STRUCTURED OL-SYSTEMS

K. ČULÍK II

Department of Applied Analysis

and Computer Science

University of Waterloo

Waterloo, Ontario, Canada

and

Geselschaft für Mathematik und Datenverarbeitung Bonn

ABSTRACT A new type of Lindenmayer systems, called Structured OL-systems (SOL systems), is studied which gives a formal tool for investigation of structured organisms and structurally dependent developments. It is shown that a restricted version of SOL-systems is equivalent to codings (length-preserving homomorphisms) of OL-Languages. The properties of unrestricted SOL-systems are then studied. It is for example shown that the languages generated by them properly include the languages generated by extended table OL-Systems.

1. Introduction.

Lindenmayer systems have been the object of extensive study during recent years. The systems, also called developmental systems, were introduced in connection with a theory proposed to model the development of filamentous organisms. The stages of development are represented by strings of symbols correspondings to states of individual cells of an organism. The developmental instructions are modelled by grammar-like productions. These productions are applied simultaneously to all symbols to reflect the simultanity of the growth in the organism. This parallelism is the main difference between Lindenmayer systems and ordinary generative grammars. Another difference is that in most of the versions of Lindenmayer systems only one type of symbols (terminals) is considered which means that all the intermediate strings in a derivation are

strings in the generated language. The simplest type of Lindenmayer systems are the OL-Systems [61] in which every symbols is rewritten independently of its neighbours.

When we compare OL-systems and the corresponding class of grammars, context-free grammars, we see that OL-systems are missing one important feature of context-free grammars, namely they are not structuring the generated strings. A derivation in a context-free grammar can be represented as a derivation-tree which describes the structure of a string with respect to this derivation. We can consider the analogous derivation tree for a derivation in an OL-system but in this case the branching nodes are labelled by terminal symbols rather than nonterminals (grammatical categories) and the tree does not reflect the possibly interesting structure of an organism.

In this paper we introduce Structured OL systems (SOL-systems) which not only allow to describe the structure of generated strings but also give a formal tool to study the cases when the development is structurally dependent. A simple example is the case when all the stages of a developing organism consist of certain fixed number of partes and there are different development rules for every part of the organism. From a mathematical point of view the SOL-systems give another interesting type of context-sensitivity in parallel rewriting (compare with |CO|†). A structured organism is represented in an SOL-system as a labelled tree. The labels of the leaves of the tree represent the individual cells of an organism, the labels of its branching nodes represent the structural "units" of the organism.

An SOL-system is given by a single starting structure and a finite number of developmental rules. At every stage of the development the rules are applied simultaneously to all cells and structural units (nonterminals) of an organism. According the rules each structural unit can change its state or disappear (but not divide) and every cell can be replaced by a substructure, i.e. it can divide into several parts each of which can be either a single cell or another structured part. At every step a cell can divide only into a limited number of subparts but there is generally no limit on the number of subparts of a structural unit which can be created during the

---

† |CO| : K. Čulik II and J. Opatrný, Context in parallel rewriting, in this volume.

development, i.e. there is generally no limit on the number of sons of a branching node.

Types of structural units will be represented by labels from an alphabet which corresponds to the nonterminals of a context-free grammar. We will define SOL-systems in such a way that rewriting of a nonterminal may depend on its father but not on its sons in a tree. We consider parallel rewriting; at every step of a derivation all labels in a tree must be simultaneously rewritten. The language generated by an SOL-system is the set of all frontiers of the generated trees.

We will consider a special subfamily of SOL-systems, called simple SOL-systems, in which essentially only labels on leaves (individual cells) are rewritten, i.e. in a simple SOL-system it is possible to create new structural units but once a unit is created it never changes its state. We will show that the languages generated by simple SOL-systems are exactly length preserving homomorphisms (coding) of OL-languages [10,20]. Then we will investigate the properties of the family generated by unrestricted SOL-systems (SOL). We will show that SOL is closed under Kleene operation ($\cup$,.,*) and under ε-free homomorphism but not under intersection with a regular set.

The closure result will help us to establish the relations of SOL to other known families of languages. It is easy to show that SOL $\supsetneq$ TOL [81] and then, using the closure results, that SOL $\subsetneq$ ETOL [89]. The fact that the later inclusion and therefore, of course, also the former is proper follows from the result that SOL is incomparable with the family of context-sensitive languages. Actually it will be shown that any recursively enumerable set over T with an end marker can be expressed as an intersection of an SOL language over an extended alphabet with the set of all the terminal strings with the endmarker.

2. Preliminaries.

We assume the knowledge of the basic notions and notation of formal language theory, see e.g. [HU^{+},1o2]. We start with a slightly modified, but equivalent, definition of extended table OL-systems [89], involving as special cases OL-systems [61] EOL-systems [35] and TOL-systems [81].

Definition: An extended table L-system without interaction (ETOL system) is a 4-tuple $G = (V,T,\mathcal{P},\sigma)$ where

(i) V is a finite nonempty set, the alphabet of G,

(ii) $T \subseteq V$, the terminal alphabet of G,

(iii) $\mathcal{P}$ is a finite set of tables. $\mathcal{P} = \{P_1,\ldots,P_n\}$ for some $n \geq 1$, where each $P_i \subseteq V \times V^*$. Element (u,v) of P_i, $1 \leq i \leq n$, is called a production and is usually written in the form $u \to v$. Every P_i, $1 \leq i \leq n$, satisfies the following (completness) condition: For each $a \in V$ there is $w \in V^*$ so that $(a,w) \in P_i$,

(iv) $\sigma \in V^+$, the axiom of G.

Definition: An ETOL-system $G = (V,T,\mathcal{P},\sigma)$ is called

(i) a TOL-system if $V = T$;

(ii) an EOL-system if $\mathcal{P} = \{P_1\}$;

(iii) an OL-system if $V = T$ and $\mathcal{P} = \{P_1\}$.

Definition: Given an ETOL-system $G = (V,T,P,\sigma)$ we write $x \underset{G}{\Rightarrow} y$ if there exist $a_1,\ldots,a_k \in V$ and $y_1,\ldots,y_k \in V^*$ so that, $x = a_1\ldots a_k$, $y = y_1\ldots y_k$ and for some $P_i \in \mathcal{P}$, $a_j \to y_j \in P_i$, $j = 1,\ldots,k$.

The transitive and reflexive closure of binary relation $\underset{G}{\Rightarrow}$ is denoted by $\underset{G}{\overset{*}{\Rightarrow}}$.

Definition: Let $G = (V,T,\mathcal{P},\sigma)$ be an ETOL system. The language generated by G is denoted by $L(G)$ and defined as $L(G) = \{w \in T^* : \sigma \underset{G}{\overset{*}{\Rightarrow}} w\}$.

Notation: A language generated by an XYZ system, for any type XYZ will be called an XYZ-language. The family of all the XYZ languages is denoted by XYZ.

---

† |HU|: J.E. Hopcroft and J.D. Ullman: Formal Languages and their Relation to Automata, Addison-Wesley, 1969.

Before we can define SOL-systems we need to introduce a notation for labelled trees (forests). We will recursively define labelled rooted ordered forests and expressions denoting them. We are not interested in names of particular nodes in a forest, i.e. we actually consider the equivalence classes of isomorphic forests. We consider forests with labels of leaves from one alphabet and labels of branch nodes (nonleaves) from another distinct alphabet.

Definition: Let T,N be two alphabets, $T \cap N = \emptyset$, and let $[\ ,\ ]$ and λ be reserved symbols not in $T \cup N$.

(i) λ is a forest expression and denotes the empty tree (forest), i.e. the tree with no nodes.

(ii) For $a \in T$, a is a forest expression and denotes the tree with a single node (root)labelled by a.

(iii) If e_1, e_2 are forest expressions denoting nonempty forests α , β consisting from trees $\alpha_1,\ldots,\alpha_m$ and $\beta_1,\ldots,\beta_n$, respectively, then e_1e_2 is a forest expression and denotes the forest consisting from trees $\alpha_1,\ldots,\alpha_m,\beta_1,\ldots,\beta_n$.

(iv) If $A \in N$ and e is a forest expression denoting nonempty forest α consisting of trees $\alpha_1,\ldots,\alpha_n$ then $A[e]$ is a forest expression and denotes the tree with the root labelled by A and the sons of the root, from left to right, the the roots of subtrees $\alpha_1,\ldots,\alpha_n$.

Notation. The set of all the forest expressions (forests) over alphabets N (labels of branch nodes) and T (labels of leaves) is denoted by $(N,T)_*$. Let $(N,T)_+ = (N,T)_* - \{\lambda\}$. In the following the elements of $(N,T)_*$ will be called structures (over N,T) and we will not distinguish between an expression and the forests denoted by it.

Notation. Let a be a particular occurence of symbol a in forest expression α. The label at the father of the node labelled by the considered occurence of a will be denoted by $\text{Father}_\alpha(a)$. This notation will be used in such a way that there will be no confusion of which occurence of a is being considered.

Notation. Mapping Y (yield) maps λ to empty string (denoted by ε) and a forest in $(N,T)_+$ to the string of the labels of its leaves (from left to right).

3. Structered OL Systems.

Now we are ready to define formally structred OL systems.

Definition: A structured OL-schema (SOL-system) is a tuple $G = (N,T,P,\sigma)$ where

N is an alphabet of nonterminals (states of structural parts),

T is an alphabet of terminals,

P is a finite set of productions from $(N \cup (N\times N)) \times (N \cup \{\lambda\}) \cup (T \cup (N\times T)) \times (N,T)_*$,

$\sigma \in (N,T)_+$, the initial structure.

The productions of an SOL-schema will be written in the following form (all possible types are given). $A \to C$, $A\cdot B \to C$, $a \to \alpha$ and $A\cdot a \to \alpha$ where $A,B \in N$, $C \in N \cup \{\lambda\}$, $a \in T$, $\alpha \in (N,T)_*$ and "$\to$", "$\cdot$" are special reserved delimiters.

Definition: We write $\alpha \Longrightarrow \beta$ for $\alpha \in (N,T)_+$ $\beta \in (N,T)_*$ where $\alpha = x_1 x_2 \ldots x_n$ for $x_i \in N \cup T \cup \{[,]\}$, $1 \leq i \leq n$, if there exists $\beta' = w_1 w_2 \ldots w_n$ so that for $1 \leq i \leq n$:

(i) if $x_i \in \{|,|\}$ then $w_i = x_i$;

(ii) if $x_i \in T \cup N$ then either $x_i \to w_i \in P$ or

$A\cdot x_i \to w_i \in P$ where $A_i = \text{father}_\alpha(x_i)$;

(iii) β is obtained from β' by repeated replacing of each subexpression of the form $X\lambda Y, \lambda[Z]$ or $A[\lambda]$ by XY, Z or λ, respectively, until either $\beta = \lambda$ or there is no occurence of λ in β.

Let $\Longrightarrow^*$ be the transitive and reflexive closure of relation $\Longrightarrow$.

Definition: The set of structures generated by G is denoted $T(G)$ and defined to be the set $T(G) = \{\alpha : \sigma \Longrightarrow^* \alpha\}$.

The language generated by G, denoted by $L(G)$, is the set of the yields of all the structures generated by G. Formally $L(G) = \{Y(\alpha) : \alpha \in T(G)\}$.

Definition: A structured OL-schema G is called a structured OL-system (SOL-system) if it satisfies the condition of completness: For every $\alpha \in T(G)$ there exists β such that $\alpha \underset{G}{\Longrightarrow} \beta$.

Our definition of completness requires that for every structure which can be developed from the starting structure there is a "next step" in development. After some definitions and auxiliary results it will be shown that the choice of a more restrictive definition of completness (strong completeness) requiring existence of a

"next step" for every structure in $(N,T)_+$ does not change the families of sets of structures or languages generated by SOL-systems.

Definition: An SOL-system $G = (N,T,P,\sigma)$ is called full if

$P \subseteq \{S \to S\} \cup ((N\times\bar{N}) \times (\bar{N} \cup \{\lambda\})) \cup ((N\times T) \times (\bar{N},T)_*)$,

where $\bar{N} = N - \{S\}$, S being the root of σ, i.e. the father-context appears on the left side of every production with the exemption of the production which keeps unchanged the (reserved) label of the root.

Definition: SOL-systems G_1 and G_2 are called equivalent if $L(G_1) = L(G_2)$.

Lemma 1: For every SOL-system there exists an equivalent full SOL-system.

Proof. Given an SOL-system $G = (N,T,P,\sigma)$ we construct a full SOL-system $G' = (N',T,P',\sigma')$ where $N' = N \cup \{S\}$ for a new symbol S not in $N \cup T$, $\sigma' = S[\sigma]$, and $P' = \{A\cdot X\to w : A\cdot X\to w \in P\} \cup \{A\cdot X\to w : X\to w \in P \text{ and } A \in N'\} \cup \{S\to S\}$.

Clearly, $L(G') = L(G)$.

Definition: An SOL-system $G = (N,T,P,\sigma)$ is called strongly complete if for every $\alpha \in (N,T)_+$ there exists β such that $\alpha \underset{G}{\Longrightarrow} \beta$.

Theorem 1: For every SOL-system G there exists an equivalent strongly complete SOL system.

Proof. By Lemma 1 we may assume that $G = (N,T,P,\sigma)$ is full. We construct the SOL-system $G' = (N,T,P',\sigma)$ where $P' = P \cup \{A\cdot X\to X : A \in N, X \in N \cup T$ and there is no production of the form $A\cdot X\to W$ in $P\}$.

Clearly, G' is full and $L(G') = L(G)$.

Now we will study a special case of SOL-system, called simple SOL-system (SSOL-systems), in which essentially only terminal symbols are rewritten at every step of a derivation. We will show this subclass of SOL-systems generates exactly the same family of languages as several other types of systems known already to be equivalent, namely FMOL systems [9], EOL systems [35], and length preserving homomorphisms of OL languages [10,20]. This does not mean SSOL-systems are without interest, on the contrary they give an alternative mechanism for the description of languages from a very natural class with the advantage to exhibit explicitly the structure of generated objects. They also contribute another evidence to our opinion that the family of the

length-preserving homomorphisms of OL-languages is a very natural class of languages.

Definition: AN SOL-system $G = (N,T,P,\sigma)$ is called simple (SSOL) if $P = \{A \to A : A \in N\} \cup P'$ where $P' \subseteq (N \times T \cup T) \times (N,T)_*$.

In the following we omit the "identity productions" of form $A \to A$ whenever an SSOL-system is exhibited.

Notation. Let COL = {h(L) : L ∈ OL, h is a length-preserving homomorphism}.

Lemma 2: COL ⊆ SSOL.

Proof. Let $G = (\Sigma,P,\sigma)$ be an OL-system and h be a homomorphism from Σ^* to T^*. Clearly, we can assume withnout loss of generality that $\Sigma \cap T = \emptyset$.

Construct SSOL system $G' = (\Sigma \cup \{S\},T,P',\sigma')$ where

(i) S is a new symbol not in $\Sigma \cup T$.

(ii) Let $\sigma = a_1 \ldots a_n$, $a_i \in \Sigma$ for $i = 1,2,\ldots n$. Then

$\sigma' = S[a_1[h(a_1)]a_2[h(a_2)] \ldots a_n[h(a_n)]]$.

(iii) $P' = \{a \cdot h(a) \to b_1[h(b_1)]b_2[h(b_2)] \ldots b_k[h(b_k)] :$

$: a,b_1,b_2,\ldots,b_k \in \Sigma$ and $a \to b_1 b_2 \ldots b_k \in P\}$.

Clearly, $h(L(G')) = L(G)$.

Lemma 3: SSOL ⊆ COL.

Proof. Let $G = (N,T,P,\sigma)$ be an SSOL-system. By modification of Lemma 1 we can clearly assume that $P \subseteq (N \times T) \times (N,T)_*$ with "identity" productions omitted. Let $\Sigma = N \times T$ and let g be the mapping from $(N,T)_*$ into Σ^* which maps a structure $\alpha \in (N,T)_*$ to the string $(A_1,a_1) \ldots (A_n,a_n)$ such that $a_1 \ldots a_n = Y(\alpha)$ and $A_i = \text{father}_\alpha(a_i)$ for $1 \leq i \leq n$. In particular $g(\lambda) = \varepsilon$. We construct OL system $G' = (\Sigma,P',\sigma')$ where $\sigma' = g(\sigma)$ and $P' = \{(A,a) \to g(A|\beta|) : A \in N, a \in T$ and $A \cdot a \to \beta \in P\}$.

Note that if forest β does not include a tree consisting from a single node only, then $g(A|\beta|) = g(\beta)$. Let h be the homomorphism from Σ^* to T^* defined by $h((A,a)) = a$ for every $(A,a) \in \Sigma$.

It is easy to verify that $h(L(G')) = L(G)$.

Theorem 2: SSOL = COL = EOL = FMOL.

Proof. By Lemma 2 and 3 we have the first equation; the definitions of FMOL languages and other results are in [9,20]

4. Closure Properties of SOL.

To show the relation of SOL to other known families of languages we will need some closure results which will be shown first. The most interesting is the closure of SOL under ε-free homomorphisms which in particular means that codings (length-preserving homomorphisms) do not increase the descriptive power of SOL-systems.

Theorem 3: Family SOL is closed under ε-free homomorphisms.

Proof. Given an SOL language L over T and a homomorphism h from T^* to Σ^* we may assume by Lemma 1 that L is generated by full SOL system $G = (N,T,P,\sigma)$ and we construct SOL-system $G' = (N',\Sigma,P'\sigma')$ as follows.

Let $N' = N \cup T \cup (N\times T) \cup \{\bar{A} : A \in N\} \cup \{Q\}$ where Q is a new symbol not in $N \cup T$. Let f be the homomorphism from $(N,T)_*$ into $(N',\Sigma)_*$ defined as follows. The forest expression $f(\alpha)$ is obtained from expression α by replacing every terminal symbol from T, say a, by subexpression $a[a_1Q[a_2]Q[a_3] \ldots Q[a_n]]$ where $h(a) = a_1a_2\ldots a_n$.

Let $\sigma' = f(\sigma)$ and productions P' be constructed as follows:

(i) $A \to \bar{A}$ is in P' for every A in N.

(ii) $A \cdot a \to (A,a)$ is in P' for all A in N and a in T.

(iii) $a \cdot t \to t$ is in P' for all a in T and t in Σ.

(iv) $\bar{S} \to S$ is in P'.

(v) If $A \cdot B \to X$ is in P then $\bar{A} \cdot \bar{B} \to X$ is in P' for all A,B in N and X in $N \cup \{\lambda\}$.

(vi) If $A \cdot a \to \alpha$ is in P and $h(a) = a_1a_2\ldots a_n$ then $(A,a) \cdot a_1 \to f(\alpha)$ is in P' for all A in N and a in T.

(vii) $(A,a) \to \lambda$ is in P' for all A in N and a in T.

(viii) $Q \to \lambda$ is in P'.

(ix) $Q \cdot t \to \lambda$ for every t in Σ.

Clearly, in any derivation in G' the productions (i)-(iii) and the productions (iv)-(ix) can be used only in alternative steps, namely, the former in odd steps and later in even steps of any derivation. Realizing this it is straigthforward to verify that $L(G') = h(L(G))$.

Theorem 4: The family SOL is closed under union, concatenation and star.

Proof. Let $G_1 = (N_1,T_1,P_1,\sigma_1)$ and $G_2 = (N_2,T_2,P_2,\sigma_2)$ be SOL-systems. Assume that $N_1 \cap N_2 = \emptyset$.

To show the closure under union we construct an SOL-system $(N_3,T_1 \cup T_2,P_3,\sigma_3)$ as follows. Let $N_3 = N_1 \cup N_2 \cup \{S,Q\}$ with S and Q new symbols not in $N_1 \cup N_2 \cup T_1 \cup T_2$. Assume $Y(\sigma_1) = a_1 \ldots a_n$. Let $\sigma_3 = S[a_1 Q[a_2 \ldots a_n]]$ and $P_3 = P_1 \cup P_2 \cup \{S \cdot a_1 \to \sigma_1$, $S \cdot a_1 \to \to \sigma_2, Q \cdot a_i \to \lambda$ for $i = 2,\ldots,n\}$.

Clearly, $L(G_3) = L(G_1) \cup L(G_2)$.

To show the closure under concatenation we construct SOL-system $(N_4,T_1 \cup T_2,P_4,\sigma_4)$ as follows. $N_4 = N_1 \cup N_2 \cup \{S,A,B,C,D,E,F,H,Q\}$ where $S,A,\ldots,Q$ are new symbols. Assume $Y(\sigma_1) = a_1 \ldots a_n$ and $Y(\sigma_2) = b_1 \ldots b_m$. Let $\sigma_4 = S[A[C[a_1]E[a_2 \ldots a_n]]A[D[b_1]E[b_2 \ldots b_m]]]$ and $P_4 = P_1 \cup P_2 \cup P_4'$ where P_4' consists of the following productions:

(I)		(II)	
	$A \to A$		$A \to B$
	$A \cdot C \to C$		$B \cdot C \to F$
	$A \cdot D \to D$		$B \cdot D \to H$
	$A \cdot E \to E$		$B \cdot E \to Q$
	$C \cdot a_1 \to a_1$		$F \cdot a_1 \to \sigma_1$
	$D \cdot b_1 \to b_1$		$Q \cdot a_i \to \lambda$ for $i=2,\ldots,n$
	$E \cdot a_i \to a_i$ for $i=2,\ldots,n$		$H \cdot b_1 \to \sigma_2$
	$E \cdot b_i \to b_i$ for $i=2,\ldots,n$		$Q \cdot b_i \to \lambda$ for $i=2,\ldots,m$.

Productions of group (I) allow to delay the start of the generation of strings in $L(G_1)$ or $L(G_2)$ to assure that even by parallel generation all strings in $L(G_1) \cdot L(G_2)$ are obtained. Once the production $A \to B$ is used the productions of group (II) assure the start of generation from σ_1 by productions P_1 or from σ_2 by productions P_2. It is straightforward to verify that $L(G_4) = L(G_1) \cdot L(G_2)$.

Finaly, to show the closure of SOL under star we construct SOL-system (N_5,T_1,P_5,σ_5) as follows. Let $N_5 = N_1 \cup \{S,A,B,C,D,E\}$ where S,A,B,C,D,E are new symbols not in $N_1 \cup T_1$. Assume $\sigma_1 = a_1 a_2 \ldots a_n$. Let $\sigma_5 = S[a_1 Q[a_2 a_3 \ldots a_n]]$ and $\alpha = A[C[a_1]D[a_2 a_3 \ldots a_n]]$. Let $P_5 = P_1 \cup P'$ where P' consists of the following productions:

$S \to S$	$A \cdot C \to C$
$S \cdot a_1 \to \alpha\sigma_5$	$A \cdot D \to D$
$S \cdot a_1 \to \lambda$	$B \cdot C \to E$
$Q \to \lambda$	$B \cdot D \to Q$
$Q \cdot a_i \to \lambda$ for $2 \leq i \leq n$	$C \cdot a_1 \to a_1$
$A \to A$	$D \cdot a_i \to a_i$ for $2 \leq i \leq n$
$A \to B$	$E \cdot a_1 \to \sigma_1$

It is straigthforward to verify that G_5 generates only strings in $L(G_1)^*$. To see that all such strings are generated we observe that $S \underset{G_5}{\Longrightarrow}^* S[\alpha S[\alpha S[\alpha \ldots S[\alpha]]\ldots]]$ and that $\alpha \Longrightarrow^k \alpha \Longrightarrow B[C[a_1]D[a_2a_3\ldots a_n]] \Longrightarrow B[E[a_1]Q[a_2a_3\ldots a_n]] \Longrightarrow B[E[\sigma_1]] \Longrightarrow B[E[\beta]]$ for all $k \geq 0$ and β in $T(G_1)$. Therefore for any $m \geq 1$ and $\beta_1,\ldots,\beta_m$ in $T(G_1)$ we have $\alpha \Longrightarrow^{k_i} \alpha \Longrightarrow^* B[E[\beta_i]]$ for $1 \leq i \leq m$ and by suitable choice of k_i, $1 \leq i \leq m$, we can "synchronized" derivation $S \Longrightarrow^* S[B[E[\beta_1]]\ S[B[E[\beta_2]]\ \ldots$ $\ldots\ S[B[E[\beta_m]]]]\ldots]]$.

Since $\sigma_5 \Longrightarrow \lambda$ also $\varepsilon \in L(G_5)$.

5. Relation of SOL to other Families of Languages.

First we show that SOL-systems can simulate TOL-systems and then using the closure of SOL under ε-free homomorphisms this result will be generalized to ETOL-systems. It is easy to see that these results can be further generalized to (E)TOL-systems with some restrictions on the sequences of productions which may be used in a derivation.

Lemma 4: TOL $\subseteq$ SOL.

Proof. Given TOL system $G = (T,\{P_1,\ldots,P_n\},\sigma)$ we construct SOL-system $G' = (T,N,P,\sigma')$ where $N = \{0,1,\ldots,n\}$, $\sigma' = 0[\sigma]$ and $P = \{i \cdot a \to w : a \to w \in P_i\} \cup$ $\cup \{i \to j : i \in N,\ j \in N - \{0\}\} \cup \{0 \cdot a \to a : a \in T\}$.

Clearly, $L(G') = L(G)$ and therefore TOL $\subseteq$ SOL.

The following lemma shows that in certain restricted manner every recursively enumerable set can be represented by an SOL-system.

Lemma 5: Let L be a recursively enumerable set over Σ and let \$,# be not in Σ. There exists an SOL language $L' \subseteq (\Sigma \cup \{\$\})^*\{\#\}$ such that $L\{\#\} = L' \cap \Sigma^*\{\#\}$.

Proof. It follows from results in [G,H,P]†, see for example Lemma 1 in [P]† that L can be generated by a phrase-structure grammar $G = (N,T,P,S)$ with productions only of the form $A \to B$, $A \to BC$, $AB \to AC$, $A \to a$ or $A \to \varepsilon$ for $A,B,C \in N$ and $a \in T$, i.e. there are only context-free or "left-context-sensitive" productions in P. We can then construct SOL system $G' = (N',T',P',\sigma)$ where $T' = T \cup \{\$,\#\}$, $N' = N \cup N^2 \cup \{\bar{A} : A \in N\} \cup \{Q\}$ for Q not in $N \cup T \cup \{\$,\#\}$, $\sigma = S[\$\ \#]$ and P' is defined as follows.

(i) For all A,B,C in N and a in $T \cup \{\$\}$ the following productions are in P'.

$A \cdot B \to B$ $\qquad$ $(A,B) \cdot a \to \$$

$(A,B) \cdot C \to (B,C)$ $\qquad$ $Q \to \lambda$

$(A,B) \cdot (C,D) \to (B,C)$ $\qquad$ $Q \cdot a \to \lambda$

$A \cdot (B \cdot C) \to B$ $\qquad$ $A \cdot \# \to \#$

$A \cdot \bar{B} \to B$ $\qquad$ $\bar{A} \cdot \# \to \$\#$

$(A,B) \cdot \bar{C} \to (B,C)$ $\qquad$ $(A,B) \cdot \# \to \bar{B}[\#]$

$A \cdot a \to \$$

(ii) If $A \to a$, $B \to b$ are in P and d is in $T \cup \{\$\}$ then the following productions are in P'.

$A \cdot d \to a$

$(A,B) \cdot d \to a\ Q|b|$

$\bar{A} \cdot \# \to a\#$.

(iii) If $A \to \varepsilon$ is in P then $D \cdot A \to Q$ is in P' for every D in N.

If $A \to B$ is in P then $D \cdot A \to B$ is in P' for every D in N.

If $A \to BC$ is in P then $D \cdot A \to (B,C)$ is in P' for every D in N.

If $AB \to AC$ is in P then $A \cdot B \to C$ is in P'.

Let h be the homomorphism from $(N \cup N^2 \cup \{\bar{A} : A \in N\})^*$ into N^* defined by $h(A) = h(\bar{A}) = A$ for every A in N and $h(A,B) = AB$ for all A,B in N. It can be verified (by induction on the length of a derivation) that if $\sigma \overset{*}{\underset{G'}{\Longrightarrow}} \alpha$ then α must be of the

† |G|: A.V. Gladkij, Formal grammars and languages (in Russian) Mir, Moscow, 1973

|H|: L.H. Haines, A representation for context sensitive languages, Transaction of the Amer. Math. Society, to appear.

|P|: M. Penttonen, LCS = CS, to appear in Information and Control.

form $X_1[t_1X_2[t_2X_3[\ldots X_{k-1}[t_{k-1}X_k[t_k\#]]\ldots]]]$ where t_i is in $T\cup\{\$\}$ for $i = 1,\ldots,k$, X_i is in $N\cup N^2$ for $i = 1,\ldots,k-1$ and X_k is in $N\cup N^2\cup\{\bar{A} : A\in N\}$ and $h(X_1X_2\ldots X_k)$ is in L(G). Moreover, if $t_1t_2\ldots t_k$ is in T^* then also $t_1t_2\ldots t_k$ is in L(G). Thus $L(G')\subseteq L(G)\cap T^*\{\#\}$.

To show the reverse inclusion we observe that every string x in L(G) can be generated by a derivation $S \Longrightarrow^* w \Longrightarrow^* x$ such that $w \in N^*$ and we do not use the productions of the form $A\to a$ in the derivation $S \Longrightarrow^* w$ and on the other hand we use only such productions in $w \Longrightarrow^* x$. Further we can see that

(i) If $A_1A_2\ldots A_n \underset{G}{\Longrightarrow} B_1B_2\ldots B_m$ for A_i in N for $1 \le i \le n$ and B_j in N for $1 \le j \le m$ then there exist $t_1,\ldots,t_n$, $s_1,\ldots,s_m$ in $T\cup\{\$\}$ such that
$A_1[t_1A_2[t_2\ A_3[\ldots A_{n-1}[t_{n-1}\ A_n[t_n\#]]\ldots]]] \underset{G'}{\Longrightarrow}^* B_1[s_1B_2[s_2\ B_3[\ldots$
$\ldots\ B_{m-1}[s_{m-1}B_m[s_m\#]]\ldots]]]$.

(ii) If $A_i\to a_i \in P$ for $1 \le i \le k$ then
$A_1[t_1A_2[t_2\ldots A_{k-1}[t_{k-1}A_k[t_k\#]]\ldots]] \underset{G'}{\Longrightarrow} A_1[a_1A_2[a_2\ldots A_{k-1}[a_{k-1}A_k[a_k\#]]\ldots]]$.

Therefore every derivation in G can be simulated by a derivation in G' and $L(G)\cap T^*\{\#\} \subseteq L(G')$.

Theorem 5. Family SOL is incomparable with the family of context sensitive languages (CSL).

Proof. Every SOL languages is clearly exponencially dense in the terminilogy of [CO]†, i.e. for every SOL language L there exist constants p,q such that for every string u in L of length n, $n \ge p$ there is string v in L of length m so that $\frac{n}{q} \le m < n$. There are context-sensitive languages not satisfying this property, e.g. the language $\{a^{2^{2^n}} : n \ge 0\}$ therefore CSL $\not\subseteq$ SOL.

CSL is closed under intersection with a regular set therefore for every context-sensitive language L' the language $L'\cap \Sigma^*\{\#\}$ is again context sensitive. Thus the assumption SOL $\subseteq$ CSL is in contradiction with Lemma 5.

Corollary: Family SOL is not closed under intersection with a regular set.

Proof. By Lemma 5 and Theorem 5.

Theorem 6. ETOL $\subsetneq$ SOL $\cup$ {{ε}}.

---

† |CO| K. Culik II and J. Opatrny, Context in parallel rewriting, in this volume.

Proof. In [27] it is shown that ETOL $-\{\{\varepsilon\}\}$ is equal to the closure of TOL under length-preserving homomorphisms (codings). Therefore it follows by Lemma 4 and Theorem 3 that ETOL $\subseteq$ SOL $\cup \{\{\varepsilon\}\}$. In [7] it is shown that ETOL is included in the family of indexed languages $[A]^{\dagger}$. Thus our inclusion is proper by Theorem 5.

From Lemma 4 and results in $[CO]^{*}$ it follows that SOL is included neither in the family of languages generated by L-systems with interaction [88] nor in the family of predictive context languages $[CO]^{*}$. We conjecture that both these families are incomparable with SOL.

---

† |A|: A.V. Aho, Indexed grammars - an extention of context-free grammars, JACM 15, (1968), 647 - 671.

* |CO| K. Čulik II and J. Opatrný, Context in parallel rewriting, in this volume.