ON THE DECIDABILITY OF SOME EQUIVALENCE PROBLEMS FOR DOL-SYSTEMS

by

Mogens Nielsen

DAIMI PB-20
December 1973

ON THE DECIDABILITY OF SOME EQUIVALENCE PROBLEMS FOR DOL-SYSTEMS

by

Mogens Nielsen
Department of Computer Science
University of Aarhus
Denmark

December 1973

48 pages.

1 figure.

Proposed running head: "Equivalence of DOL-systems".

Symbols used:

N – may be typed as an italic or a gothic N.

All script letters are handwritten in this manuscript ($\mathcal{W}$, $\mathcal{P}$, $\mathcal{N}$, $\mathcal{L}$, and $\mathcal{S}$).

ABSTRACT

One of the questions of the longest open standing in the area of Lindenmayer-systems is the decidability of the equivalence-problem for deterministic, informationless L-systems (DOL-systems). This and some related equivalence-problems (equivalence with respect to the set and the sequence of generated words, Parikh-vectors and word-lengths) are investigated. Some of these related problems are shown to be recursively solvable, and the implications of htese results on the main open problem mentioned above are discussed.

0. INTRODUCTION

L-systems were introduced in 1968 by A. Lindenmayer [2], originally in connection with some problems in theoretical biology (L-systems are models describing the development of filamentous organisms). Since then the systems have been studied intensively from the viewpoint of the theory of formal languages [8].

L-systems can be viewed as generating devices corresponding to the grammars usually considered in the theory of formal languages. But in L-systems one does not distinguish between terminals and non-terminals, productions are applied in parallel on a word, and the starting string of the systems can be of length greater than one.

One of the problems of the longest open standing in the area of L-systems, is the decidability of the equivalence problem for deterministic, informationless (context-free) L-systems (DOL-systems), i.e., the problem of deciding for any two DOL-systems whether or not they generate the same language. It has been shown, see [5], [6], and [7], that the equivalence problem for corresponding non-deterministic L-systems is undecidable.

The biological motivation behind this problem is clear: given descriptions of the development of two filamentous organisms one wants to decide whether the sets of cell-patterns occurring in the life time of the two organisms are the same. But from a biological point of view it is perhaps more interesting to ask the question of whether two organisms develop identically, i.e., whether two DOL-systems generate the same words in the same sequence.

Also other related problems are biologically motivated. Suppose one is not interested in where particular cell-types occur in patterns, but just the number of occurrences of different cell-types in patterns, then the problems mentioned above are "weakened" to the problems of deciding whether two DOL-systems generate the same set (sequence) of Parikh-vectors. Furthermore, suppose one is not even interested in which cell-types occur in patterns, but just in the sizes of patterns, then the corresponding problems are to decide whether two DOL-systems generate the same set (sequence) of lengths.

The above mentioned problems are the subject of this paper.

1. DEFINITIONS

Definition 1.1

A DOL-system is an ordered triple $H = <\Sigma, h, x>$, where Σ is a finite non-empty set of symbols (the alphabet of the system), h is a homomorphism (with respect to the concatenation-operator) mapping Σ to Σ^* $(h : \Sigma \to \Sigma^*)$, and x is a non-empty string of symbols from Σ, $x \in \Sigma^+$, (the axiom of the system).

A DOL-system is called propagating (PDOL) iff $h(\sigma) \neq \lambda$ (the empty word) for every $\sigma \in \Sigma$, i.e., $h : \Sigma \to \Sigma^+$.

Definition 1.2

The word-language (usually just called the language) generated by a DOL-system H as in definition 1.1 is the set of words over Σ defined as follows:

$$\mathcal{W}\mathcal{L}(H) = \{h^i(x) \mid i \geq 0\}$$

(where $h^0(x) = x$).

The word-sequence generated by H is defined as the ordered, infinite sequence of words generated by H – conveniently denoted as the set

$$\mathcal{W}\mathcal{S}(H) = \{(i, h^i(x)) \mid i \geq 0\}$$

Let N denote the set of non-negative integers:

Let $\Sigma = \{\sigma_1, \sigma_2, \ldots, \sigma_n\}$ be a finite set of symbols. For any $\sigma_i \in \Sigma$ and $x \in \Sigma^*$, $\#_{\sigma_i}(x)$ denotes the number of occurrences of σ_i in x. The function $\pi : \Sigma^* \to N^n$ is defined in the following way:

$$\pi(x) = (\#_{\sigma_1}(x), \#_{\sigma_2}(x), \ldots, \#_{\sigma_n}(x)),$$

i.e., π associates with each word $x \in \Sigma^*$ its corresponding Parikh-vector.

The relations $=$, $<$, and $\leq$ on $\grave{N}^n$ are frequently referred to in this paper. They are defined as follows:

$i = 1, 2 : v_i = (v_1^i, v_2^i, \ldots, v_n^i) \in \grave{N}^n$

$v_1 = v_2$ iff $\forall 1 \leq j \leq n : v_j^1 = v_j^2$

$v_1 \leq v_2$ iff $\forall 1 \leq j \leq n : v_j^1 \leq v_j^2$

$v_1 < v_2$ iff $(v_1 \leq v_2) \wedge (v_1 \neq v_2)$

$\leq$ is a partial ordering of $\grave{N}^n$. If neither $v_1 \leq v_2$ nor $v_2 \leq v_1$, then v_1 and v_2 are said to be incomparable.

Definition 1.3

The Parikh-language (Parikh-sequence) generated by a DOL-system H as in definition 1.1 with $\Sigma = \{\sigma_1, \sigma_2, \ldots, \sigma_n\}$, is the set of vectors from $\grave{N}^n$ (from $\grave{N} \times \grave{N}^n$) defined as

$$\mathcal{PL}(H) = \{ \pi(h^i(x)) \mid i \geq 0 \}$$

$$(\mathcal{PS}(H) = \{ (i, \pi(h^i(x))) \mid i \geq 0 \})$$

Remark

Let H be a DOL-system as in definition 1.1 with $\Sigma = \{\sigma_1, \sigma_2, \ldots, \sigma_n\}$. Define the $n \times n$ matrix $M_H = [m_{ij}^H] = [\#_{\sigma_j}(h(a_i))]$ called the growth-matrix of H. Then the following equality (see [4]) can be used to characterize $\mathcal{PL}$ and $\mathcal{PS}$ (vectors written as row-vectors) :

$$\pi(h^i(x)) = \pi(x) \cdot M_H^i$$

The length of a word $x \in \Sigma^*$ is denoted by $|x|$.

Definition 1.4

The length-language (length-sequence) generated by a DOL-system H as in definition 1.1 is the set of natural numbers (of pairs of natural numbers) defined as:

$$\mathcal{NL}(H) = \{\ |h^i(x)|\ \mid i \geq 0\}$$

$$(\ \mathcal{NS}(H) = \{\ (i,\ |h^i(x)|)\ \mid i \geq 0\}).$$

Remark

$\mathcal{NL}$ and $\mathcal{NS}$ can be characterized in the same terms as $\mathcal{PL}$ and $\mathcal{PS}$ in the remark after definition 1.3. Let η denote the n-dimensional column-vector with every entry equal to 1. Then since $|x| = \sum_{\sigma \in \Sigma} \#_\sigma(x)$

$$|h^i(x)| = \pi(x) \cdot M_H^i \cdot \eta.$$

Definition 1.5

Let $\mathbb{X}$ be one of the operators $\mathcal{WL}$, $\mathcal{WS}$, $\mathcal{PL}$, $\mathcal{PS}$, $\mathcal{NL}$ or $\mathcal{NS}$, then $\mathbb{X}$ is defined in definition 1.2, 1.3 or 1.4 in the form

$$\mathbb{X}(H) = \{ f_{\mathbb{X}}^{H}(i) \mid i \geq 0\}$$

For any of the six operators, $\mathbb{X}$, and $n_1, n_2 \in \dot{N}$ define:

$$\mathbb{X}_{n_1}^{n_2}(H) = \{ f_{\mathbb{X}}^{H}(i) \mid n_1 \leq i < n_2\}$$

i.e. $\mathbb{X}(H) = \bigcup_{n \in \dot{N}} \mathbb{X}_0^n(H) = \mathbb{X}_0^\infty(H).$

Definition 1.6

Let $H = <\Sigma, h, x>$ and $G = <\Sigma, g, y>$ be two DOL-systems over the same alphabet, and $\mathbb{X}$ one of the operators $\mathcal{WL}$, $\mathcal{WS}$, $\mathcal{PL}$, $\mathcal{PS}$, $\mathcal{NL}$, or $\mathcal{NS}$. Then H and G are said to be $\mathbb{X}$-equivalent iff $\mathbb{X}(G) = \mathbb{X}(H)$. The $\mathbb{X}$-equivalence problem is the problem of deciding for any two DOL-systems H and G as above whether or not H and G are $\mathbb{X}$-equivalent.

The purpose of the next sections is to discuss whether the six equivalence problems defined in this section are recursively solvable, i.e., whether or not there exists an algorithm that decides for any two DOL-systems whether or not they are $\underline{X}$-equivalent.

<u>Remark</u>

It might seem a little strange to require H and G to be over the same alphabet in the definition of $\mathcal{NL}$- and $\mathcal{NS}$-equivalence, but from the viewpoint of decidability, which is the subject of this paper, this is no restriction. Assume that H and G are two DOL-systems over disjoint alphabets (this can always be obtained by an eventual "marking" of the symbols of one of the alphabets), $H = <\Sigma_H, h, x>$, $G = <\Sigma_G, g, y>$. Define $H' = <\Sigma_H \cup \Sigma_G, h, x>$ and $G' = <\Sigma_H \cup \Sigma_G, g, y>$ where h and g are defined at random on Σ_G and Σ_H respectively. Then clearly H and G are $\mathcal{NL}(\mathcal{NS})$-equivalent iff H' and G' are $\mathcal{NL}(\mathcal{NS})$-equivalent.

The requirements on the alphabets seem more reasonable in the definitions of word- and Parikh-equivalences. However, one might argue that trivial cases,in which the systems are intuitively "equivalent", are not included in definition 1.6 (like the case where one of the systems contains dummy symbols never occurring in any of the generated words or the case where equivalent symbols are called by different names). One might suggest the following alternative definition of language-equivalence:

Given the two DOL-systems H and G, extend these systems to H' and G' as above. H and G are said to be $\mathcal{WL}$ ($\mathcal{PL}$)-equivalent iff there exists a coding c (an automorphism (with respect to concatenation) in $\Sigma_H \cup \Sigma_G$) such that $\mathcal{WL}(H') = c(\mathcal{WL}(G'))$ ($\mathcal{PL}(H') = c(\mathcal{PL}(G'))$).

A similar alternative definition of sequence-equivalence can be given in a straightforward way. But since there exist only finitely many different codings from a finite alphabet to itself, there is no difference between the original and the alternative definitions of equivalence with respect to decidability.

2. VARIOUS RESULTS ON DOL-SYSTEMS

This section contains a few results on DOL-systems needed in the next sections.

Lemma 2.1

Let $H = <\Sigma, h, x>$ be a DOL-system. Then there exists a constant $k_H \in \mathbb{N}$ such that

1) $\{\pi(h^i(x)) \mid 0 \leq i < k_H\} = \mathcal{SL}_0^{k_H}(H)$ is a set of linearly independent vectors

2) $\forall j, j \geq k_H : \pi(h^j(x))$ is a linear combination of the set from 1).

Proof

Define k_H as the largest integer satisfying 1). Construct the growth-matrix of H, M_H, as mentioned in section 1. By the construction $\pi(h^{k_H}(x))$ is a linear combination of the set from 1)

$$\pi(h^{k_H}(x)) = \sum_{i=0}^{k_H-1} c_i \cdot \pi(h^i(x)) = \sum_{i=0}^{k_H-1} c_i \cdot (\pi(x) \cdot M_H^i).$$

From this it follows that

$$\pi(h^{k_H+1}(x)) = (\pi(x) \cdot M_H^{k_H}) \cdot M_H$$

$$= \left(\sum_{i=0}^{k_H-1} c_i \cdot (\pi(x) \cdot M_H^i)\right) \cdot M_H = \sum_{i=0}^{k_H-1} c_i \cdot (\pi(x) \cdot M_H^{i+1})$$

$$= \left(\sum_{i=0}^{k_H-2} c_i \cdot (\pi(x) \cdot M_H^{i+1})\right) + c_{k_H-1} \cdot (\pi(x) \cdot M_H^{k_H})$$

$$= \left(\sum_{i=1}^{k_H-1} c_{i-1} \cdot (\pi(x) \cdot M_H^i)\right) + c_{k_H-1} \cdot \left(\sum_{i=0}^{k_H-1} c_i \cdot (\pi(x) \cdot M_H^i)\right)$$

$$= \sum_{i=0}^{k_H-1} c_i' \cdot (\pi(x) \cdot M_H^i) = \sum_{i=0}^{k_H-1} c_i' \cdot \pi(h^i(x)).$$

The equations above can easily be extended to a formal proof of 2) by induction.

Lemma 2.2

Let $H = <\Sigma, h, x>$ and $G = <\Sigma, g, y>$ be two DOL-systems over the same alphabet, and let k_H be the constant from lemma 2.1 corresponding to system H. Then

$$\forall i, 0 \le i \le k_H \quad : \quad \pi(h^i(x)) = \pi(g^i(y))$$

implies

$$\forall i, i \ge 0 \quad : \quad \pi(h^i(x)) = \pi(g^i(y))$$

Proof

Let M_H and M_G be the growth-matrices of H and G respectively. Assume that the condition of the lemma holds for H and G, i.e.,

$$\forall i, 0 \le i \le k_H \; : \; \pi(x) \cdot M_H^i = \pi(y) \cdot M_G^i \; .$$

Then

$$\pi(h^{k_H+1}(x)) = (\pi(x) \cdot M_H^{k_H}) \cdot M_H$$

$$= \left(\sum_{i=0}^{k_H-1} c_i \cdot (\pi(x) \cdot M_H^i)\right) \cdot M_H = \sum_{i=0}^{k_H-1} c_i \cdot (\pi(x) \cdot M_H^{i+1})$$

$$= \sum_{i=0}^{k_H-1} c_i \cdot (\pi(y) \cdot M_G^{i+1}) = \left(\sum_{i=0}^{k_H-1} c_i \cdot (\pi(y) \cdot M_G^i)\right) \cdot M_G$$

$$= \quad (\sum_{i=0}^{k_H-1} c_i \cdot (\pi(x) \cdot M_H^i)) \cdot M_G = (\pi(x) \cdot M_H^{k_H}) \cdot M_G$$

$$= \quad (\pi(y) \cdot M_G^{k_H}) \cdot M_G = \pi(y) \cdot M_G^{k_H+1} = \pi(g^{k_H+1}(y)).$$

A formal proof of lemma 2.1 can be obtained by straightforward extensions of the equations above (proof by induction).

Lemma 2.3

Let H and G be two DOL-systems as in Lemma 2.1. Let $|\Sigma|$ denote the number of symbols of Σ. Then

$$\forall i, 0 \leq i \leq |\Sigma| \quad : \quad \pi(h^i(x)) = \pi(g^i(y))$$

$$\text{iff} \quad \forall i, i \geq 0 \quad : \quad \pi(h^i(x)) = \pi(g^i(y))$$

Proof

It follows directly from lemma 2.1 that the constant k_H is less than or equal to $|\Sigma|$. Lemma 2.3 is now an immediate consequence of lemma 2.2.

Remark

Let $H = <\Sigma, h, x>$ be a DOL-system. Then $H_{nm} = <\Sigma, h^m, h^n(x)>$ is a well defined DOL-system (for which $\mathcal{WL}(H_{nm}) \subseteq \mathcal{WL}(H)$) and from the arguments above it then follows that

$$\forall m, n \in \dot{N}, \forall j \in \dot{N} \; : \; \pi(h^{n+mj}(x)) = \pi((h^m)^j (h^n(x)))$$

is a linear combination of

the set of vectors $\{\pi(h^{n+mi}(x)) | \; 0 \leq i < |\Sigma| \}$.

Lemma 2.4

Let $H = \langle\Sigma, h, x\rangle$ be a DOL-system, and let R be one of the relations $\leq$ ($\geq$) or $=$ on $\dot{N}^{|\Sigma|}$. Then

$$\exists n, m \in \dot{N} \quad : \pi(h^n(x))\; R\; \pi(h^{n+m}(x))$$

implies

$$\forall i, i \geq 0, \; \forall j, 0 \leq j < m : \pi(h^{n+mi+j}(x))\; R\; \pi(h^{n+m(i+1)+j}(x))$$

Proof

The proof is very simple using the fact that the entries in all π-values and the growth-matrix, M_H, of H are non-negative integers. This implies that for any words $z, z' \in \Sigma^*$:

$$\pi(z)\; R\; \pi(z') \quad \text{implies} \quad (\pi(z) \cdot M_H)\; R\; (\pi(z') \cdot M_H).$$

From this and the assumption of the lemma you get

$$\pi(h^n(x)) \quad R \quad \pi(h^{n+m}(x))$$

implies

$$\pi(h^n(x)) \cdot M_H^{mi+j} \quad R \quad \pi(h^{n+m}(x)) \cdot M_H^{mi+j}$$

iff

$$\pi(h^{n+mi+j}(x)) \quad R \quad \pi(h^{n+m(i+1)+j}(x))$$

Lemma 2.5

Let $H = \langle\Sigma, h, x\rangle$ be a DOL-system. Then

$$\exists n, m \in \dot{N} \; : \; \pi(h^n(x)) \geq \pi(h^{n+m}(x))$$

iff

$\mathcal{PL}(H)$ is finite.

Proof

Assume the existence of n and m, then the finiteness of $\mathcal{PL}(H)$ follows from lemma 2.4 and the observation that the entries in the π-values are non-negative integers.

The reverse implication is trivial.

Theorem 2.6

There exists an algorithm that decides for any DOL-system $H = <\Sigma, h, x>$ whether $\mathcal{PL}(H)$ is finite or not.

Proof

Compute the smallest integer n for which there exists an m such that $\pi(h^n(x))$ and $\pi(h^{n-m}(x))$ are comparable. It follows from [1] that any infinite sequence of π-values always contains at least two comparable elements, hence n is well-defined and computable.

Assume that $\pi(h^{n-m}(x)) \geq \pi(h^n(x))$. Then it follows from lemma 2.5 that $\mathcal{PL}(H)$ is finite.

If $\pi(h^{n-m}(x)) < \pi(h^n(x))$ then $\pi(h^{n+m(i-1)}(x)) \leq \pi(h^{n+mi}(x))$ for any $i \in \dot{N}$ (lemma 2.4). Define

$$\forall\, i \in \dot{N} : d_i = \pi(h^{n-mi}(x)) - \pi(h^{n-m(i-1)}(x)) \in \dot{N}^{|\Sigma|}.$$

Then it follows from lemma 2.5 that:

(*) $\mathcal{PL}(H)$ is finite iff d_{i_0} is equal to the zero-vector (the vector with all zero-entries) for some $i_0 \in \dot{N}$ (which implies that d_i is equal to the zero-vector for all $i \geq i_0$).

$$\begin{aligned} d_i &= \pi(x)\cdot M_H^{n+mi} - \pi(x)\cdot M_H^{n+m(i-1)} \\ &= (\pi(x)\cdot M_H^{n} - \pi(x)\cdot M_H^{n-m})\cdot M_H^{mi} \\ &= d_0 \cdot (M_H^m)^i, \end{aligned}$$

i.e., the sequence of vectors, d_i, is the $\mathcal{PS}$-value of the DOL-system $<\Sigma, h^m, x_0>$, where x_0 is some word from Σ^* for which $\pi(x_0) = d_0$. It then follows by arguments used previously that one can compute n_1, $m_1 \in \dot{N}$, $m_1 \geq 1$, such that $d_{n_1-m_1} \leq d_{n_1}$, which implies by lemma 2.4 that $d_{n_1+jm_1} \leq d_{n_1+(j+1)m_1}$ for any $j \in \dot{N}$. But then $\mathcal{PL}(H)$ is finite iff d_{n_1} is equal to the zero-vector (follows from (*) above).
(Theorem 2.6 could also have been proved using the theory of growth-functions.)

Since only a finite number of different words from Σ^* are associated (through π) with a single Parikh-vector, results similar to lemma 2.5 and theorem 2.6 hold for the $\mathcal{WL}$-operator. Furthermore, since only a finite number of Parikh-vectors are associated with words of a particular length, similar results also hold for the $\mathcal{NL}$-operator.

<u>Corollary 2.7</u>

Let H be a DOL-system. Then $\mathcal{WL}(H)$ ($\mathcal{NL}(H)$) is finite iff $\mathcal{PL}(H)$ is finite. Furthermore, lemma 2.5 and theorem 2.6 hold if the $\mathcal{PL}$-operator is replaced with the $\mathcal{WL}$-operator ($\mathcal{NL}$-operator).

Remark

Note that the algorithm given in theorem 2.6 is constructive in the sense that if $\mathcal{PL}(H)$ is finite and if n and m are the computed values for which $\pi(h^{n-m}(x)) = \pi(h^n(x))$, then (lemma 2.4) :

$$\mathcal{PL}(H) = \{\pi(h^i(x)) \mid 0 \leq i < n\}$$
$$(\mathcal{NL}(H) = \{\ |h^i(x)|\ \mid 0 \leq i < n\})$$

The corresponding constructive algorithm solving the finiteness-problem for the $\mathcal{WL}$-operator computes n and m as above and then continues computing the smallest i_1 for which there exists an $i_2 < i_1$ such that $h^{n+i_2 m}(x) = h^{n+i_1 m}(x)$ (i_1 is well-defined and computable). Then

$$\mathcal{WL}(H) = \{h^i(x) \mid 0 \leq i < n+i_1 m\}$$

In the following the term "a finite (infinite) DOL-system" refers to a system for which the $\mathcal{PL}$-value (and thereby the $\mathcal{WL}$- and the $\mathcal{NL}$-value) is a finite (infinite) set.

3. ON THE PARIKH-EQUIVALENCE PROBLEMS

In this section it is shown that the Parikh-sequence and the Parikh-language equivalence problems are recursively solvable for DOL-systems.

Theorem 3.1

The $\mathcal{PS}$-equivalence problem is recursively solvable for DOL-systems.

Proof

Follows immediately from lemma 2.3.

Theorem 3.2

The $\mathcal{PL}$-equivalence problem is recursively solvable for DOL-systems.

Proof

Let $H = <\Sigma, h, x>$ and $G = <\Sigma, g, y>$ be two DOL-systems over the same alphabet. First apply the algorithm of theorem 2.6 to H and G. If one or both of the systems turn out to be finite then $\mathcal{PL}$-equivalence is trivially decidable, since the algorithm then effectively constructs the finite set(s) of Parikh-vectors (see remark after corollary 2.7).

Assume now that both systems turn out to be infinite. The idea in the algorithm then applied is the following: based on finitely many generated vectors from $\mathcal{PL}(H)$ and $\mathcal{PL}(G)$ either 1) to state that $\mathcal{PL}(H) \neq \mathcal{PL}(G)$ or 2) split H and G into finitely many subsystems in such a way that the $\mathcal{PL}$-equivalence of H and G can be decided by applying the given algorithm to decide $\mathcal{PS}$-equivalence to given pairs of these systems.

The flow-diagram in figure 1 describes this algorithm. It consists of one main loop with two possible outcomes, steps 4 and 14, corresponding to 1) and 2) above respectively. Before the remarks to the essential steps of the algorithm are listed, two general remarks on any infinite DOL-system $H = \langle\Sigma, h, x\rangle$ should be noted:

R1. $\forall i, j \in \dot{N}, i \neq j$: $\pi(h^i(x)) \neq \pi(h^j(x))$ (follows from lemma 2.5)

R2. $\forall n, m, i, j \in \dot{N}$: $\pi(h^n(x)) < \pi(h^{n+m}(x))$ implies $\pi(h^{n+mi+j}(x)) < \pi(h^{n+m(i+1)+j}(x))$ (follows from R1 and lemma 2.4)

Remarks to the steps of the algorithm in figure 1.

Step 2:

As will be shown, the situation on entrance to step 2 is always:

$$\mathcal{PL}_0^{n_0}(H) = \mathcal{PL}_0^{n_0}(G), \text{ so}$$

$$\mathcal{PL}(H) = \mathcal{PL}(G) \text{ iff } \mathcal{PL}_{n_0}^{\infty}(H) = \mathcal{PL}_{n_0}^{\infty}(G)$$

(follows from R1), and each iteration of the main loop starting in step 2 is investigating whether the last equality holds or not.

The requirement is trivially met on the first entrance to step 2 with $n_0 = 0$.

Since $\mathcal{PL}_{n_0}^{\infty}(H)$ and $\mathcal{PL}_{n_0}^{\infty}(G)$ are infinite sets, the constants n_H, m_H, n_G and m_G are welldefined and computable (see proof of theorem 2.6 and R2).

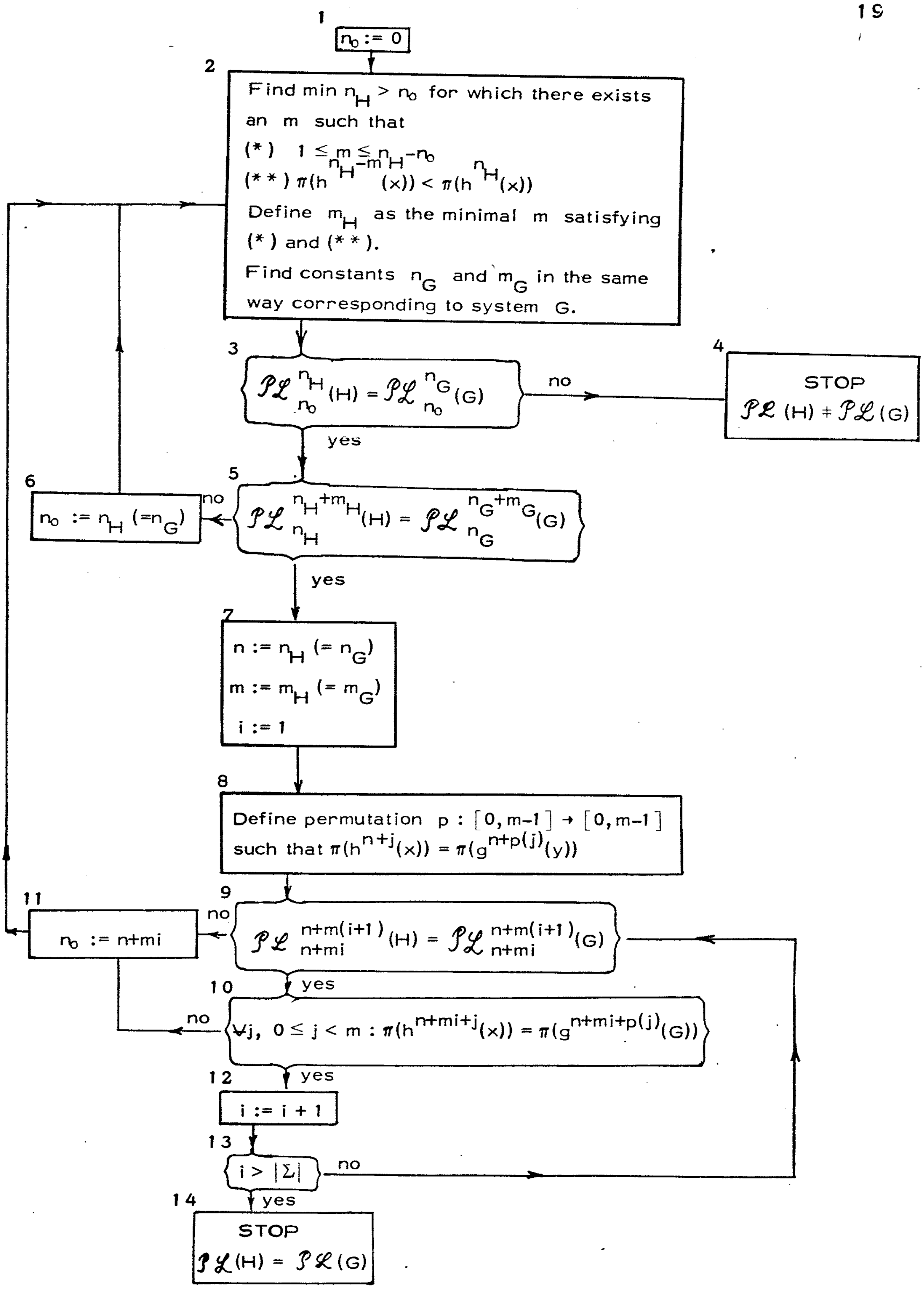
1
$n_0 := 0$
2
Find min $n_H > n_0$ for which there exists an m such that
(*) $1 \leq m \leq n_H - n_0$
(**) $\pi(h^{n_H - m}(x)) < \pi(h^{n_H}(x))$
Define m_H as the minimal m satisfying (*) and (**).
Find constants n_G and m_G in the same way corresponding to system G.
3
$\mathcal{PL}_{n_0}^{n_H}(H) = \mathcal{PL}_{n_0}^{n_G}(G)$
no
yes
4
STOP
$\mathcal{PL}(H) \neq \mathcal{PL}(G)$
5
$\mathcal{PL}_{n_H}^{n_H+m_H}(H) = \mathcal{PL}_{n_G}^{n_G+m_G}(G)$
no
yes
6
$n_0 := n_H \ (= n_G)$
7
$n := n_H \ (= n_G)$
$m := m_H \ (= m_G)$
$i := 1$
8
Define permutation $p : [0, m-1] \to [0, m-1]$ such that $\pi(h^{n+j}(x)) = \pi(g^{n+p(j)}(y))$
9
$\mathcal{PL}_{n+mi}^{n+m(i+1)}(H) = \mathcal{PL}_{n+mi}^{n+m(i+1)}(G)$
no
yes
11
$n_0 := n+mi$
10
$\forall j,\ 0 \leq j < m : \pi(h^{n+mi+j}(x)) = \pi(g^{n+mi+p(j)}(G))$
no
yes
12
$i := i + 1$
13
$i > |\Sigma|$
no
yes
14
STOP
$\mathcal{PL}(H) = \mathcal{PL}(G)$

Figure 1.

The algorithm of theorem 3. 2.

Step 3:

If the equality of step 3 does not hold, then it follows from R1 that there exists an i $(n_0 \leq i < n_H)$ such that $\pi(h^i(x)) \in \mathcal{PL}_{n_0}^{n_H}(H)$ and $\pi(h^i(x)) \notin \mathcal{PL}_{n_0}^{n_G}(G)$ (or the other way round). But then

$$\mathcal{PL}_{n_0}^{\infty}(H) = \mathcal{PL}_{n_0}^{\infty}(G)$$

implies

$$\exists\, j \geq n_G : \pi(h^i(x)) = \pi(g^j(y))$$

implies

$$\exists\, j', n_0 \leq j' \leq n_G : \pi(g^{j'}(y)) < \pi(g^j(y)) \qquad \text{(from R2)}$$

implies

$$\exists\, i', n_0 \leq i' < i : \pi(h^{i'}(x)) = \pi(g^{j'}(y))$$

implies

$$\exists\, i, i', n_0 \leq i' < i < n_H : \pi(h^{i'}(x)) < \pi(h^i(x))$$

which is a contradiction to the construction of n_H, i.e., if the equality of step 3 does not hold, then $\mathcal{PL}_{n_0}^{\infty}(H) \neq \mathcal{PL}_{n_0}^{\infty}(G)$, and H and G are not $\mathcal{PL}$-equivalent.

Notice that if the equality does hold, then $n_H = n_G$ (follows from R1).

Step 5:

Assume that the equality of step 5 does not hold, then the flow of the algorithm leads you back to step 2 after the assignment in step 6 $(n_0 := n_H\ (=n_G)$ - notice that after this assignment, the requirement $\mathcal{PL}_0^{n_0}(H) = \mathcal{PL}_0^{n_0}(G)$ holds on entrance to step 2). Let $n_H^{(k)}$, $m_H^{(k)}$, $n_G^{(k)}$ and $m_G^{(k)}$ denote the values defined at the k'th entrance of step 2, and let $n_0^{(k)}$ denote the value of n_0 in the k'th iteration of the main loop, $k \geq 1$. Assume that the equality of step 5 fails for values $m_H^{(k)}$ and $m_G^{(k)}$. Then it follows directly from step 6 that $n_0^{(k+1)} = n_H^{(k)}\ (=n_G^{(k)})$ and from R2 that

$$\pi(h^{n_H^{(k)}}(x)) < \pi(h^{n_H^{(k)} + m_H^{(k)}}(x))$$

which implies

$$n_H^{(k+1)} \leq n_H^{(k)} + m_H^{(k)}$$

and hence

$$m_H^{(k+1)} \leq n_H^{(k+1)} - n_0^{(k+1)} = n_H^{(k+1)} - n_H^{(k)} \leq m_H^{(k)}.$$

In the same way you get that $m_G^{(k+1)} \leq m_G^{(k)}$. If $m_H^{(k+1)} = m_H^{(k)}$ <u>and</u> $m_G^{(k+1)} = m_G^{(k)}$, and thereby $n_H^{(k+1)} = n_H^{(k)} + m_H^{(k)}$ <u>and</u> $n_G^{(k+1)} = n_G^{(k)} + m_G^{(k)}$, then it is directly verified that the equality of step 3 fails in the (k+1)st iteration of the main loop

$$\mathcal{PL}_{n_0^{(k+1)}}^{n_H^{(k+1)}}(H) = \mathcal{PL}_{n_H^{(k)}}^{n_H^{(k)}+m_H^{(k)}}(H)$$

$$\neq \mathcal{PL}_{n_G^{(k)}}^{n_G^{(k)}+m_G^{(k)}}(G) = \mathcal{PL}_{n_0^{(k+1)}}^{n_G^{(k+1)}}(G),$$

and the algorithm stops in step 4. Otherwise, $m_H^{(k+1)} < m_H^{(k)}$ <u>or</u> $m_G^{(k+1)} < m_G^{(k)}$.

Notice that if the equality of step 5 does hold, then $m_H = m_G$ (follows from R1), and the assignments of step 7 are welldefined. It also follows from R1 that the permutation p in step 8 is welldefined.

Step 9:

It will be shown that the situation on entrance to step 9 is always

(*) $\mathcal{PL}_n^{n+mi}(H) = \mathcal{PL}_n^{n+mi}(G)$

(**) $\forall k,j,\ 0 \leq k < i,\ 0 \leq j < m:\ \pi(h^{n+mk+j}(x)) = \pi(g^{n+mk+p(j)}(y))$

At entrance from step 8 these requirements are met from steps 5,7 and 8.

Assume that the equality of step 9 fails in the k'th iteration of the main loop, that is for values i, $n^{(k)}$ and $m^{(k)}$. The flow of the algorithm leads you back to a new iteration of the main loop, after the assignment of step 11 ($n_0^{(k+1)} = n^{(k)} + m^{(k)}i$), which ensures the condition on entrance to step 2 to hold (from (*) above and the equality of step 3)). From R2 it now follows that

$$\pi(h^{n^{(k)}+m^{(k)}i}(x)) < \pi(h^{n^{(k)}+m^{(k)}(i+1)}(x))$$

which implies that

$$n_H^{(k+1)} \leq n^{(k)} + m^{(k)}(i+1)$$

and hence

$$m_H^{(k+1)} \leq n_H^{(k+1)} - n_0^{(k+1)} = n_H^{(k+1)} - (n_H^{(k)} + m_H^{(k)}i)$$
$$\leq m_H^{(k)} = m^{(k)}.$$

In the same way you get that $m_G^{(k+1)} \leq m^{(k)}$. If $m_H^{(k+1)} = m_G^{(k+1)} = m^{(k)}$ and hence $n_H^{(k+1)} = n_G^{(k+1)} = n^{(k)} + m^{(k)}(i+1)$, then it is directly verified that the equality of step 3 fails in the (k+1)st iteration of the main loop, and the algorithm stops in step 4 (see the remarks to step 5 – notice in this connection that step 5 is equal to step 9 with $i = 0$.). Otherwise $m_H^{(k+1)} < m_H^{(k)}$ or $m_G^{(k+1)} < m_G^{(k)}$.

Step 11:

Assume that the requirements of step 11 are not fulfilled. Pick the greatest j for which the equality does not hold, then (from the requirement of step 9) there exists a j', $0 \leq j' < j$, such that

$$\pi(h^{n+mi+j}(x)) \neq \pi(g^{n+mi+p(j)}(y)) = \pi(h^{n+mi+j'}(x))$$

which implies

$$\pi(h^{n+mi}(x)) = \pi(x)\cdot M_H^{n+mi} = \pi(x)\cdot M_H^{n+m(i-1)+j} \cdot M_H^{m-j}$$
$$= (\pi(y)\cdot M_G^{n+m(i-1)+p(j)})\cdot M_H^{m-j} \quad \text{(from (**) in remarks to step 9)}$$
$$< (\pi(y)\cdot M_G^{n+mi+p(j)})\cdot M_H^{m-j} \quad \text{(from R2)}$$
$$= (\pi(x)\cdot M_H^{n+mi+j'})\cdot M_H^{m-j}$$
$$= \pi(x)\cdot M_H^{n+m(i+1)+j'-j}$$
$$= \pi(h^{n+mi+(m-(j-j'))}(x)),$$

where M_H and M_G are the growth-matrices of systems H and G respectively. Assume that the requirements fail for values $n^{(k)}$ and $m^{(k)}$, then the flow of the algorithm leads you back to a new iteration of the main loop with $n_0^{(k+1)} = n^{(k)} + m^{(k)}i$ (follows from the assignment of step 11). Now, from the above equations it follows immediately that

$$n_H^{(k+1)} \leq n^{(k)} + m^{(k)}i + (m^{(k)} - (j-j')) < n^{(k)} + m^{(k)}(i+1)$$

and hence

$$m_H^{(k+1)} \leq n_H^{(k+1)} - n_0^{(k+1)} = n_H^{(k+1)} - (n_H^{(k)} + m_H^{(k)}i)$$
$$< (n_H^{(k)} + m_H^{(k)}(i+1)) - (n_H^{(k)} + m_H^{(k)}i) = m_H^{(k)}.$$

From the remarks to step 9 it also follows that $m_G^{(k+1)} \leq m_G^{(k)}$.

Step 13:

If $i \leq |\Sigma|$ then the flow of the algorithm leads you back to step 9, and the equalities of steps 9 and 10 ensure that the requirements on entrance to step 9 mentioned above are met after the assignment of step 12. Obviously the algorithm will exit the small loop (from step 9 to step 13) after finitely many iterations (at most $|\Sigma|$). Exits to a new iteration of the main loop from steps 9 and 10 are treated above. If the algorithm exits the small loop from step 13 to step 14 then the situation is (follows from the remarks to step 9 and the equalities of step 10):

$$\forall i,j,\ 0 \leq i \leq |\Sigma|,\ 0 \leq j < m : \pi(h^{n+im+j}(x)) = \pi(g^{n+im+p(j)}(y))$$

Step 14:

On entrance to step 14 the following holds:

1) $\mathcal{PL}_0^n(H) = \mathcal{PL}_0^n(G)$ (follows from the requirements on entrance to step 2 and the equality of step 3).

2) For each j, $0 \leq j < m$ define systems

$H^j = <\Sigma, h^m, h^{n+j}(x)>$

$G^j = <\Sigma, g^m, g^{n+p(j)}(y)>$

then(from the remarks to step 13)

$\forall i,\ 0 \leq i \leq |\Sigma| : (h^m)^i\ (h^{n+j}(x)) = (g^m)^i\ (g^{n+p(j)}(y))$

Now an application of lemma 2.3 to systems H^j and G^j gives you $\mathcal{PS}(H^j) = \mathcal{PS}(G^j)$ and hence
$\mathcal{PL}(H^j) = \mathcal{PL}(G^j)$ for each j, $0 \leq j < m$.

3) By the construction of systems H^j and G^j above it follows that

$$\mathcal{PL}(H) = \mathcal{PL}_0^n(H) \cup \left(\bigcup_{j=0}^{m-1} \mathcal{PL}(H^j)\right)$$

$$\mathcal{PL}(G) = \mathcal{PL}_0^n(G) \cup \left(\bigcup_{j=0}^{m-1} \mathcal{PL}(G^j)\right)$$

But 1), 2), and 3) above imply that $\mathcal{PL}(H) = \mathcal{PL}(G)$.

From these remarks to the essential steps of the algorithm, it follows that for any k :

1) $m_H^{(k+1)} \leq m_H^{(k)}$ and $m_G^{(k+1)} \leq m_G^{(k)}$

2) if the algorithm does not stop in step 4 with a negative answer in the (k+1)st iteration of the main loop starting in step 2, then $m_H^{(k+1)} < m_H^{(k)}$ or $m_G^{(k+1)} < m_G^{(k)}$.

Now, since all the m_H- and m_G-values are natural numbers greater than or equal to one, then only finitely many iterations of the main loop are possible. From this one concludes that the algorithm stops after finitely many steps in either step 4 or step 14 with a negative or a positive answer to the question of $\mathcal{PL}$-equivalence respectively.
This completes the proof of theorem 3.2.

Example 3.3

Let H and G be the two DOL–systems defined as follows:

$$H = <\{a, b, c\}, h, a> \qquad G = <\{a, b, c\}, g, b>$$

where	where
$h(a) = b$	$g(a) = cba$
$h(b) = abc$	$g(b) = a$
$h(c) = ac$	$g(c) = cc$

The algorithm of figure 1 defines in the first iteration of the main loop: $n_H^{(1)} = n_G^{(1)} = 2$, $m_H^{(1)} = m_G^{(1)} = 1$. The answer to step 3 is positive since

$$\mathcal{PL}_0^2(H) = \mathcal{PL}_0^2(G) = \{(1, 0, 0), (0, 1, 0)\}$$

and so is the answer to step 5 since

$$\mathcal{PL}_2^3(H) = \mathcal{PL}_2^3(G) = \{(1, 1, 1)\}.$$

The permutation p defined in step 8 is the identity $(p(0) = 0)$. The answer to step 9 is negative since

$$\mathcal{PL}_3^4(H) = \{(2, 2, 2)\} \neq \mathcal{PL}_3^4(G) = \{(2, 1, 3)\}.$$

The algorithm then starts a new iteration of the main loop with values $n_0^{(2)} = 3$, $n_H^{(2)} = n_G^{(2)} = 4$, $m_H^{(2)} = m_G^{(2)} = 1$ and then stops with a negative answer to step 3 because of the inequality above.

Let $G' = <\{a, b, c\}, g', b>$, where $g'(a) = g(a)$, $g'(b) = g(b)$, but $g'(c) = cb$. The algorithm of figure 1 applied to systems H and G' gives a first iteration identical to the one described above for systems H and G, except that the answer to step 9 is now positive since

$$\mathcal{PL}_3^4(H) = \mathcal{PL}_3^4(G') = \{(2, 2, 2)\}.$$

The equality of step 10 is trivially fulfilled, and the algorithm now iterates the small loop from step 9 to step 13 twice, since

$$\mathcal{PL}_4^5(H) = \mathcal{PL}_4^5(G') = \{(4, 4, 4)\}$$

and

$$\mathcal{PL}_5^6(H) = \mathcal{PL}_5^6(G') = \{(8, 8, 8)\}$$

And then the algorithm stops in step 14 with a positive answer to the question of $\mathcal{PL}$-equivalence.

Definition 3.4

Let $H = <\Sigma, h, x>$ be a DOL-system. A symbol $\sigma \in \Sigma$ is called <u>useful</u> iff there exists an $i \in \mathbb{N}$ such that $\#_\sigma(h^i(x)) > 0$. H is called <u>reduced</u> iff any $\sigma \in \Sigma$ is useful.

Definition 3.5

Let $H = <\Sigma, h, x>$ be a DOL-system. Then P_H denotes the $|\Sigma| \times |\Sigma|$ matrix for which the (i, j)'th entry is equal to $\#_{\sigma_j}(h^{i-1}(x))$, $1 \leq i, j \leq |\Sigma|$, i.e., the rows of P_H are the vectors of $\mathcal{PL}_0^{|\Sigma|}(H)$.

Lemma 3.6

Let $H = <\Sigma, h, x>$ be a DOL-system. Then $\sigma \in \Sigma$ is a useful symbol iff the j'th column of P_H is not an all zero column.

Proof

Assume that the j'th column of P_H is not an all zero column, then σ_j is useful by definition.

Assume that the j'th column of P_H is an all zero column, then by lemma 2.1 any vector $\pi(h^i(x))$ is a linear combination of the rows of P_H, and hence σ_j is not a useful symbol.

Theorem 3.7

For any DOL-system $H = <\Sigma, h, x>$ there exists a reduced DOL-system $H' = <\Sigma', h', x'>$ such that $\mathcal{WS}(H) = \mathcal{WS}(H')$.

Proof

Define Σ' as the set of useful symbols of Σ, h' the restriction of h to Σ', and $x' = x$. Then H' is reduced and $\mathcal{WS}(H) = \mathcal{WS}(H')$.

Theorem 3.8

For any reduced DOL-system $H = <\Sigma, h, x>$ there exist only finitely many (reduced) DOL-systems $G = <\Sigma, g, y>$ such that $\mathcal{PL}(H) = \mathcal{PL}(G)$, and one can effectively construct a finite set of DOL-systems including all such systems, G, $\mathcal{PL}$-equivalent to H.

Proof

If H is finite (infinite) then define n as the smallest natural number for which there exists an m, $1 \leq m \leq n$, such that

$$\pi(h^n(x)) = \pi(h^{n-m}(x)) \quad (\pi(h^n(x)) > (\pi(h^{n-m}(x))).$$

Define q as $\max\{n, |\Sigma|+m\}$. Then it follows from the proofs of lemma 2.6 and theorem 3.2 that

$$\mathcal{PL}(H) = \mathcal{PL}(G)$$

implies

$$\mathcal{PL}_0^{|\Sigma|+1}(G) \subseteq \mathcal{PL}_0^{q}(H)$$

Define r as the maximal entry in the set of vectors $\mathcal{PL}_0^{q}(H)$, i.e.,

$$r = \max \{\#_{\sigma_j}(h^i(x)) \mid 1 \le j \le |\Sigma|, \ 0 \le i < q\}$$

Notice that by definition the rows of the matrix $P_G \cdot M_G$, where M_G is the growth matrix of system G, are all vectors from $\mathcal{PL}_0^{|\Sigma|+1}(G)$. Denote the entries of P_G and M_G by p_{ij} and m_{ij} respectively. Since H is reduced, then $\mathcal{PL}(H) = \mathcal{PL}(G)$ implies that G is also reduced, but then you get from lemma 2.6:

$$(*) \ \forall j, \exists i, \ 1 \le i, j \le |\Sigma| : \ p_{ij} \neq 0$$

Now from the observations above you also have

$$(**) \ \forall i, j, k, \ 1 \le i, j, k \le |\Sigma| : \ p_{ij} \cdot m_{jk} \le \sum_{l=1}^{|\Sigma|} p_{il} \cdot m_{lk} \le r$$

From $(*)$ and $(**)$ it follows that

$$\forall j, k, \exists i, \ 1 \le i, j, k \le |\Sigma| : p_{ij} \neq 0 \ \underline{\text{and}} \ m_{jk} \le \frac{r}{p_{ij}}$$

which implies that

$$\forall j, k, \ 1 \le j, k \le |\Sigma| : \ m_{jk} \le r$$

Thus the entries in any growth matrix M_G of a system G for which $\mathcal{PL}(H) = \mathcal{PL}(G)$ are bounded by the number r, and hence only a finite number of such growth matrices are possible. But now it is easily seen that for any $|\Sigma| \times |\Sigma|$ matrix M_G as above only finitely many mappings $g : \Sigma \to \Sigma^*$ exist such that

$$\forall i, j, 1 \leq i, j \leq |\Sigma| \ : \ \#_{\sigma_j}(g(\sigma_i)) = m_{ij} \quad .$$

Furthermore, it follows from the proofs of lemma 2.6 and theorem 3.2 that any axiom y of a system G for which $\mathcal{PL}(H) = \mathcal{PL}(G)$ satisfies $\pi(y) \in \mathcal{PL}_0^n(H)$. And obviously only a finite number of such words $y \in \Sigma^*$ exist.

This completes the proof of the first part of theorem 3.8. The second part follows almost immediately from the above proof.

The following corollary is an immediate consequence of theorem 3.8. The corollary can also be easily obtained using the theory of growth-functions [3].

Corollary 3.9

For any reduced DOL-system H there exist only finitely many (reduced) DOL-systems G such that $\mathcal{PG}(H) = \mathcal{PG}(G)$.

Theorem 3.10

Let H and G be any two $\mathcal{PG}$-equivalent DOL-systems, for which the constant k_H of system H defined in lemma 2.1 is equal to $|\Sigma|$. Then $M_H = M_G$ where M_H and M_G are the growth matrices of H and G respectively.

Proof

Define matrices P_H and P_G according to definition 3.5. Then the assumptions of the theorem imply that P_H is non-singular and

$P_H = P_G$. Furthermore, by definition it follows that

$$\mathcal{PS}(H) = \mathcal{PS}(G)$$

implies

$$P_H \cdot M_H = P_G \cdot M_G \ (= P_H \cdot M_G)$$

But since P_H is non-singular, P_H^{-1} exists, and thereby $M_H = M_G$.

From example 3.3 it follows that theorem 3.10 does not hold for the $\mathcal{PL}$-operator. The constant k_H for system H is 3, which is the cardinality of the alphabet, but system G' is a $\mathcal{PL}$-equivalent system for which $M_H \neq M_{G'}$.

4. ON THE WORD-EQUIVALENCE PROBLEMS

Theorem 4.1

If the $\mathcal{WS}$-equivalence problem is recursively solvable for DOL-systems, then so is the $\mathcal{WL}$-equivalence problem.

Proof

Let $H = <\Sigma, h, x>$ and $G = <\Sigma, g, y>$ be any two DOL-systems over the same alphabet. Assume that an algorithm to solve $\mathcal{WS}$-equivalence is given, then the following is an informal description of an algorithm to solve $\mathcal{WL}$-equivalence.

First apply the algorithm to decide finiteness to H and G (corollary 2.7). If one or both of the systems are finite then $\mathcal{WL}$-equivalence is trivially solvable (see remarks just after corollary 2.7).

If both of the systems turn out to be infinite, then notice that

1) $\mathcal{WL}(H) = \mathcal{WL}(G)$ implies $\mathcal{PL}(H)=\mathcal{PL}(G)$.
2) By remark R1 in the proof of theorem 3.2 there is a one-to-one correspondence between $\mathcal{PL}(H)$ and $\mathcal{WL}(H)$ and similarly for system G.

The idea is now simply to apply the algorithm of figure 1 to systems H and G.

a) If the algorithm stops in step 4, then $\mathcal{PL}(H) \neq \mathcal{PL}(G)$ and hence $\mathcal{WL}(H) \neq \mathcal{WL}(G)$ (see 1) above).

b) If the algorithm stops in step 14 then

i) define systems H^j and G^j as in the remarks to step 14 in the proof of theorem 3.2. Then it follows from 2) above that $\mathcal{WL}_0^n(H)$ and $\mathcal{WL}(H^j)$, $0 \le j < m$, are mutually disjoint sets for which

$$\mathcal{WL}(H) = \mathcal{WL}_0^n(H) \cup \left(\bigcup_{j=0}^{m-1} \mathcal{WL}(H^j)\right)$$

Similarly, $\mathcal{WL}_0^n(G)$ and $\mathcal{WL}(G^j)$, $0 \le j < m$, are mutually disjoint sets for which

$$\mathcal{WL}(G) = \mathcal{WL}_0^n(G) \cup \left(\bigcup_{j=0}^{m-1} \mathcal{WL}(G^j)\right)$$

ii) systems H^j and G^j, $0 \le j < m$, are $\mathcal{PL}$-equivalent, and from i) above and the observed one-to-one correspondence between $\mathcal{PL}$- and $\mathcal{WL}$-values it then follows that

$$\mathcal{WL}(H) = \mathcal{WL}(G) \text{ iff } \begin{cases} \mathcal{WL}_0^n(H) = \mathcal{WL}_0^n(G) \text{ and} \\ \mathcal{WL}(H^j) = \mathcal{WL}(G^j) \text{ for } 0 \le j < m, \end{cases}$$

iii) since the one-to-one correspondence between $\mathcal{PL}$- and $\mathcal{WL}$-values does also hold for the infinite systems H^j and G^j, $0 \le j < m$, and since H^j and G^j, $0 \le j < m$, are also known to be $\mathcal{PS}$-equivalent, then

$$\forall j,\ 0 \le j < m : \mathcal{WL}(H^j) = \mathcal{WL}(G^j) \text{ iff } \mathcal{WS}(H^j) = \mathcal{WS}(G^j).$$

The observations above under ii) and iii) imply that if the algorithm stops in step 14, then you can decide $\mathcal{WL}$-equivalence between systems H and G by generating and comparing the sets $\mathcal{WL}_0^n(H)$ and $\mathcal{WL}_0^n(G)$ and applying the given algorithm to solve $\mathcal{WS}$-equivalence to systems H^j and G^j, $0 \le j < m$.

This completes the proof of theorem 4.1.

Theorem 4.2

If the $\mathcal{WL}$-equivalence problem is recursively solvable for DOL-systems, then so is the $\mathcal{WS}$-equivalence problem.

Proof

Now assuming that an algorithm to solve $\mathcal{WL}$-equivalence is given, an algorithm to solve $\mathcal{WS}$-equivalence is to be constructed.

Let H and G be any two DOL-systems as in the proof of theorem 4.1.

First apply the algorithm to decide finiteness to systems H and G.

1) If both systems are finite, then compute the minimum n_H (n_G) for which there exists an m_H (m_G) $1 \leq m_H \leq n_H$ $(1 \leq m_G \leq n_G)$ such that

$$h^{n_H}(x) = h^{n_H - m_H}(x)$$

$$(g^{n_G}(y) = g^{n_G - m_G}(y))$$

Then clearly

$$\forall i,j,\ 0 \leq i,\ 0 \leq j < m_H : h^{n_H + m_H i + j}(x) = h^{n_H - m_H + j}(x)$$

$$(\forall i,j,\ 0 \leq i,\ 0 \leq j < m_G : g^{n_G + m_G i + j}(y) = g^{n_G - m_G + j}(y))$$

which implies that $\mathcal{WS}(H) = \mathcal{WS}(G)$ iff

$$n_H = n_G$$

and

$$\forall i,\ 0 \leq i < n_H = n_G : h^i(x) = g^i(y).$$

2) If one of the systems is finite and the other infinite then clearly $\mathcal{WS}(H) \neq \mathcal{WS}(G)$.

3) If both systems are infinite then

a) apply the given algorithm to decide $\mathcal{WL}$-equivalence to H and G. If $\mathcal{WL}(H) \neq \mathcal{WL}(G)$ then $\mathcal{WS}(H) \neq \mathcal{WS}(G)$. Otherwise

b) using the one-to-one correspondence between the $\mathcal{PL}$-value and the $\mathcal{WL}$-value of infinite DOL-systems mentioned in the proof of theorem 4.1, one gets that for the infinite, $\mathcal{WL}$-equivalent DOL-systems H and G:

$$\mathcal{WS}(H) = \mathcal{WS}(G) \text{ iff } \mathcal{PS}(H) = \mathcal{PS}(G)$$

And by theorem 3.1 the Parikh-sequence equivalence problem is solvable.

This completes the proof of theorem 4.2.

<u>Remark</u>

In the case of 3.b) in the proof of theorem 4.2 one might also have applied the algorithm of figure 1 to systems H and G. Since H and G are known to be $\mathcal{WL}$-equivalent (and thereby $\mathcal{PL}$-equivalent) in case 3.b), the algorithm would stop in step 14. Furthermore, the systems H^j and G^j defined in the remarks to step 14 in the proof of theorem 3.2 satisfy $\mathcal{WS}(H^j) = \mathcal{WS}(G^j)$, $0 \leq j < m$ (see proof of theorem 4.1.).

But then

$$\mathcal{WS}(H) = \mathcal{WS}(G)$$

iff

$$(*)\ \forall i,\ 0 \leq i < n+m\ :\ h^i(x) = g^i(y).$$

This is proven by the following immediate implications of (*):

1) $\mathcal{PS}_0^n(H) = \mathcal{PS}_0^n(G)$

2) The permutation p defined in step 8 in the last iteration of the main loop of the algorithm of figure 1 is the identity.

From theorems 4.1 and 4.2 it follows that $\mathcal{WL}$-equivalence is recursively solvable iff $\mathcal{WS}$-equivalence is recursively solvable. Unfortunately, it is still an open problem whether any of the equivalence problems are actually solvable, as a matter of fact the solvability of the $\mathcal{WL}$-equivalence is one of the most outstanding open problems in the area of L-systems. However, the proof of theorem 4.1 suggests some results that intuitively seem to simplify the $\mathcal{WL}$-equivalence problem.

Definition 4.3

Let $H = <\Sigma, h, x>$ be a DOL-system. For any $z \in \Sigma^*$ let min(z) denote the subset of Σ consisting of exactly the symbols from Σ occurring in z. H is said to be conservative iff $\min(h^i(x)) = \Sigma$ for every $i \geq 0$, i.e., iff any symbol from Σ occurs in any word generated by H.

Theorem 4.4

The $\mathcal{WL}$-equivalence problem is recursively solvable for DOL-systems iff it is recursively solvable for conservative DOL-systems.

Proof

Assume that an algorithm to solve $\mathcal{WL}$-equivalence for conservative systems is given (the reverse implication is trivial) and that H and G are infinite (otherwise the theorem is also trivial) DOL-systems. Define n_H as the smallest natural number for which there exists an m_H, $1 \leq m_H \leq n_H$ such that

$$\min(h^{n_H}(x)) = \min(h^{n_H - m_H}(x)).$$

Then clearly

$$\forall i,j,\ i \geq 0,\ 0 \leq j < m^H : \min(h^{n_H + m_H i + j}(x)) = \min(h^{n_H - m_H + j}(x))$$

which means that

$$H^j = <\min(h^{n_H + j}(x)), h_j^{m_H}, h^{n_H - m_H + j}(x)> ,$$

where $h_j^{n_H}$ is the restriction of h^{n_H} to $\min(h^{n_H + j}(x))$, are well-defined conservative DOL-systems for $0 \leq j < m_H$. Define n_G, m_G and systems G^j $(0 \leq j < m_G)$ similarly for system G. By definition $\mathcal{WL}(H^j)$ ($\mathcal{WL}(G^j)$) consists of precisely the infinite set of words from $\mathcal{WL}$(H) ($\mathcal{WL}$(G)) over the alphabet $\min(h^{n_H + j}(x))$ $(\min(g^{n_G + j}(y))$. But then $\mathcal{WL}(H) = \mathcal{WL}(G)$ iff

1) $n_H = n_G$ (= n) and $m_G = m_H$(= m)

2) $\mathcal{WL}_0^{n-m}(H) = \mathcal{WL}_0^{n-m}(G)$

3) $\{\min(h^{n+j}(x)) \mid 0 \leq j < m\} = \{\min(g^{n+j}(y)) \mid 0 \leq j < m\}$

4) $\forall j,\ 0 \leq j < m : \mathcal{WL}(H^j) = \mathcal{WL}(G^{p(j)})$ where p is the permutation $p : [0, m-1] \to [0, m-1]$ for which $\min(h^{n+j}(x)) = \min(g^{n+p(j)}(y))$. p is known to exist from 3 and the construction of H^j and G^j.

But 1-4 are easily checked (assuming that an algorithm to solve $\mathcal{WL}$-equivalence for conservative DOL-systems is given). This completes the proof of theorem 4.4.

Definition 4.5

Let $H = \langle \Sigma, h, x \rangle$ be a DOL-system. H is said to be growing iff $\pi(h^i(x)) < \pi(h^{i+1}(x))$ for every $i \geq 0$.

Remark

Theorem 4.4 could have been established more directly by theorem 4.1, but the construction in the above proof indicates as a matter of fact a much easier way of proving that the $\mathcal{WL}$-equivalence problem is recursively solvable for PDOL-systems iff the $\mathcal{WS}$-equivalence problem is recursively solvable for PDOL-systems. (If H and G are infinite PDOL-systems then all the constructed subsystems H^j and G^j will be growing and hence these subsystems are $\mathcal{WL}$-equivalent iff they are $\mathcal{WS}$-equivalent.)

From theorem 4.4 and the proof of theorem 4.1 the following is an obvious corollary, intuitively simplifying the $\mathcal{WL}$-equivalence problem for DOL-systems a great deal:

Corollary 4.6

The $\mathcal{WL}$-equivalence problem is recursively solvable for DOL-systems iff the $\mathcal{WS}$-equivalence problem is recursively solvable for conservative, growing, $\mathcal{PS}$-equivalent DOL-systems.

Although corollary 4.5 seems to simplify the $\mathcal{WL}$-equivalence problem, the $\mathcal{WS}$-equivalence problem is not at all trivial, not even for conservative, growing, $\mathcal{PS}$-equivalent DOL-systems. It seems likely that a result similar to lemma 2.3 would hold also for the words generated by two DOL-systems, i.e., that a constant k_Σ exists, depending on the cardinality of the alphabet of the systems, such that:

$$\forall i,\ 0 \le i \le k_\Sigma \ :\ h^i(x) = g^i(y)$$

(*) iff

$$\forall i,\ 0 \le i \qquad :\ h^i(x) = g^i(y).$$

It can be shown that for any two systems over the alphabet $\Sigma = \{a, b\}$ with axioms $x = y = ab$, (*) above holds for $k_\Sigma = 3$ (but unfortunately the arguments are very difficult to generalize). That 3 is a lower bound for k_Σ in this case is seen from the following example:

$H = <\{a, b\}, h, ab>$	$G = <\{a, b\}, g, ab>$
where	where
$h(a) = abb$	$g(a) = abbaabb$
$h(b) = aabba$	$g(b) = a$

For these systems:

$$\forall i,\ 0 \le i \le 2 \ :\ h^i(ab) = g^i(ab)$$

and

$$h^3(ab) \neq g^3(ab)$$

From the results of section 2 one gets some corollaries, stating results about $\mathcal{WS}$- and $\mathcal{WL}$-equivalence.

Corollary 4.7

The $\mathcal{WL}$- and the $\mathcal{WS}$-equivalence problems are recursively solvable for DOL-systems over a one-letter alphabet.

Actually, the result of corollary 4.7 is rather trivial in itself.

Corollary 4.8

For any reduced DOL-system H there exist only finitely many (reduced) $\mathcal{WL}$-($\mathcal{WS}$-)equivalent DOL-systems, and one can effectively construct a finite set of DOL-systems including all systems $\mathcal{WL}$-($\mathcal{WS}$-)equivalent to H.

Proof

Follows from theorem 3.8 and the fact that $\mathcal{WL}$-($\mathcal{WS}$-)equivalence implies $\mathcal{PL}$-equivalence.

Theorem 4.9

Let H and G be any two reduced $\mathcal{WS}$-equivalent DOL-systems, and let m_{ij}^{H} and m_{ij}^{G} denote the entries in the growth-matrices M_H and M_G of systems H and G respectively. Then

$$\forall i,\ 1 \leq i \leq |\Sigma| \ : \sum_{j=1}^{|\Sigma|} m_{ij}^{H} = \sum_{j=1}^{|\Sigma|} m_{ij}^{G}$$

implies

H = G.

Proof

The alphabets and the axioms respectively of two $\mathcal{WS}$-equivalent DOL-systems are identical by definition. So, to prove the lemma it is sufficient to prove that $h = g$, where h and g are the homomorphisms of the two systems.

Let σ be any symbol from Σ. Since H and G are reduced, there exists a j, $0 \leq j < |\Sigma|$, such that σ is occurring in $h^j(x) = g^j(y) = z_1 \sigma z_2$. But then

$$(*) \quad h(z_1)\, h(\sigma)\, h(z_2) = h(z_1 \sigma z_2) = h^{j+1}(x) = g^{j+1}(y) = g(z_1 \sigma z_2)$$
$$= g(z_1)\, g(\sigma)\, g(z_2)$$

The assumptions of the theorem imply that $|h(\sigma)| = |g(\sigma)|$ for any symbol $\sigma \in \Sigma$ and thereby

$$(**) \quad |h(z_1)| = |g(z_1)|, \ |h(\sigma)| = |g(\sigma)|, \ |h(z_2)| = |g(z_2)|$$

Now $(*)$ and $(**)$ imply that $h(\sigma) = g(\sigma)$, and this completes the proof of theorem 4.9.

Corollary 4.10

Let H and G be any two $\mathcal{WS}$-equivalent DOL-systems, for which the constant k_H of system H defined in lemma 2.1 is equal to $|\Sigma|$. Then $H = G$.

Proof

Since $\mathcal{WS}$-equivalence implies $\mathcal{TS}$-equivalence it follows from theorem 3.10 that $M_H = M_G$, where M_H and M_G are the growth-matrices of of systems H and G.

$k_H = |\Sigma|$ implies that the matrix P_H defined in definition 3.5 is non-singular, which implies (lemma 3.6) that H (and thereby G) is reduced.

Thus, the assumptions of theorem 4.9 are fulfilled for systems H and G, and therefore H = G.

Corollary 3.9, Theorem 3.10 and corollary 4.10 are also consequences of results in [3].

From the following trivial example it is seen that corollary 4.10 does not hold if H and G are only assumed to be $\mathcal{WL}$-equivalent:

$H = < \{a, b\}, h, a>$	$G = < \{a, b\}, g, b>$
where	where
$h(a) = b$	$g(a) = a$
$h(b) = b$	$g(b) = a$

Obviously, $H \neq G$, $\mathcal{WL}(H) = \mathcal{WL}(G) = \{a, b\}$, and the matrix P_H defined in definition 3.5 is equal to

$$P_H = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix},$$

i.e., the constant k_H of lemma 2.1 is equal to 2, which is the cardinality of the alphabet of the systems.

5. ON THE LENGTH-EQUIVALENCE PROBLEMS

Theorem 5.1

The $\mathcal{NS}$-equivalence problem is recursively solvable for DOL-systems.

Proof

The reader is referred to [4] for a complete proof of this theorem.

One might think that the idea of the algorithm in figure 3.1 would carry over and establish a proof of the decidability of the $\mathcal{NL}$-equivalence problem for DOL-systems by means of theorem 5.1. Unfortunately, this is not the case. The main reason is that there need not be any relation between the number of useful symbols in two length-equivalent DOL-systems, contrary to the case of Parikh- and word-equivalence.

It is possible, however, to show that the $\mathcal{NL}$-equivalence problem is solvable for PDOL-systems, using the result of theorem 5.1.

Definition 5.2

Let Σ be a finite alphabet and z any string from Σ^*, $z = \sigma_1 \sigma_2 \dots \sigma_n$, $\sigma_i \in \Sigma$. Then define

$$\forall i,j,\ 1 \le i \le j \le n : [z]_i^j = \sigma_i \sigma_{i+1} \dots \sigma_j$$

Definition 5.3

A DOL-system $H = <\Sigma, h, x>$ is called length-growing iff

$$\forall i,\ i \ge 0 : |h^i(x)| < |h^{i+1}(x)|.$$

Lemma 5.4

For any infinite PDOL-system $H = <\Sigma^H, h, x>$ there exists a PDOL-system $G = <\Sigma^G, g, y>$ such that

1) $\mathcal{NL}(H) = \mathcal{NL}(G)$

2) G is length-growing.

Proof

Define n as the smallest natural number for which there exists an m such that

$$\min(h^n(x)) = \min(h^{n-m}(x)).$$

As in the proof of theorem 4.4, n and m are well-defined and computable, and

$$\forall i, j,\ i \geq 0,\ 0 \leq j < m : \min(h^{n+mi+j}(x)) = \min(h^{n-m+j}(x)).$$

Furthermore, the following is an immediate consequence of the fact that H is a propagating DOL-system:

$$|h^{n-m+j}(x)| = |h^{n-m+j+1}(x)| \quad \text{for } j,\ 0 \leq j < m$$

(*) iff

$$\forall i,\ i \geq 0 \quad : |h^{n+mi+j}(x)| = |h^{n+mi+j+1}(x)|.$$

Let c_k, $0 \leq k < m$, denote the number of different symbols from Σ^H in $\min(h^{n-m+k}(x))$. Define m mutually disjoint, finite sets of symbols (all of them disjoint from Σ^H):

$$\forall k,\ 0 \leq k < m : \Gamma_k = \{\gamma_{kj} \mid 1 \leq j \leq c_k\},$$

and define $\Gamma = \bigcup_{k=0}^{m-1} \Gamma_k$. Finally, define m isomorphisms (with respect to concatenation), one for each Γ_k:

$$\forall k,\ 0 \le k < m : \varphi_k : \min(h^{n-m+k}(x)) \rightarrow \Gamma_k$$

where

$$\forall \sigma \in \min(h^{n-m+k}(x)) : \varphi_k(\sigma) = \gamma_{kj}$$

iff

σ is the j'th symbol of $\min(h^{n-m+k}(x))$

Notice that the φ_k's are defined for some fixed enumerations of the sets $\min(h^{n-m+k}(x))$.

Now, system G is going to be constructed. Γ will be a subset of the alphabet Σ^G of system G, and the homomorphism of G, g, is defined on Γ as follows.

Define the sequence $k_1, k_2, \ldots, k_r$ of natural numbers satisfying:

1) $0 \le k_1 < k_2 < \ldots < k_r < m$
2) $\forall i,\ 1 \le i \le r : |h^{n-m+k_i}(x)| < |h^{n-m+k_i+1}(x)|$
3) $\forall k,\ 0 \le k < m : \forall i,\ 1 \le i \le r : k \neq k_i$

implies

$|h^{n-m+k}(x)| = |h^{n-m+k+1}(x)|$

Since H is propagating, the sequence $k_1, k_2, \ldots, k_r$ is well-defined and computable, and since H is infinite, the sequence is non-empty (follows from (*) above).

Now define:

a) $\forall i,\ 1 \leq i < r\ \forall j,\ 1 \leq j \leq c_{k_i}$: $g(\gamma_{k_i j}) = \varphi_{k_{i+1}}(h^{k_{i+1}-k_i}(\varphi_{k_i}^{-1}(\gamma_{k_i j})))$

b) $\forall j,\ 1 \leq j \leq c_{k_r}$: $g(\gamma_{k_r j}) = \varphi_{k_1}(h^{m-k_r+k_1}(\varphi_{k_r}^{-1}(\gamma_{k_r j})))$

Note that the use of φ_{k_1} is well-defined since

$$\min(h^{n-m+k_r+(m-k_r+k_1)}(x)) = \min(h^{n+k_1}(x)) = \min(h^{n-m+k_1}(x))$$

c) For any $\gamma_{kj} \in \Gamma$ for which g has not been defined according to a) and b) above (k is not in the sequence $k_1, k_2, \ldots, k_r$) define

$$g(\gamma_{kj}) = \gamma_{kj}.$$

As a matter of fact $g(\gamma_{kj})$ can be defined at random, since γ_{kj} will not be a useful symbol in G.

Let $l_1, l_2, \ldots, l_p$ be the set of different elements from $\mathcal{NL}_0^{n-m}(H)$ for which $l_1 < l_2 \ldots < l_p < l = |h^{n-m}(x)| = |h^{n-m+k_1}(x)|$.

Introduce a new set of symbols

$$\Delta = \{\delta_{ij} \mid 1 \leq i \leq p,\ 1 \leq j \leq l_i\}$$

which is also going to be a subset of Σ^G.

Define

$$\forall i,\ 1 \leq i < p\ \forall j,\ 1 \leq j < l_i : g(\delta_{ij}) = \delta_{i+1,j}$$

$$\forall i,\ 1 \leq i < p \qquad : g(\delta_{i l_i}) = \delta_{i+1, l_i}\, \delta_{i+1, l_i+1} \cdots \delta_{i+1, l_{i+1}}$$

$$\forall j,\ 1 \leq j < l_p \qquad : g(\delta_{pj}) = \varphi_{k_1}([h^{n-m+k_1}(x)]_j^j)$$

$$g(\delta_{p l_p}) = \varphi_{k_1}([h^{n-m+k_1}(x)]_{l_p}^{l})$$

If y is defined as the string $\delta_{11}\,\delta_{12} \cdots \delta_{1 l_1}$ from Δ^+ then $G = <\Gamma \cup \Delta,\ g,\ y>$ is a PDOL-system satisfying 1) and 2) in the lemma. This completes the proof of lemma 5.4.

Theorem 5.5

The $\mathcal{NL}$-equivalence problem is recursively solvable for PDOL-systems.

Proof

Let H and G be any two PDOL-systems over the same alphabet. If one or both of the systems are finite, then the $\mathcal{NL}$-equivalence problem is trivially solvable.

If both systems are infinite, then construct length-growing systems H' and G' according to lemma 5.4 (note that the proof is constructive) such that $\mathcal{NL}(H) = \mathcal{NL}(H')$ and $\mathcal{NL}(G) = \mathcal{NL}(G')$. Then

$$\mathcal{NL}(H) = \mathcal{NL}(G)$$

iff

$$\mathcal{NL}(H') = \mathcal{NL}(G')$$

iff (since H' and G' are length-growing)

$$\mathcal{NS}(H') = \mathcal{NS}(G')$$

which is decidable by theorem 5.1 (nothing is known about the alphabets of H' and G' - see the remark in section 1 about this problem).

Acknowledgement

The author is grateful to the group of people in Aarhus working on L-systems, especially to Arto Salomaa for valuable help during the preparation of this paper, and to Jean Berstel who made some useful comments on an earlier version of the paper.

6. REFERENCES

[1] Konig, D., Theorie der endlichen und unendlichen Graphen, Chelsea, New York (1959).

[2] Lindenmayer, A., Mathematical models for cellular interactions in development, Parts I and II, J. Theoret. Biol., 18, 1968, 280-315.

[3] Paz, A., Similarity in DTOL and related problems, Technical Report no. 15, May 1973, SUNY at Stony Brook, Dept. of Computer Science.

[4] Paz, A., and A. Salomaa, Integral sequential word functions and growth equivalence of Lindenmayer systems. Information and Control, vol. 23, no. 4, 1973, 313-344.

[5] Rozenberg, G., Direct proofs of the undecidability of the equivalence problem for sentential forms of linear context-free grammars and the equivalence problem for OL-systems, Information Processing Letters, vol. 1, no. 6, 1972, 233-236.

[6] Rozenberg, G., The equivalence problem for deterministic TOL-systems is undecidable, Information Processing Letters, vol. 1, no. 5, 1972, 201-205.

[7] Salomaa, A., On sentential forms of context-free grammars, Acta Informatica, vol. 2, fasc. 1, 1973, 40-50.

[8] Salomaa, A., Formal Languages, Academic Press, 1973.