t < \sum_{i=0}^{\tau+1} f_G(i)^r. \end{cases}$$

Example 7. Suppose that f_G is exponential, say $f_G(t) = 2^t$. Then $FSS(f_G) = f$ where $f(t) = 2^{\tau+1}$ for $\Sigma_{i=0}^{\tau} 2^i \le t < \Sigma_{i=0}^{\tau+1} 2^i$. Hence $f(2^{\tau+1}-1) = 2^{\tau+1}$ and $f(t) = 2^{\lfloor \log_2 t \rfloor}$, i.e. $f(t) \asymp t$.[5] We can obtain analogous results for arbitrary exponential functions.

[5] $f \asymp g$ asserts that f is of the same *order of magnitude* as g, i.e. $c_1 g(t) < f(t) < c_2 g(t)$ for all t and some constants c_1, c_2.

Example 8. Suppose that f_G is polynomial, e.g. $f_G(t) = p(t)$ where $p(t)$ is a polynomial of degree r. Then $FSS(f_G) = f$ where $f(\Sigma_{i=0}^{t} p(i)) = p(t+1)$. Since $\Sigma_{i=0}^{t} p(i) = q(t)$ where $q(t)$ is a polynomial of degree r+1 we have $f(t) \asymp t^{r/r+1}$. By (3) we see that $FSS^j(f_G) = f$ where $f(t) \asymp t^{r/(r+j)}$.

Hence we have:

Theorem 4. For each rational number r, $0 < r \leq 1$, we can effectively find a PD2L G such that $f_G(t) \asymp t^r$.

Proof. Since $r = r'/r''$ such that r'', r' are natural numbers and $r'' \geq r'$, and according to Szilard [111] we can, for every monotonic ultimately polynomial function g, find a PDOL G' such that $f_{G'} = g$; by example 8 we can find a PD2L G such that $f_G(t) \asymp t^{r'/r''}$.
□

Example 9. Let $f_G(t) = \lfloor \log_2 t \rfloor$. Then $FSS(f_G) = f$, where $f((t-1)2^{t+1}+4) = t+1$, i.e. $f(t) \asymp \log t$.

Hence we see that the relative slowing down gets less when the growth function is slower.

By theorem 3 everything we have obtained for D2Ls holds for D1Ls if we substitute $\lfloor t/2 \rfloor$ for t in the expression for the growth function and add 1. However, even for D1Ls we can achieve a greater slowing down. Let G be some D2L. We can construct a D1L G' which simulates G such that for each production step of G, G' does the following.

(a) G' counts all strings of length $f_G(t)$ over an r letter alphabet by the method of example 3. When an increase of length is due on e.g. the left side,
(b) G' initializes a firing squad, making use of the simulation technique of lemma 2. When the firing squad fires, G' simulates one production step of G and subsequently starts again at (a).

Hence, if $h(t) \leq f_G(t) \leq g(t)$ for a D2L G and monotonic increasing functions h and g then we can effectively find a D1L G' such that $f_{G'}(\Sigma_{i=0}^{t} r^{h(i)}) < g(t+1)$. For instance, if $f_G(t) = t$ then $f_{G'}(t) < \log_r t$, $t > 1$.

We can combine processes like the above to obtain stranger and stranger, slower and slower growth functions. Similar to the above application of the Firing Squad Synchronization Problem we could apply the *French Flag Problem* (see e.g. [37]).

The next theorem tells us under what operations the family of growth functions is closed. In particular, the subfamilies of (P)D2L, (P)D1L and (P)DOL growth functions are closed under (i)-(iii).

Theorem 5. Growth functions are closed under (i) addition, (ii) multiplication with a natural number $r > 0$, (iii) entier division of the argument by a natural number $r > 0$, (iv) FSS. Growth functions are not closed under (v) subtraction, (vi) division, (vii) composition.

Proof.

(i) Let $G_1 = \langle W_1, \delta_1, w_1 \rangle$ and $G_2 = \langle W_2, \delta_2, w_2 \rangle$ be two DILs with disjoint alphabets. Define $G_3 = \langle W_1 \cup W_2, \delta_3, w_1 w_2 \rangle$. Then it is easy to construct δ_3, given δ_1 and δ_2, such that $f_{G_3} = f_{G_1} + f_{G_2}$.

(ii) Follows from (i).

(iii) Let $G_1 = \langle W_1, \delta_1, w_1 \rangle$ be a DIL. Define $G_2 = \langle W_2, \delta_2, w_2 \rangle$ such that $f_{G_2}(t) = f_{G_1}(\lfloor t/r \rfloor)$. This is easily achieved by introducing a cycle of length r for each direct production of G_1.

(iv) By lemma 3.

(v)-(vi) Trivial.

(vii) 2^t is a growth function while $2^{(2^t)}$ is not. □

We conclude this section with some conjectures. The evidence in favor of in particular conjecture 1 is overwhelming, but we have not been able to derive a formal proof.

Conjecture 1. Growth functions are not closed under multiplication. (E.g. $2^{t+\lfloor \log_2 t \rfloor}$ can hardly be a growth function.)

Conjecture 2. Unbounded growth functions are closed under function inverse. (E.g. $f(t) = r^t$ is a growth function for r is a constant. $g(t) \sim f^{-1}(t) = \log_r t$ is a growth function too.)

Conjecture 3. There are no PD1L growth functions $f(t) \asymp t^r$ where r is not a natural number. (It is hard to see how a string can determine its length in the PD1L case.)

3.3. Hierarchy

The first PD1L growth function of growth type $1\frac{1}{2}$ was "Gabor's sloth" in [75, p.338]. Examples 3-6 and section 3.2. provide us with an ample supply of this growth type. A more difficult problem is to construct a DIL of growth type $2\frac{1}{2}$. The first (and until now only) DIL of growth type $2\frac{1}{2}$ is the PD2L of Karhumäki [50] with growth function f where $2^{\sqrt{t}} \leq f(t) \leq (2^{\sqrt{3}})^{\sqrt{t}}$. By lemma 2 we can construct a PD1L G such that $2^{\sqrt{\lfloor t/2 \rfloor}} + \lfloor t/2 \rfloor \leq f_G(t) \leq (2^{\sqrt{3}})^{\sqrt{\lfloor t/2 \rfloor}} + \lfloor t/2 \rfloor$. From these results and theorem 5 (i) follows:

Theorem 6. There are PD1L growth functions of growth types $1\frac{1}{2}, 2, 2\frac{1}{2}, 3$ which are not DOL growth functions.

Hence the family of (P)DOL growth functions is properly contained in the family of (P)D1L growth functions. However, if we restrict ourselves to the bounded growth functions the situation is different.

Theorem 7. Let G be any DIL such that f_G is of (i) growth type 0, or, (ii) growth type 1. Then we can construct a DOL G' such that $f_{G'} = f_G$.

Proof.

(i) Let $f_G(t) > 0$ for all $t \le t_0$ for some t_0 and $f_G(t) = 0$ otherwise. Then $f_{G'} = f_G$ where $G' = \langle W',\delta',w'\rangle$ is a DOL constructed as follows:

$$W' = \{a_0,a_1,\dots,a_{t_0},b\};\ w' = a_0 b^{f_G(0)-1};$$

$$\delta'(a_i) = a_{i+1} b^{f_G(i+1)-1} \qquad \text{for all } i,\ 0 \le i < t_0,$$

$$\delta'(b) = \delta'(a_{t_0}) = \lambda.\ [6]$$

(ii) If f_G is of growth type 1 for some DIL G then f_G is ultimately periodic, i.e. $f_G(t) = f_G(t-u)$ for all $t > t_0+u$ for some t_0 and u. The construction of the appropriate DOL G' is similar to the construction in (i). □

Corollary 1. The family of bounded (P)DIL growth functions coincides with the family of bounded (P)DOL growth functions.

Theorem 8. Let $G = \langle W,\delta,w\rangle$ be a *unary* (i.e. $\#W = 1$) DIL. Then there is a DOL G' such that $f_{G'} = f_G$.

Proof. Suppose f_G is bounded. By theorem 7 the theorem holds. Suppose f_G is unbounded, and let G be a D(m,n)L. Furthermore, let $p = lg(\delta(a^m,a,a^n))$, $x = \Sigma_{i=0}^{m-1} lg(\delta(a^i,a,a^n)) + \Sigma_{j=0}^{n-1} lg(\delta(a^m,a,a^j))$. Since f_G is unbounded there is a t_0 such that $f_G(t_0) \ge 2(m+n)+x+1$. For all $t \ge t_0$ the following equation holds:

$$(4) \qquad f_G(t+1) = p(f_G(t)-m-n) + x.$$

Case 1. $p = 0$. Then $f_G(t) \le (m+n)y$ where $y = \max\{lg(\delta(v_1,a,v_2)) \mid v_1,v_2 \in W^*\}$. Therefore f_G is bounded: contradiction.

Case 2. $p = 1$. Then $x-m-n > 0$ since f_G is bounded otherwise. It is easy to construct a DOL G' such that $f_{G'} = f_G$ in this case.

Case 3. $p > 1$. Construct a DOL $G'' = \langle W'',\delta'',w''\rangle$ as follows:

$$W'' = \{a_0,a_1,a_2,a_3\};\ \delta''(a_0) = \lambda,\ \delta''(a_1) = a_0a_1a_3^{p-2},\ \delta''(a_2) = a_2a_3^{x+p-1},$$

$$\delta''(a_3) = a_3^{p};\ w'' = (a_0a_1)^{m+n}a_2a_3^{f_G(t_0)-2(m+n)-1}\ [6]$$

It is easy to prove by induction on t that $f_{G''}(t) = f_G(t+t_0)$ for all t. By using

[6] We define δ for DOLs as in section 2.

theorem 7 we construct a DOL $G' = \langle W', \delta', w' \rangle$ such that $W'' \subseteq W'$, $\delta'' \subseteq \delta'$, $\delta'^{t_0}(w') = w''$ and $f_{G'}(t) = f_G(t)$ for $0 \le t < t_0$. Then $f_{G'} = f_G$. □

It may be worthwhile to note that the solution to the difference equation (4) is given by:

$$f_G(t) = \begin{cases} f_G(t_0) + (x-m-n)(t-t_0) & \text{for } p = 1, \\ \\ p^{t-t_0} f_G(t_0) + (x-p(m+n)) \dfrac{1-p^{t-t_0}}{1-p} & \text{for } p > 1, \end{cases}$$

for all $t > t_0$.

Therefore, the growth function of a unary DIL is either linear or purely exponential, which by equation (1) gives us

<u>Corollary 2</u>. The family of growth functions of unary DILs is properly contained in the family of growth functions of DOLs.

<u>Theorem 9</u>. There is a *binary* PD1L $G = \langle W, \delta, w \rangle$, (i.e. $\#W = 2$), with a one letter axiom such that there is no DOL G' such that $f_{G'} = f_G$.

<u>Proof</u>. Let $G = \langle W, \delta, w \rangle$ be a PD(1,0)L where

$$W = \{a,b\};\ w = a;\ \delta(\lambda,a,\lambda) = b,\ \delta(\lambda,b,\lambda) = aa,\ \delta(a,a,\lambda) = a,$$
$$\delta(b,a,\lambda) = b,\ \delta(b,b,\lambda) = b,\ \delta(a,b,\lambda) = aa.$$

The initial sequence of produced strings is:

a, b, aa, ba, aab, $baaa$, $aabaa$, ba^3ba, a^2ba^4b, ba^3ba^5, $a^2ba^4ba^4$, $ba^3ba^5ba^3$, $a^2ba^4ba^6ba^2$, $ba^3ba^5ba^7ba$, $a^2ba^4ba^6ba^8b$, $ba^3ba^5ba^7ba^9$, ...

Every second time step one b is introduced on the left and starts moving along the string to the right. Every time step b moves one place to the right and leaves a string a^2 on the place it formerly occupied. When a letter b reaches the right end of the string it disappears in the next step leaving aa. Therefore, on the one hand, every second production step there enters a length increasing element in the string; on the other hand, with exponentially increasing time intervals one of these elements disappears. The strings where a b has just disappeared in the above sequence are:

$$\delta^5(a) = baaa,\ \delta^9(a) = ba^3ba^5,\ \delta^{15}(a) = ba^3ba^5ba^7ba^9.$$

Now introduce the notational convenience $\Pi_{i=1}^{x} v(i)$ where $v(i)$ is a function from $\mathbb{N}$ into W^*. E.g. if $v(i) = a^i b^{2i}$ then $\Pi_{i=1}^{3} v(i) = ab^2a^2b^4a^3b^6$.

Claim. $\delta^{t(x)}(a) = \prod_{i=1}^{2^x} ba^{2i+1}$ where $t(x) = 2^{x+1} + 2x + 3$.

Proof of claim. By induction on x.
$x = 0$. $\delta^5(a) = ba^3$.
$x > 0$. Suppose the claim is true for all $x \le n$. Then

$$\delta^{t(n)}(a) = \prod_{i=1}^{2^n} ba^{2i+1} = \ldots ba^{2.2^n+1}.$$

This last occurence of b will just have disappeared at time $t' = t(n) + 2.2^n + 2 = t(n+1)$. The distance with the preceding occurence of b was 2.2^n-1 and therefore

(5) $$\delta^{t(n+1)}(a) = \ldots ba^{2.2^n-1 + 2(2.2^n+2) - 2(2^n+1)} = \ldots ba^{2.2^{n+1}+1}.$$

At time t(n) the total number of occurences of b in the string was 2^n; at time t(n+1) this is $2^n+2^n+1-1 = 2^{n+1}$ and

(6) $$\delta^{t(n+1)}(a) = ba^3b\ldots .$$

Since it is easy to see that for all $t \ge 0$ holds if $\delta^t(a) = v_1ba^{i_1}ba^{i_2}bv_2$ for some v_1,v_2 then $i_2 = i_1+2$, it follows from (5) and (6) that $\delta^{t(n+1)}(a) = \prod_{i=1}^{2^{n+1}} ba^{2i+1}$, which proves the claim.

Hence,

$$f_G(t(x)) = \sum_{i=1}^{2^x} 2(i+1) = 2^x(2^x+3) = 1/4(t(x)-2x-3)(t(x)-2x+3)$$

$$= 1/4\ t(x)^2 - x\ t(x) + x^2 - 9/4.$$

Since $t(x) = 2^{x+1}+2x+3$ we have $x \approx \lfloor \log_2 t(x)/2 \rfloor$ and therefore:

(7) $$f_G(t(x)) \approx 1/4\ t(x)^2 - \lfloor \log_2 t(x)/2 \rfloor t(x) + \lfloor \log_2 t(x)/2 \rfloor^2 - 9/4.$$

From (7) and the general formula for a DOL growth function (1) it follows that f_G cannot be a DOL growth function since

$$f_G(t) - 1/4\ t^2 \sim t \log t. \quad \square$$

That context dependent L systems using a two letter alphabet cannot yield all DOL growth functions is ascertained by the counterexample $f(0) = f(1) = f(2) = 1$ and

$f(t) = t$ for $t > 2$, which is surely a (P)D0L growth function.

Corollary 3. The family of binary (P)D1L growth functions has nonempty intersections with the family of (P)D0L growth functions and neither contains the other.

An open problem in this area is: does the family of (P)D1L growth functions coincide with the family of (P)D2L growth functions. A proof of conjecture 3 would show that the family of PD1L growth functions is properly contained in the family of PD2L growth functions.

Using a similar technique as in lemma 2 we can, however, say the following.

Theorem 10.
(i) If $f(t)$ is a PD2L growth function then $f(t)$ is a D(2,0)L growth function.
(ii) If $f(t)$ is a D2L growth function then $f(t)+1$ is a D(2,0)L growth function.

Proof.
(i) Let $G = \langle W,\delta,w\rangle$ be a PD2L. Define a D(2,0)L $G' = \langle W',\delta',w'\rangle$ as follows.

$$W' = W \cup W \times \{¢\} \text{ where } ¢ \notin W;\ w' = a_1a_2\ldots a_{n-1}(a_n,¢) \text{ for } w = a_1a_2\ldots a_n;$$
$$\delta'(ab,c,\lambda) = \delta(a,b,c), \qquad \delta'(\lambda,c,\lambda) = \lambda,$$
$$\delta'(ab,(c,¢),\lambda) = \delta(a,b,c)a_1a_2\ldots a_{m-1}(a_m,¢) \quad \text{if } \delta(b,c,\lambda) = a_1a_2\ldots a_m,$$
$$\delta'(\lambda,(c,¢),\lambda) = a_1a_2\ldots a_{m-1}(a_m,¢) \quad \text{if } \delta(\lambda,c,\lambda) = a_1a_2\ldots a_m,$$

for all $b,c \in W$ and all $a \in W\cup\{\lambda\}$.

Then $\delta'^t(w') = b_1b_2\ldots b_{m-1}(b_m,¢)$ if $\delta^t(w) = b_1b_2\ldots b_m$, and therefore $f_{G'} = f_G$.

(ii) Let $G = \langle W,\delta,w\rangle$ be a D2L. Define a D(2,0)L $G' = \langle W',\delta',w'\rangle$ as follows.

$$W' = W\cup\{¢\} \text{ where } ¢ \notin W;\ w' = w¢;$$
$$\delta'(ab,c,\lambda) = \delta(a,b,c),\ \delta'(\lambda,c,\lambda) = \lambda,$$
$$\delta'(ab,¢,\lambda) = \delta(a,b,\lambda)¢,\ \delta'(\lambda,¢,\lambda) = ¢,$$

for all $b,c \in W$ and all $a \in W\cup\{\lambda\}$.

Then $\delta'^t(w') = \delta^t(w)¢$ and therefore $f_{G'}(t) = f_G(t)+1$. □

Rozenberg [86] proved that a D(m,n)L can be simulated in real time by a D(k,ℓ) if $k+\ell = m+n$ and $k,\ell,m,n > 0$. Therefore, by using the same trick as above we have the following:

Corollary 4.
(i) If $f(t)$ is a PD(m,n)L growth function then $f(t)$ is a D(k,ℓ) growth function where $k+\ell = m+n$.

(ii) If f is a $D(m,n)L$ growth function then $f(t)+1$ is a $D(k,\ell)$ growth function where $k+\ell = m+n$.

In particular, (i) and (ii) hold for $k = m+n$ and $\ell = 0$ and vice versa.

3.4. Decision problems

According to section 2 and the beginning of section 3 (and the references contained therein) the analysis, synthesis, growth equivalence, classification and structural problems all have a positive solution for context independent growth, i.e. there is an algorithm which gives the required answer or decides the issue. (This is not completely true for the synthesis problem, see theorem 33 in Paz & Salomaa [75].) The corresponding problems for the general case of DIL systems have been open. It will be shown here that for DILs these problems all have a negative solution essentially because already PD1Ls can simulate any effective process. (Note that by theorems 8 and 9 the above problems have a positive solution if we restrict ourselves to unary DILs or to DILs with a bounded growth function.) Furthermore, we shall show that similar questions concerning growth ranges of DILs have similar answers. First we need the notion of a *Tag system*[7]. A Tag system is a 4 tuple $T = \langle W,\delta,w,\beta\rangle$ where W is a finite nonempty *alphabet*, δ is a total mapping from W into W^*, $w \in WW^*$ is the *initial string*, and β is a positive integer called the *deletion number*. The operation of a Tag system is inductively defined as follows: the initial string w is generated by T in 0 steps. If $w_t = a_1a_2\ldots a_n$ is the t-th string generated by T then $w_{t+1} = a_{\beta+1}a_{\beta+2}\ldots a_n\delta(a_1)$ is the (t+1)-th string generated by T.

Lemma 4 (Minsky). It is undecidable for an arbitrary Tag system T with $\beta = 2$ and a given positive integer k whether T derives a string of length less than or equal to k. In particular it is undecidable whether T derives the empty word.

We shall now show that if it is decidable whether or not an arbitrary PD1L has a growth function of growth type 1 then it is decidable whether or not an arbitrary Tag system with deletion number 2 derives the empty word λ. Therefore, by lemma 4 it is undecidable whether a PD1L has a growth function of type 1.

Let $T = \langle W_T,\delta_T,w_T,2\rangle$ be any Tag system with deletion number 2. Define a PD(1,0)L $G = \langle W,\delta,w\rangle$ as follows:[8]

$$W = W_T \cup W_T' \cup W_T{\times}W_T \cup \{\text{¢},\$\},$$

where $W_T' = \{\underset{\sim}{a} \mid a \in W_T\}$, $W_T' \cap W_T = \phi$ and $\text{¢},\$ \notin W_T \cup W_T'$;

---

[7] Minsky, Op. cit.

[8] The idea of simulating Tag systems with 1Ls occurs already in the first papers on L systems i.e. [33] and [12].

$$w = w_T ¢;$$

$$\delta(\lambda,a,\lambda) = \delta(\$,a,\lambda) = \delta(\$,(a,b),\lambda) = \underset{\sim}{a},$$

$$\delta(\lambda,\underset{\sim}{a},\lambda) = \delta(\$,\underset{\sim}{a},\lambda) = \delta(\lambda,\$,\lambda) = \delta(\$,\$,\lambda) = \delta(\$,¢,\lambda) = \delta(\lambda,¢,\lambda) = \$,$$

$$\delta(a,b,\lambda) = \delta(\underset{\sim}{a},(b,c),\lambda) = \delta(a,(b,c),\lambda) = b,$$

$$\delta(a,¢,\lambda) = ¢,$$

$$\delta(\underset{\sim}{b},c,\lambda) = \delta((a,b),c,\lambda) = (c,b),$$

$$\delta(\underset{\sim}{b},¢,\lambda) = \delta((a,b),¢,\lambda) = \delta_T(b)¢,$$

for all $a,b,c \in W_T$ and all $\underset{\sim}{a},\underset{\sim}{b}, \in W_T'$.

A sample derivation is:

T	G
$a_1a_2a_3a_4a_5$	$a_1a_2a_3a_4a_5¢$
$a_3a_4a_5\delta_T(a_1)$	$\underset{\sim}{a}_1a_2a_3a_4a_5¢$
$a_5\delta_T(a_1)\delta_T(a_3)$, etc.	$\$(a_2,a_1)a_3a_4a_5¢$
	$\$\underset{\sim}{a}_2(a_3,a_1)a_4a_5¢$
	$\$\$a_3(a_4,a_1)a_5¢$
	$\$\$\underset{\sim}{a}_3a_4(a_5,a_1)¢$
	$\$\$\$(a_4,a_3)a_5\delta_T(a_1)¢$
	$\$\$\$\underset{\sim}{a}_4(a_5,a_3)\delta_T(a_1)¢$, etc.

In the simulating PD1L G signals depart from the left, with distances of one letter in between, and travel to the right at an equal speed of one letter per time step. Therefore, the signals cannot clutter up. It is clear that if the Tag system T derives the empty word, then there is a time t_0 such that $\delta^{t_0}(w) = \$^k¢$ and $\delta^t(w) = \$^{k+1}$ for some k and for all $t > t_0$. Conversely, the only way for G to be of growth type 1 is to generate a string of the form $\$^k¢$. (If the string always contains letters other than $ and ¢ then at each second production step there appears a new occurrence of $ and the string grows indefinitely long.) Therefore, T derives the empty word iff G is of growth type 1. Since it is undecidable whether or not an arbitrary Tag system with deletion number 2 derives the empty word it is undecidable whether or not a PD1L is of growth type 1.

Theorem 11.

(i) It is undecidable whether or not an arbitrary PD1L is of growth type i, $i \in \{1,1\frac{1}{2},2,2\frac{1}{2},3\}$.

(ii) It is undecidable whether or not an arbitrary D1L is of growth type i, $i \in \{0,1,1\frac{1}{2},2,2\frac{1}{2},3\}$.

(iii) It is undecidable whether an arbitrary PD1L has an unbounded growth function.

Proof.

(i) Let $G_1 = \langle W_1, \delta_1, w_1 \rangle$ be a PD(1,0)L simulating a Tag system T as discussed above. Let $G_2 = \langle W_2, \delta_2, w_2 \rangle$ be a PD(1,0)L of growth type i, $i \in \{1, 1\frac{1}{2}, 2, 2\frac{1}{2}, 3\}$ such that $W_2 \cap W_1 = \phi$. Define $G_3 = \langle W_3, \delta_3, w_3 \rangle$ as follows:

$$W_3 = W_2 \cup \{\$\};\ w_3 = w_2;$$
$$\delta_3 = \delta_2 \cup \{\delta_3(\$,\$,\lambda) = \delta_3(\lambda,\$,\lambda) = \$\} \cup \{\delta_3(\$,a,\lambda) = \delta_2(\lambda,a,\lambda) \mid a \in W_2\}.$$

Clearly, $f_{G_3} = f_{G_2}$. Now construct a PD(1,0)L $G_4 = \langle W_4, \delta_4, w_4 \rangle$ as follows:

$$W_4 = W_3 \cup W_1;\ w_4 = w_1;$$
$$\delta_4 = \delta_3 \cup (\delta_1 - \{\delta_1(\$,¢,\lambda) = \$\}) \cup \{\delta_4(\$,¢,\lambda) = w_3\}.$$

If there is a time t_0 such that $\delta_1^{t_0}(w_1) = \$^k¢$ for some k then $\delta_4^{t_0}(w_4) = \$^k¢$ and $\delta_4^{t+t_0+1}(w_4) = \$^k\delta_3^t(w_3)$ for all t, i.e. $f_{G_4}(t+t_0+1) = f_{G_2}(t)+k$. If there is no such time t_0 then $f_{G_4}(t) = f_{G_1}(t)$ for all t. In this latter case it is easy to see that $f_{G_1}(t) \asymp t$, i.e. G_4 is of growth type 2. By the previous discussion it is undecidable whether such a time t_0 exists and therefore whether f_{G_4} is of growth type i or 2.

(ii) Follows by a similar argument if we talk about D(1,0)Ls instead of PD(1,0)Ls, change everywhere $\delta_.(\lambda,\$,\lambda) = \$$ into $\delta_.(\lambda,\$,\lambda) = \lambda$, and let i range over $\{0, 1, 1\frac{1}{2}, 2, 2\frac{1}{2}, 3\}$.

(iii) Follows from (i). □

Corollary 5. There is no algorithm which, for an arbitrary PD1L G, gives an explicit expression for f_G in a formalism we can use.

The undecidability of whether a (P)D1L is of a certain growth type holds (because of the proof method) also for future refinements of the classification. We could have proved theorem 11 by simulating Turing machines with PD1Ls (cf. lemmas 1 and 2) and reducing everything to the printing problem for Turing machines. This, however, would have caused some difficulties with the slow growth types.

Theorem 11 has some interesting corollaries. Two DIL systems G_1, G_2 are said to be *language equivalent* if $L(G_1) = L(G_2)$. Now it is known that the language equivalence for e.g. OL languages is undecidable. The status of the language equivalence problem for DOL languages is unknown as yet. (Cf. [5,99,84].) By the special tractable nature of PD1L systems it might well be that the language equivalence problem is decidable in this case. However, in the proof of theorem 11 (i) it is clearly undecidable whether $L(G_4) = L(G_1)$. Therefore we have:

Corollary 6. The language equivalence of PD1L languages is undecidable. (According to theorem 12 this is even the case if we are informed in advance that both PD1Ls concerned are of the same growth type i, $i \in \{2, 2\frac{1}{2}, 3\}$.)

V.I. Varshavsky proposed the following problem: "Consider the class of D2L grammars producing strings which stabilize at a certain length. Make some reasonable assumptions about the maximal production length (e.g. 2) and axiom length (e.g. 1) and find the maximal stable string length as a function of the number of letters in the alphabet."[9] The restrictions as stated in the above problem are no restrictions on the generating power of any usual subfamily of DILs since it is clear that by enlarging the alphabet we can simulate any DIL G_1 by a DIL G_2 where G_2 takes k_1 production steps to generate the axiom of G_1 and takes a constant number k_2 of productions steps of G_2 to simulate one production step of G_1, i.e. $\delta_2^{k_1+k_2t}(w_2) = \delta_1^t(w_1)$ for all t. (This is similar to deriving e.g. the Chomsky Normal Form for context free grammars.) Suppose we restrict ourselves to the family of PD1Ls and there is a function as proposed by Varshavsky where, moreover, this function is computable. Then it would also be decidable whether or not a PD1L G simulating a Tag system T ever generates a string of the form $\$^k¢$ for some k: contradicting lemma 4. Therefore, we have

Corollary 7. Let V_i be the family of PD1Ls $G = \langle W,\delta,w\rangle$ such that $\#W = i$, $w \in W$, $lg(\delta(a,b,\lambda)) \leq 2$ for all $b \in W$ and $a \in W\cup\{\lambda\}$, and $lg(\delta^{t_0+t}(w)) = lg(\delta^{t_0}(w))$ for some t_0 and all t. Let $v(i) = \max\{lg(v) | v\in L(G) \text{ and } G\in V_i\}$. There is no computable function f such that $v(i) \leq f(i)$ for all i, i.e. v increases faster than any computable function and hence Varshavsky's problem has a negative solution.

Theorem 12.

(i) It is undecidable whether or not two PD1Ls are growth equivalent even if we have the advance information that they are of the same growth type i, $i \in \{2, 2\frac{1}{2}, 3\}$.

(ii) It is undecidable whether or not two D1Ls are growth equivalent even if we have the advance information that they are of the same growth type i, $i \in \{1\frac{1}{2}, 2, 2\frac{1}{2}, 3\}$.

(iii) The growth equivalence of two DILs is decidable if we have the advance information that they both have bounded growth functions.

Proof. Take an arbitrary Tag system T and simulate it with a PD1L G_1 as in the proof of theorem 11.

(i) Now construct two variants of G_1, called G_2 and G_3, which act like G_1 until $\$¢$ occurs in a string. Then G_2 and G_3 start different growths albeit of the same

[9] In: *Unusual automata theory*. Univ. of Aarhus, Comp. Sci. Dept. Tech. Rept. DAIMI PB-15 (1973), 20.

growth type i, $i \in \{2,2\frac{1}{2},3\}$. Now let f be another growth function of type i. Since PD1L growth functions are closed under addition (theorem 5) both $g = f_{G_2}+f$ and $h = f_{G_3}+f$ are PD1L growth functions of type i, say of G_4 and G_5. If $\$¢$ never occurs in a string then $f_{G_4} = f_{G_5} = f_{G_1}+f$ and $f_{G_1}(t) \asymp t$. If $\$¢$ occurs in a string then $f_{G_4} \neq f_{G_5}$. Since it is undecidable whether $\$¢$ occurs in a string it is undecidable whether or not $f_{G_4} = f_{G_5}$, where it is known that both f_{G_4} and f_{G_5} are of growth type i, $i \in \{2,2\frac{1}{2},3\}$.

(ii) Similar to (i). Since we talk here about D1Ls we can slow the growth function f_{G_1} down to $f_{G'_1}$ where $f_{G'_1} < \log_r t$, $r > 1$, (cf. discussion after example 9).

(iii) Trivial. □

Note that the theorem above leaves open the decidability of the question of two PD1Ls being growth equivalent if we are informed in advance that they are both of growth type $1\frac{1}{2}$. This is because in our simulation method of Tag systems all simulating PD1Ls are either of growth type 1 or growth type 2.

Theorem 13. It is undecidable whether two PD2Ls are growth equivalent even if we are informed in advance that they are both of growth type $1\frac{1}{2}$.

Proof. Take a PD2L G_1 simulating a Tag system T. Construct a PD2L G_2 which simulates G_1 such that $f_{G_2}(t) < \log_r f_{G_1}(t)$ (cf. discussion after example 9). Since $f_{G_1}(t) \asymp t$ or $f_{G_1}(t) \leq m$ for some constant m, f_{G_2} is of growth type $1\frac{1}{2}$ or 1. Then use the method of proof of theorem 12 (i). □

Theorems 11-13 have analogues for the growth ranges of DIL systems. The *growth range* of a DIL G is defined by $R(G) = \{lg(v) \mid v \in L(G)\}$. Although the results on growth ranges are not corollaries of theorems 11-13 they follow by the same proof method. Two DILs G_1 and G_2 are said to be *growth range equivalent* iff $R(G_1) = R(G_2)$.

Theorem 14. The growth range equivalence is undecidable for two PD1Ls G_1 and G_2 even if we have advance information that they both are of growth type i, $i \in \{2,2\frac{1}{2},3\}$.

Proof. The proof of theorem 12 (i) will do since we can choose f_{G_1} and f_{G_2} such that they are strictly increasing at different rates iff a substring $\$¢$ occurs. □

Under appropriate interpretation we can prove the undecidability of growth range type classification etc. analogous to theorem 11-13. Note, however, that the growth range type can be different from the growth function type of a DIL. E.g. $f_G(t) = 2^{\lfloor \log_2 t \rfloor}$ is of growth type 1 whereas $R(G) = \{2^i \mid i \geq 0\}$ and therefore is exponential.

A fortiori all undecidability results above hold under appropriate interpretation also for nondeterministic context dependent L systems.