NONTERMINALS VERSUS HOMOMORPHISMS IN DEFINING LANGUAGES FOR SOME CLASSES OF REWRITING SYSTEMS+

A. Ehrenfeucht
Department of Computer Science
University of Colorado
Boulder, Colorado 80302 U.S.A.

G. Rozenberg
Institute of Mathematics
University of Utrecht
Utrecht-Uithof, The Netherlands

Report # CU-CS-027-73 August, 1973

All correspondence to:

G. Rozenberg
Institute of Mathematics
Utrecht University
Utrecht - Uithof
The Netherlands

+ This work supported by National Science Foundation Grant # GJ-660

TABLE OF CONTENTS.

ABSTRACT.

Given a rewriting system G (its alphabet, the set of productions and the axiom) one can define the language of G by

(i) taking out of all strings generated by G only those which are over a distinguished subalphabet of G, or

(ii) translating the set of all strings generated by G by a fixed homomorphism.

The "trade-offs" between these two mechanisms for defining languages are discussed for both, "parallel" rewriting systems from developmental systems hierarchy and "sequential" rewriting systems from the Chomsky hierarchy.

0. INTRODUCTION.

Given a rewriting system G consisting of an alphabet V, the set of productions P and the axiom ω, the most natural set of strings associated with G is the set of all strings one can "derive in G" starting with ω and using productions from P. Now one can define "the language of G" in at least two different ways:

(i) One defines a subalphabet V_T of V and then takes as the language of G only this subset of the set of all strings derived in G which consists of all strings over the alphabet V_T.

(ii) One defines a homomorphism h and then takes as the language of G the set of all images under h of all words derived in G.

The first of these definitional mechanisms is called "defining languages by the use of nonterminals" and the second one is called "defining languages by the use of homomorphic tables".

Thus, if $G = \langle V, P, \omega \rangle$ then its language in the first case is defined by the system $G' = \langle V, P, \omega, V_T \rangle$ and in the second case by the system $G' = \langle V, P, \omega, h \rangle$.

Both approaches are used in formal language theory.

The use of nonterminals is a very well-established mechanism in formal language theory and dates back at least to the fundamental works of Chomsky (see [Chomsky, 1956]). It also has a deep linguistical motivation (for a discussion of which the reader is referred to [Chomsky, 1957]).

Homomorphic images of languages were the subject of quite a number of papers in formal language theory (see, e.g., [Ginsburg and Greibach]). But it is really the theory of developmental languages (see, e.g., [Lindenmayer], [Lindenmayer and Rozenberg], and [Herman and Rozenberg]) in which the set of all strings generated by a rewriting system G is of primary interest and then the homomorphic mappings (especially those in which a letter is mapped to a letter, the so-called codings) of such languages are the "second natural objects". Indeed, it is the fact that the class of languages generated by the use of codings turned out to be a subclass of the class of languages generated by the use of nonterminals, for some classes of developmental systems, which made the use of nonterminals interesting at all within the theory of developmental systems and languages (for a discussion of this subject see, e.g., [Herman, Lindenmayer and Rozenberg]).

Hence, the trade-off between these two mechanisms for defining languages becomes an interesting and well-motivated problem to investigate.

For example, in [Ehrenfeucht and Rozenberg] it was proven that if one considers a particular class of rewriting systems (the so-called)L systems) then the use of codings or nonterminals for defining languages are equivalent. This motivated quite strongly the investigation of the use of nonterminals within this class of systems (see, e.g., [Herman], [Herman, Lindenmayer, and Rozenberg], and [Rozenberg and Doucet]).

This paper continues the work from [Ehrenfeucht and Rozenberg]. We investigate the trade-off between the use of nonterminals and the use of codings. It is shown that whereas these mechanisms are equivalent in the class of developmental systems without interactions (the so-called EOL and ETOL systems) they are not equivalent when one deals with the class of systems in which rewritings may be "context-dependent" (the so-called EIL systems). In the latter case we also show how to modify codings to get the equivalence of these two mechanisms for language definition.

The last part of this paper deals with the trade-off between the use of nonterminals and the use of homomorphic mappings in the classes of context-free and context-sensitive grammars from the Chomsky hierarchy of classes of rewriting systems.

1. PRELIMINARIES.

We assume the reader to be familiar with basics of formal language theory, e.g., in the scope of [Hopcroft and Ullman], whose notation and terminology we shall mostly follow. In addition, we shall use the following notation:

(i) N denotes the set of nonnegative integers and $N^+ = N - \{0\}$.

(ii) If x is a word, then $|x|$ denotes its length and Min(x) denotes the set of letters which occur in x.

(iii) If A is a set then 2^A denotes the set of subsets of A and in the case when A is finite, #A denotes its cardinality. If B is also a set then $A \subseteq B$ denotes the inclusion of A in B and $A \subset B$ denotes the strict inclusion of A in B.

(iv) If A is an ultimately periodic set of nonnegative integers then thres(A) denotes the smallest integer j for which there exists a positive integer q such that, for every $i \geq j$, if i is in A then $(i + q)$ is in A. The smallest positive integer p such that, for every $i \geq$ thres(A), whenever i is in A then also $(i + p)$ is in A, is denoted by per(A).

(v) If $\tau = (w_1, w_2, w_3, \ldots)$ is a sequence of words then a sequence $w_{i_1}, w_{i_2}, \ldots$ such that, for each $j \geq 1$, $i_j < i_{j+1}$ is called a subsequence of τ.

(vi) If Σ is an alphabet and ℓ is in N then $\Sigma^\ell = \{x \in \Sigma^* : |x| = \ell\}$.

(vii) $\emptyset$ denotes the empty set and Λ denotes the empty word.

(viii) If A is a finite automaton then L(A) denotes its language.

(ix) If G is a context-sensitive grammar then L(G) denotes its language and Sent(G) denotes the set of sentential forms of G (the set of all strings that can be derived in G starting with its axiom).

(x) A homomorphism which maps a letter to a letter is called a coding and a homomorphism which maps a letter either to a letter or to the empty word is called a weak coding.

In some portions of this paper we shall assume familiarity with some additional material, but this will be always explicitly stated.

2. ETOL SYSTEMS AND LANGUAGES.

In this section we introduce ETOL systems and languages which is the first class of systems and languages to be investigated in this paper. We refer the reader to [Rozenberg, a] and [Rozenberg, b] where different properties of this class of systems and languages are proved.

Definition 1. An ETOL system is a construct $G = \langle V_N\ V_T, \mathcal{P}, \omega\rangle$ such that
V_N is a finite alphabet (of nonterminal letters or symbols),
V_T is a finite nonempty alphabet (of terminal letters or symbols), such that $V_N \cap V_T = \emptyset$,
ω is an element of $(V_N \cup V_T)^+$ (called the axiom of G),
$\mathcal{P}$ is a finite nonempty family, each element of which is a finite nonempty set of the form $\{a \to \alpha : a$ is in $V_N \cup V_T$ and α is in $(V_N \cup V_T)^*\}$ (where we assume that the symbol $\to$ is not in $V_N \cup V_T$).
Each element P of $\mathcal{P}$ (called a table of G) satisfies the condition:

for every a in $V_N \cup V_T$ there exists at least one α in $(V_N \cup V_T)^*$ such that $a \to \alpha$ is in P.

(If P is in $\mathcal{P}$, and $a \to \alpha$ is in P then $a \to \alpha$ is called a production in P, or a production of G. We often write $a \underset{P}{\to} \alpha$ for "$a \to \alpha$ is in P". $V_N \cup V_T$ is called the alphabet of G).

Definition 2. Let $G = \langle V_N, V_T, \mathcal{P}, \omega\rangle$ be an ETOL system.

(i) A production $a \to \alpha$ of G such that $\alpha = \Lambda$ is called an erasing production. G is called propagating if no production of G is an erasing production.

(ii) G is called a TOL system, if $V_N = \emptyset$. (In this case we often write G as $\langle V_T, \mathcal{P}, \omega\rangle$.)

(iii) G is called an <u>EOL system</u>, if $\#\mathcal{P} = 1$. (In this case, if $\mathcal{P} = \{P\}$, then we often write G as $\langle V_N, V_T, P, \omega\rangle$.)

(iv) G is called a <u>OL system</u> if $\#\mathcal{P} = 1$ and $V_N = \emptyset$. (In this case, if $\mathcal{P} = \{P\}$, then we often write G as $\langle V_T, P, \omega\rangle$.)

We shall use the letter P to denote the propagating restriction. For example, "an EPTOL system" means a propagating EPTOL system.

<u>Definition 3</u>. Let $G = \langle V_N, V_T, \mathcal{P}, \omega\rangle$ be an ETOL system.

(i) Let $x \in (V_N \cup V_T)^+$, say $x = b_1 b_2 \dots b_t$ for some $b_1, b_2, \dots, b_t$ in $V_N \cup V_T$, and let $y \in (V_N \cup V_T)^*$. We say that x <u>directly derives</u> y (<u>in</u> G), denoted as $x \underset{G}{\Longrightarrow} y$, if there exist a P in $\mathcal{P}$ and a sequence $\pi_1, \dots, \pi_t$ of productions from P such that, for every i in $\{1, \dots, t\}$, $\pi_i = b_i \to \alpha_i$ and $y = \alpha_1 \dots \alpha_t$. (In this case we also write $x \underset{P}{\Longrightarrow} y$, and we say that x <u>directly derives</u> y (<u>in</u> G) <u>using</u> P).

(ii) Let $x \in (V_N \cup V_T)^+$ and $y \in (V_N \cup V_T)^*$. We say that x <u>derives</u> y <u>in</u> G, denoted as $x \overset{*}{\underset{G}{\Longrightarrow}} y$, if,

either $x = y$,

or, for some $n > 0$, there exists a sequence $x_0, x_1, \dots, x_n$ of words in $(V_N \cup V_T)^*$, such that $x_0 = x$, $x_n = y$ and $x_{i-1} \underset{G}{\Longrightarrow} x_i$ for $1 \leq i \leq n$. (If the latter holds then we also write $x \overset{+}{\underset{G}{\Longrightarrow}} y$. Also, if $\tau = T_1 \dots T_n$ is a sequence of tables from $\mathcal{P}$ and $x_0 \underset{T_1}{\Longrightarrow} x_1 \underset{T_2}{\Longrightarrow} x_2 \dots \underset{T_n}{\Longrightarrow} x_n$, then we write $x \overset{\tau}{\Longrightarrow} y$).

(iii) For x in $(V_N \cup V_T)^+$ and y in $(V_N \cup V_T)^*$, a <u>derivation of</u> y <u>from</u> x (<u>in</u> G) is a sequence $D = (x_0, x_1, \dots, x_n)$ of words in $(V_N \cup V_T)^*$ such that for $1 \leq i \leq n$, $x_{i-1} \underset{G}{\Longrightarrow} x_i$. If $x_0 = \omega$, then D is called a <u>derivation of</u> y (<u>in</u> G), and then we say that x is <u>derived in</u> G <u>in</u> n <u>steps</u>, and write $x_0 \overset{n}{\underset{G}{\Longrightarrow}} x_n$ (by definition $x \overset{0}{\underset{G}{\Longrightarrow}} x$, for every x in $(V_N \cup V_T)^*$).

(iv) Let $D = (x_0, x_1, \ldots, x_n)$ be a derivation of x_n from x_0 in G and let $T_1 \ldots T_n$ be a sequence of tables from P such that $x_0 \underset{T_1}{\Longrightarrow} x_1 \underset{T_2}{\Longrightarrow} \ldots \underset{T_n}{\Longrightarrow} x_n$. A control sequence of D is a sequence $\overline{T}_1, \ldots, \overline{T}_n$ of sets of productions such that, for each $1 \leq i \leq n$, $\overline{T}_i \subseteq T_i$ and $\overline{T}_i$ consists of all and only these productions which are "used" in deriving x_i from x_{i-1}. (Note that the sets of productions in a control sequence of D may contain no productions for some letters. For this reason we often "complete them" by adding productions of the form $a \to a$ for all letters a for which the table does not contain a production. The resulting sequence is also called a control sequence of D but this should not lead to any confusion).

Definition 4. Let $G = \langle V_N, V_T, P, \omega \rangle$ be an ETOL system. The language of G, denoted as L(G), is defined by $L(G) = \{x \in V_T^* : \omega \overset{*}{\underset{G}{\Longrightarrow}} x\}$.

Definition 5. A nonempty language K different from $\{\Lambda\}$ is called an ETOL (TOL, EOL, EPTOL, etc.) language if, and only if, there exists an ETOL (TOL, EOL, EPTOL, etc.) system G such that $L(G) = K$. A language K is called a CTOL language (an HTOL language) if, and only if, there exists a TOL language $\overline{K}$ and a coding h (a homomorphism h) such that $h(\overline{K}) = K$.

In the sequel F_{ETOL}, F_{CTOL} and F_{HTOL} will denote the classes of ETOL, CTOL and HTOL languages, respectively. In fact in this paper whenever F_X denotes the class of languages, then F_{CX}, F_{WX}, F_{HX} and F_{PHX} denote the class of coding of languages from F_X, the class of weak codings of languages from F_X, the class of homomorphic images of languages from F_X and the class of images under Λ-free homomorphisms of languages from F_X.

Remark 1. In the sequel, given an ETOL system $G = \langle V_N, V_T, P, \omega \rangle$, we shall sometimes consider P as a set of tables as defined in Definition 1, and sometimes we shall consider P to be the set of symbols ("names" of tables), but this should not lead to any confusion. (For example, we may talk about an alphabet P, words over P, etc.).

Remark 2. It is well-known (see e.g., [Rozenberg, a]) that for every ETOL system G there exists an ETOL system $H = \langle V_N, V_T, P, \omega \rangle$ such that ω is in V_N. Thus in the sequel we shall often assume that an ETOL system we deal with is such that its axiom is a nonterminal symbol, and in this case the axiom shall be denoted by the symbol S.

We end this section with two examples of ETOL systems and languages.

Example 1.

$G = \langle \emptyset, \{a, b\}, \{\{A \to A^2, B \to B\}, \{A \to A, A \to AB, A \to BA, B \to B\}\}, A \rangle$ is a PTOL system such that $L(G) = \{x \in \{a, b\}^+ : \#_a(x) = 2^n \text{ for some } n \geq 0\}$.

Example 2.

$G = \langle \{A\}, \{a\}, \{\{A \to A^2, a \to a\}, \{A \to a^3, a \to a\}\}, A \rangle$ is a EPTOL system such that $L(G) = \{a^{3 \cdot 2^n} : n \geq 0\}$.

3. SPECTRA OF SETS OF SYMBOLS IN ETOL SYSTEMS.

In this section we introduce the basic notion of the so-called spectrum of a set of symbols in an ETOL system. (In this section and in most of the next section we talk about EPTOL systems only and, for the sake of clarity, though most of the notions apply to arbitrary ETOL systems we define them for EPTOL systems only).

We start by defining an analogous notion for a finite automaton.

Definition 6. Let $A = \langle Q, \Sigma, \delta, q_0, F\rangle$ be a finite automaton and let q be in Q. The spectrum of q in A, denoted as Spec(A, q), is defined by $\mathrm{Spec}(A, q) = \{n \in N^+ : \delta(q, x) \in F$ for some x in Σ^+ such that $|x| = n\}$.

Spectra of states in finite automata satisfy the following basic property.

Lemma 1. If A is a finite automaton and q is its state then Spec(A, q) is an ultimately periodic set.

Proof.

Let $A = \langle Q, \Sigma, \delta, q_0, F\rangle$ and let q be in Q. Let $A(q) = \langle Q, \Sigma, \delta, q, F\rangle$. It is clear that, for n in N^+, n is in Spec(A, q) if, and only if, there exists a word x in L(A(q)) such that $|x| = n$. But it is well-known that the set of lengths of words in a regular language is an ultimately periodic set (see, e.g., [Ginsburg, Theorems 2.1.2 and 2.1.3]). Thus Lemma 1 holds.

The following notions will be useful in the sequel.

Definition 7.

(i) Let A be a finite automaton and q a state of A. We say that q is weak if Spec(A, q) is a finite set, otherwise q is called strong.

(ii) A <u>uniform period for</u> A, denoted as m_A, is the smallest positive integer j such that

1) for each q in Q, $j > \text{thres}(\text{Spec}(A, q))$, and

2) for each strong q in Q, j is divisible by per(Spec(A, q)).

<u>Definition 8</u>. Let $G = \langle V_N, V_T, P, S\rangle$ be an EPTOL system and let B be a nonempty subset of $V_N \cup V_T$. The <u>spectrum of</u> B <u>in</u> G, denoted as Spec(G, B), is defined by

$$\text{Spec}(G, B) = \{n \in N^+ : \text{there exists a word } x \text{ in } B^+ \text{ and a sequence } \tau = P_{i_1} \ldots P_{i_n} \text{ of tables in } P \text{ such that } x \overset{\tau}{\underset{G}{\Longrightarrow}} w \text{ for some word } w \text{ in } V_T^+\}.$$

Thus a positive integer n is in Spec(G, B) if, and only if, there exists a sequence τ of length n of tables from P such that each letter in B can, "using τ" derive a word consisting of terminal symbols only.

<u>Definition 9</u>. Let $G = \langle V_N, V_T, P, S\rangle$ be an EPTOL system. The <u>closure of</u> P, denoted as Clos(P), is defined as the family of sets of productions such that R is in Clos(P) if, and only if,

(i) for some P in P, $R \subseteq P$, and

(ii) for every a in $V_N \cup V_T$, R contains at least one production with a as its left-hand side.

The spectra of states in finite automata and the spectra of sets of symbols in ETOL systems are connected through the following construct.

<u>Definition 10</u>. If $G = \langle V_N, V_T, P, S\rangle$ is an EPTOL system and B is a nonempty subset of $V_N \cup V_T$, then the B-<u>spectral representation of</u> G is a finite automaton, denoted as $A_{G,B}$, defined by $A_{G,B} = \langle Q, \Sigma, \delta, q_0, F\rangle$, where

$Q = 2^{V_N \cup V_T} - \emptyset$,

$\Sigma = \mathrm{Clos}(\mathcal{P})$,

$q_0 = B$,

$F = \{q \in Q : q \subseteq V_T\}$, and

for every q, $\bar{q}$ in Q and P in $\mathrm{Clos}(\mathcal{P})$, $\delta(q, P) = \bar{q}$ if, and only if,

$\bar{q} = \bigcup\{\mathrm{Min}(\alpha) : a \in q \text{ and } a \to \alpha \text{ is in } P\}$.

$\mathcal{A}_{G,\{S\}}$ is called a <u>spectral representation of</u> G and is denoted by $\mathcal{A}_G$.

The easy proof of the following result is left to the reader.

<u>Lemma 2</u>. Let $G = \langle V_N, V_T, \mathcal{P}, S\rangle$ be an EPTOL system and B be a nonempty subset of $V_N \cup V_T$. Then Spec(G, B) = Spec($\mathcal{A}$, B).

The following result expresses the basic property of spectra of sets of symbols in ETOL systems.

<u>Lemma 3</u>. If $G = \langle V_N\ V_T, \mathcal{P}, S\rangle$ is an EPTOL system and B is a nonempty subset of $V_N \cup V_T$, then Spec(G, B) is an ultimately periodic set.

<u>Proof</u>.

This result follows directly from Lemma 1 and Lemma 2.

<u>Definition 11</u>. If G is an EPTOL system, then the <u>uniform period of</u> G, denoted as m_G, is defined by $m_G = m_{\mathcal{A}_G}$.

Spectral representations of ETOL systems do not contain by themselves enough information for our purposes. However, together with the following construct they turn out to be the basic constructs for proving equality of $\mathcal{F}_{ETOL}$ and $\mathcal{F}_{CTOL}$.

Definition 12. Let G be an EPT0L system and let $A_G = \langle Q, \Sigma, \delta, q_0, F\rangle$, where we assume some, fixed but an arbitrary, ordering on Q, say $Q = \{u_0,\ldots,u_p\}$. (For convenience, we always assume that $u_0 = q_0$.) An indexed spectral representation of G, denoted as I_G, is a finite automaton $I_G = \langle \overline{Q}, \overline{\Sigma}, \overline{\delta}, \overline{q_0}, \overline{F}\rangle$ defined as follows:

$\overline{Q} = \{u_i \times \{i\} : 0 \le i \le p\}$.

$\overline{\Sigma} = \{P^{i,j} : P \in \Sigma,\ i, j \in \{0,\ldots,p\}$ and $\delta(u_i, P) = u_j\}$, where if $P \in \Sigma$, $i, j \in \{0,\ldots,p\}$ and $\delta(u_i, P) = u_j$, then $P^{i,j} = \{[a, i] \to [a, i] : a \in V_N \cup V_T$ and $a \notin u_i\} \cup \{[a, i] \to [b_1, j]\ldots[b_t, j] : a \in u_i$ and $a \to b_1\ldots b_t$ is in $P\}$.

$\overline{q_0} = \{u_0\} \times \{0\}$.

$\overline{F} = \{u_j \times \{j\} : u_j \in F\}$.

For $u_i \times \{i\}$, $u_k \times \{k\}$ in $\overline{Q}$ and $P^{i,j}$ in $\overline{\Sigma}$, $\overline{\delta}(u_i \times \{i\}, P^{i,j}) = u_k \times \{k\}$ if, and only if, $k = j$ and $\delta(u_i, P) = u_j$.

Note that an indexed spectral representation of G depends on the ordering of states of A_G, but as we always assume some fixed ordering of states of A_G, we can talk about the indexed spectral representation of G without any confusion.

We leave for the reader the easy proof of the following result.

Lemma 4. Let G be an EPT0L system, $A_G = \langle Q, \Sigma, \delta, q_0, F\rangle$ with $Q = \{u_0,\ldots,u_p\}$ and $I_G = \langle \overline{Q}, \overline{\Sigma}, \overline{\delta}, \overline{q_0}, \overline{F}\rangle$. Then for each i in $\{0,\ldots,p\}$, $\mathrm{Spec}(I_G, u_i \times \{i\}) = \mathrm{Spec}(A_G, u_i)$ (and so $m_{A_G} = m_{I_G}$).

4. CODINGS AND HOMOMORPHIC IMAGES OF TOL LANGUAGES.

In this section we prove that each ETOL language is a coding of a TOL language, and then using previously known results we prove that $\mathcal{F}_{ETOL} = \mathcal{F}_{HTOL} = \mathcal{F}_{CTOL}$.

First we need the following construct.

Construction 1. Let $G = \langle V_N, V_T, \mathcal{P}, S\rangle$ be an EPTOL system and let k be a nonnegative integer such that $k < m_G$. Let $\mathcal{A}_G = \langle Q, \Sigma, \delta, q_0, F\rangle$ where $Q = \{u_0,\dots,u_p\}$ and let $\mathcal{I}_G = \langle \overline{Q}, \overline{\Sigma}, \overline{\delta}, \overline{q}_0, \overline{F}\rangle$.

Let Ax(G, k) be the set of all nonempty words w over an alphabet $\bigcup_{\overline{q} \in \overline{Q}} \{\overline{q}\}$ such that $w = [b_1, j]\dots[b_t, j]$ where there exists a derivation of $b_1\dots b_t$ in G with a control sequence $\tau = P_{i_1}\dots P_{i_{m_G}}$ for which $\delta(q_0, \tau) = u_j$ and $(m_G + k) \in \mathrm{Spec}(\mathcal{A}_G, u_j)$.

For every r, s in $\{0,\dots,p\}$ such that $(m_G + k) \in \mathrm{Spec}(\mathcal{A}_G, u_r)$ and $(m_G + k) \in \mathrm{Spec}(\mathcal{A}_G, u_s)$ let $\mathcal{P}^{r,s,k}$ be the set of all compositions $P^{u_1,u_2} P^{u_2,u_3} \dots P^{u_{m_G},u_{m_G+1}}$ of m_G tables from $\overline{\Sigma}$ such that $u_1 = r$ and $u_{m_G} = s$. Let $\mathcal{P}^k$ be the set-theoretical union of all $\mathcal{P}^{r,s,k}$ for all r, s in $\{0,\dots,p\}$ such that $(m_G + k) \in \mathrm{Spec}(\mathcal{A}_G, u_r)$ and $(m_G + k) \in \mathrm{Spec}(\mathcal{A}_G, u_s)$.

Let $\Sigma^k = \{[a, i] : a \in V_N \cup V_T \text{ and } i \in \{0,\dots,p\}\}$.

If y is a word over Σ^k, say $y = [c_1, i]\dots[c_s, i]$, then proj(y) is defined by $\mathrm{proj}(y) = c_1\dots c_s$.

If Ax(G, k) is not empty, then for every w in Ax(G, k) we define a PTOL system G(k, w) by $G(k, w) = \langle \Sigma^k, \mathcal{P}^k, w\rangle$. We also define M(G(k, w)) by $M(G(k, w)) = \{x \in V_T^+ :$ there exists a word y in L(G(k, w)) such $\mathrm{proj}(y) \overset{m_G + k}{\underset{G}{\Longrightarrow}} x\}$.

The languages M(G(k, w)) can be used to "generate" L(G) as follows.

Lemma 5. Let $G = \langle V_N, V_T, P, S\rangle$ be an EPTOL system and let $k < m_G$. Let $F_G = \{x \in V_T^+ : S \overset{\ell}{\underset{G}{\Longrightarrow}} x \text{ for some } \ell < 2m_G\}$. Then

$$F_G \cup \bigcup_{k < m_G} \bigcup_{w \in Ax(G, k)} M(G(k,w)) = L(G).$$

Proof.

Obviously $F_G \cup \bigcup_{k < m_G} \bigcup_{w \in Ax(G, k)} M(G(k, w)) \subseteq L(G)$.

Now, let us assume that x is in L(G).

If x can be derived in G in less than $2m_G$ steps, then x is in F_G and consequently x is in $F_G \cup \bigcup_{k < m_G} \bigcup_{w \in Ax(G, k)} M(G(k, w))$.

Thus let us assume that x is in L(G) and x is derived in G in at least $2m_G$ steps. Let $D = (S, x_1, \ldots, x_r = x)$ be a derivation of x in G with $r \geq 2m_G$ and let $T_1, \ldots, T_r$ be a control sequence of D. Let $r = \ell_r m_G + k_r$ for some ℓ_r in N^+ and k_r in $\{0, \ldots, m_G - 1\}$. Let $A_G = \langle Q, \Sigma, \delta, q_0, F\rangle$ where $Q = \{u_0, \ldots, u_p\}$.

Note that:

(i) x_{m_G} derives (in G) x in $(\ell_r - 1)m_G + k_r$ steps. But $(\ell_r - 1)m_G + k_r \geq m_G$ and so $(m_G + k_r) \in Spec(A_G, Min(x_{m_G}))$. Obviously $\delta(q_0, T_1 \ldots T_{m_G}) = Min(x_{m_G})$ and so, if $Min(x_{m_G}) = u_j$, for some j in $\{0, \ldots, p\}$, then $[b_1, j] \ldots [b_t, j]$ is in Ax(G, k), where $x_{m_G} = b_1 \ldots b_t$ for some $b_1, \ldots, b_t$ in $V_N \cup V_T$.

(ii) For every ℓ in $\{1, \ldots, \ell_r - 1\}$, we have $(\ell_r - 1)m_G + k_r \geq m_G$ and consequently $(m_G + k_r) \in Spec(A_G, Min(x_\ell))$.

(iii) Let $z = [b_1, j] \ldots [b_t, j]$ (see (i)) and let $u_i = Min(x_{\ell_r - 1})$. If $x_{\ell_r - 1} = c_1 \ldots c_s$ then $[c_1, i] \ldots [c_s, i] \in L(G(k, w))$. But $x_{\ell_r - 1} \overset{m_G + k}{\underset{G}{\Longrightarrow}} x$ and consequently $x \in M(G(k, z))$.

Hence x is in $F_G \cup \bigcup_{k < m_G} \bigcup_{w \in Ax(G, k)} M(G(k, w))$.

Thus $L(G) \subseteq F_G \cup \bigcup_{k < m_G} \bigcup_{w \in Ax(G, k)} M(G(k, w))$ and Lemma 5 holds.

Before we can prove that each language of the form $M(G(k, w))$ is a finite union of codings of TOL languages we need the following construction.

<u>Construction 2</u>. Let $G = \langle V_N, V_T, P, S\rangle$ be an ETOL system and let k be a nonnegative integer such that $k < m_G$. Let $A_G = \langle Q, \Sigma, \delta, q_0, F\rangle$, where $Q = \{u_0,\dots,u_p\}$, and let $I_G = \langle \overline{Q}, \overline{\Sigma}, \overline{\delta}, \overline{q_0}, \overline{F}\rangle$. Let $Ax(G, k) \neq \emptyset$, let w be in $Ax(G, k)$ and let $G(k, w) = \langle V, R, w\rangle$.

If B is a nonempty subset of $V_N \cup V_T$, then let $Seq(B, m_G + k)$ be the set of all compositions $T_1 \dots T_{m_G+k}$ of tables from $Clos(P)$ such that if x is a word over $(V_N \cup V_T)$ with $Min(x) = B$ then $T_1 \dots T_{m_G+k}(x)$ contains a word in V_T^+.

If a is in $V_N \cup V_T$ and τ is a sequence of tables from $Clos(P)$ then $Contr(a, \tau)$ is defined as the set of all words y in V_T^+ such that $a \overset{\tau}{\underset{G}{\Longrightarrow}} y$.

Let $Z = \{[a, i, \tau, b] : [a, i] \in V,\ \tau \in Seq(u_i, k) \text{ and } b \in V_T\} \cup \{\overline{[a, i, \tau, b]} : [a, i] \in V,\ \tau \in Seq(u_i, k) \text{ and } b \in V_T\}$.

Let T be in R. Let r, s in $\{0,\dots,p\}$ be such that T is in $P^{r,s,k}$ (see Construction 1). If τ is in $Seq(u_s, k)$ then we define T^τ to be a table consisting of the following productions:

$$\{\overline{[a, r, \rho, b]} \to [c_1, s, \tau, b_{11}][c_1, s, \tau, b_{12}]\dots\overline{[c_1, s, \tau, b_{1n_1}]}[c_2, s, \tau, b_{21}]\dots \dots\overline{[c_2, s, \tau, b_{2n_2}]}\dots[c_v, s, \tau, b_{v1}]\dots\overline{[c_v, s, \tau, b_{vn_v}]} : [a, r] \to [c_1, s]\dots \dots[c_v, s] \text{ is in } T,\ b_{11}\dots b_{2n_1} \in Contr(c_1, \tau),\ b_{21}\dots b_{2n_2} \in Contr(c_2, \tau),\dots, b_{v1}\dots b_{vn_v} \in Contr(c_v, \tau)\} \cup \{[a, r, \rho, b] \to \Lambda : [a, r, \rho, b] \in Z\}.$$

If $w = [e_1, i]\dots[e_g, i]$ then we define $W(w)$ to be the set consisting of all the words of the form

$$[e_1, i, \rho, b_{11}][e_1, i, \rho, b_{12}]\dots\overline{[e_1, i, \rho, b_{1n_1}]}[e_2, i, \rho, b_{21}]\dots\overline{[e_2, i, \rho, b_{2n_2}]}\dots \dots[e_g, i, \rho, b_{g1}]\dots[e_g, i, \rho, b_{gn_g}] \text{ where } \rho \in Seq(u_i, m_G+k) \text{ and } b_{11}\dots b_{1n_1} \in Contr(e_1, \rho),\dots,b_{g1}\dots b_{gn_g} \in Contr(e_g, \rho).$$

For every y in W(w) let G(k, w, y) be the TOL system $\langle Z, \mathcal{S}, y\rangle$, where $\mathcal{S}$ is the following set of tables:

$\mathcal{S} = \{T^{\tau}$: for some r, s in $\{0,\ldots,p\}$, T is in $\mathcal{P}^{r,s,k}$ and τ is in $Seq(u_s, k)\}$.

<u>Lemma 6</u>. Let G be an EPTOL system, $k < m_G$, $Ax(G, k) \neq \emptyset$ and let w be in Ax(G, k). Then there exist a finite set $\{H_1,\ldots,H_f\}$ of TOL systems and a coding h such that

$$M(G(k, w)) = \bigcup_{i=1}^{f} h(H_i).$$

<u>Proof</u>.

Let G be an EPTOL system, $k < m_G$, $Ax(G, k) \neq \emptyset$ and let w be in Ax(G, k). Let h be the coding from Z into V_T defined as follows:

$h([a, i, \rho, b]) = h(\overline{[a, i, \rho, b]}) = b$.

We leave to the reader the obvious, but tedious proof, (based on Lemmata 3, 4, 5 and 6) of the fact that

$$M(G(k, w)) = \bigcup_{y \in W(w)} h(L(G(k, w,y))).$$

Thus Lemma 6 holds.

The following two results are very easy to prove, and so we leave their proofs for the reader.

<u>Lemma 7</u>. If K is a finite language, then there exists a TOL system G and a coding h such that $K = h(L(G))$.

<u>Lemma 8</u>. If K is a language such that there exist a finite set $\{H_1,\ldots,H_f\}$ of TOL systems and a finite set of codings $\{h_1,\ldots,h_f\}$ such that

$K = \bigcup_{i=1}^{f} h_i(L(H_i))$, then there exist a TOL system G and a coding $\bar{h}$ such that $K = \bar{h}(L(G))$.

Now we can easily show that every EPTOL language is a coding of a TOL language.

Lemma 9. For every EPTOL language K there exist a TOL language L and a coding h such that K = h(L).

Proof.

This result follows directly from Lemmata 5 through 8.

The following obvious result is given without a proof.

Lemma 10. If a language L is a coding of a TOL language, then also the language $L \cup \{\Lambda\}$ is a coding of a OL language.

Arbitrary ETOL languages are handled now as follows.

Theorem 1. For every ETOL language K there exist a OL language L and a coding h such that K = h(L).

Proof.

This theorem follows from Lemmata 9 and 10 and from the known fact (see, e.g., [Rozenberg, a]) that for each ETOL system G there exists an EPTOL system H such that $L(G) - \{\Lambda\} = L(H)$.

Now we can prove the main result of this section.

Theorem 2. $\mathcal{F}_{ETOL} = \mathcal{F}_{CTOL} = \mathcal{F}_{HTOL}$.

Proof.

This follows from Theorem 1, from the known fact (see, e.g., [Rozenberg, a]) that the class of ETOL languages is closed with respect to homomorphic mappings and from the fact that each coding is a homomorphism.

In fact one may observe that the above result is a constructive one in the following sense (the next result is a sample of the three possible results of the same character).

Proposition 1. The classes F_{ETOL} and F_{CTOL} are effectively equal, meaning that

(i) there exists an algorithm which, given an arbitrary ETOL system G, produces a TOL system H and a coding h such that L(G) = h(L(H)), and

(ii) there exists an algorithm which, given an arbitrary TOL system and a coding h produces an EOL system H such that L(H) = h(L(G)).

Proof.

This result follows from the fact that all the constructions involved in the proof of Theorem 2 may be effectively performed (we leave to the reader the tedious, but straightforward proof of this result).

As an application of Theorem 2 we present a result concerning the sets of lengths of words in ETOL languages. We may point out here that investigating the sets of lengths of words in ETOL languages forms an active, and well motivated, research area (see, e.g., [Paz and Salamaa]). The following result, as far as we know, is a new result in this area.

Before we state it we need a definition.

Definition 12. The length set of an ETOL system G, denoted as Lg(G), is defined by $Lg(G) = \{n \in N: |w| = n \text{ for some } w \text{ in } L(G)\}$.

Corollary 1. A subset U of nonnegative integers is the length set of an ETOL if, and only if, it is the length set of a TOL system.

Proof.

This result follows directly from Theorem 1, from the fact that if x is a word and h is a coding defined for elements of Min(x), then $|x| = |h(x)|$ and from the fact that each TOL language is, by definition, also an ETOL language.

Let us recall, for the sake of completeness, the following result from [Ehrenfeucht and Rozenberg].

Theorem 3. A language is an EOL language if, and only if, it is a coding of a OL language.

Thus, within developmental systems, we have the following informal thesis:

If $\mathcal{G}$ is a class of systems in which a rewriting of a symbol is done independently of its context (like ETOL or EOL systems) then defining the language of a system G in $\mathcal{G}$ by intersecting the set of all words generated in G with the set of all words over a particular alphabet (thus using nonterminals) is equivalent, as far as the class of all languages obtained in $\mathcal{G}$ is concerned, to using "a coding table" to translate (only once) letter-by-letter all words generated in G.

The rest of this paper is devoted to a further discussion of this thesis. First we show that the result is "inherent" to systems with "context-free rewritings". For example, we show next that even a weaker result is not true for the so-called EIL systems, which are very much the same as EOL systems, except that rewriting of a letter in a word may depend on the context of the letter.

5. EIL SYSTEMS AND LANGUAGES.

In this section we give formal definitions of EIL systems and languages (see, e.g., [Herman and Rozenberg]).

Definition 13. Let k and ℓ be nonnegative integers. An E $-\langle k, \ell\rangle$-system is a construct $G = \langle V_N, V_T, P, g, \omega\rangle$, where

V_N is a finite alphabet (of nonterminal letters or symbols),

V_T is a finite nonempty alphabet (of terminal letters or symbols), such that $V_N \cap V_T = \emptyset$ (where $V_N \cup V_T$ is called the alphabet of G),

ω is an element of $(V_N \cup V_T)^+$ (the axiom of G),

g is an element which is not in $V_N \cup V_T$ (the marker of G),

P is a finite nonempty relation (set of productions of G)

$$P \subset (\Sigma \cup \{g\})^k \times \Sigma \times (\Sigma \cup \{g\})^\ell \times \Sigma^*$$

such that

1) if $\langle w_1, a, w_3, w_4\rangle \in P$,

then

1.1) if $w_1 = \bar{w}_1 g \bar{\bar{w}}_1$ for some $\bar{w}_1, \bar{\bar{w}}_1 \in (\Sigma \cup \{g\})^*$, then $\bar{w}_1 \in \{g\}^*$,

1.2) if $w_3 = \bar{w}_3 g \bar{\bar{w}}_3$ for some $\bar{w}_3, \bar{\bar{w}}_3 \in (\Sigma \cup \{g\})^*$, then $\bar{\bar{w}}_3 \in \{g\}^*$,

and

2) for every $\langle w_1, a, w_3\rangle \in (\Sigma \cup \{g\})^k \times \Sigma \times (\Sigma \cup \{g\})^\ell$, such that w_1 and w_3 satisfy conditions 1.1 and 1.2, there exists a w_4 in Σ^* such that $\langle w_1, a, w_3, w_4\rangle \in P$.

If $V_N = \emptyset$ then G is called a $\langle k, \ell\rangle$-system.

Any E-$\langle k, \ell\rangle$-system is also called an EIL system and an $\langle k, \ell\rangle$-system is also called an IL system.

If $\langle w_1, a, w_3, w_4\rangle$ is an element of P, then we denote this by $\langle w_1, a, w_3\rangle \underset{P}{\longrightarrow} w_4$ and we call $\langle w_1, a, w_3\rangle \to w_4$ a <u>production in</u> P.

For any nonnegative integer k, if $w = a_1 \ldots a_m$ is a word such that $a_1, \ldots, a_m$ are symbols and $m \geq k$, then $\mathrm{Suf}_k(w) = a_{m-k+1}\, a_{m-k+2} \ldots a_m$ and $\mathrm{Pref}_k(w) = a_1 \ldots a_k$.

<u>Definition 14</u>. If $G = \langle V_N, V_T, P, g, \omega\rangle$ is an E-$\langle k, \ell\rangle$- system, $x = a_1 \ldots a_m$ with $m \geq 1$, $a_1, \ldots, a_m \in V_N \cup V_T$, and y is in $(V_N \cup V_T)^*$, then we say that x <u>directly derives</u> y in G (denoted as $x \underset{G}{\Longrightarrow} y$) if, and only if

$$\langle g^k, a_1, \mathrm{Pref}\,(a_2 \ldots a_m g^\ell)\rangle \underset{P}{\longrightarrow} \alpha_1,$$
$$\langle \mathrm{Suf}_k(g^k a_1), a_2, \mathrm{Pref}_\ell(a_3 \ldots a_m g^\ell)\rangle \underset{P}{\longrightarrow} \alpha_2,$$
$$\vdots \qquad\qquad\qquad \vdots$$
$$\langle \mathrm{Suf}_k(g^k a_1 \ldots a_{m-1}), a_m, g^\ell\rangle \underset{P}{\longrightarrow} \alpha_m,$$

for some $\alpha_1, \alpha_2, \ldots, \alpha_m \in (V_N \cup V_T)^*$ such that $y = \alpha_1 \ldots \alpha_m$. As usually (see e.g., Definition 3) we define $\overset{+}{\underset{G}{\Longrightarrow}}$ and $\overset{*}{\underset{G}{\Longrightarrow}}$ as transitive, and reflexive and transitive closure of the relation $\underset{G}{\Longrightarrow}$, respectively.

<u>Definition 15</u>. Let $G = \langle V_N, V_T, P, g, \omega\rangle$ be an E-$\langle k, \ell\rangle$-system. The <u>language of G</u>, denoted as L(G), is defined by $L(G) = \{x \in (V_N \cup V_T)^* : \omega \overset{*}{\underset{G}{\Longrightarrow}} x\}$.

<u>Definition 16</u>. Let k and ℓ be nonnegative integers. If M is a nonempty language different from $\{\Lambda\}$ such that there exists an E-$\langle k, \ell\rangle$-system (a $\langle k, \ell\rangle$-system) G such that L(G) = M, then M is called an <u>E-$\langle k, \ell\rangle$-language</u> (a <u>$\langle k, \ell\rangle$-language</u>). Each E-$\langle k, \ell\rangle$-language ($\langle k, \ell\rangle$-language) is also called an <u>EIL language</u> (an <u>IL language</u>).

We shall use symbols $\mathcal{F}_{EIL}$ and $\mathcal{F}_{IL}$ to denote the classes of EIL and IL languages, respectively.

We end this section with an example of an EIL system and language.

Example 3.

$G = \langle \{S\}, \{a, b\}, \{\langle b, a, a\rangle \to \Lambda, \langle g, a, a\rangle \to a^2, \langle a, a, a\rangle \to a^2, \langle a, a, g\rangle \to a^2, \langle g, b, a\rangle \to ba, \langle g, S, g\rangle \to ba^5, \langle g, S, g\rangle \to a^2$ and $\langle x, y, z\rangle \to y$ otherwise$\}, g, S\rangle$ is an E-$\langle 1,1\rangle$-system such that $L(G) = \{a^{2^n} : n \geq 1\} \cup \{ba^{2^n+1} : n \geq 2\}$.

6. HOMOMORPHIC IMAGES OF IL LANGUAGES.

In this section we show that the class of EIL languages does not result from codings of IL languages. Then we discuss a situation when codings are replaced by arbitrary homomorphisms.

We shall use the following known fact (for its proof see, e.g., [Herman and Rozenberg, Chapter 10]).

Lemma 11. A language is recursively enumerable if, and only if, it is an EIL language.

But it is well-known (see, e.g., [Hopcroft and Ullman, Theorem 9.10]) that recursively enumerable languages are closed under an arbitrary homomorphic mapping. Thus we have the following result.

Corollary 2. A homomorphic image of an EIL language is an EIL language.

However, unlike for the situation in Section 4, one does not get all EIL languages by taking codings of IL languages. To prove this we need first the following result.

Lemma 12. If L is an infinite IL language, then there exists a positive integer constant C_L such that if $\{w_1, w_2, \ldots\}$ is any ordering of elements of L in such a way that, for $j > i$, $|w_j| \geq |w_i|$, then, for every $k > 1$, $|\omega_k| \leq C_L \cdot |\omega_{k-1}|$.

Theorem 5. There exist EIL languages which are not images under Λ-free homomorphisms of IL languages.

Proof.

By Lemma 11 each recursively enumerable language is an EIL language. Let L be any infinite recursively enumerable language such that no two different words in L are of the same length and if $\{x_1, x_2, \ldots\}$ is an ordering of L according to the growing length of words, then $|x_n| - |x_{n-1}| \geq 2^{2^n}$. (An example of such a language is $\{a^{2^{2n}} : n \geq 0\}$).

Let us assume to the contrary, that there exist a language K (over some alphabet Σ) and a Λ-free homomorphism h such that $h(K) = L$. Then, obviously K must be an infinite language and thus, by Lemma 12, for every word x in L there exist a word y in L such that $x \neq y$ and the absolute value of $|x| - |y|$ is not larger than $C_k \cdot D_h$ where C_k is a constant defined as in Lemma 12 and $D_h = \max \{|\alpha| : h(a) = \alpha \text{ for some } a \text{ in } \Sigma\}$. This however contradicts the definition of L.

Thus L is not an image of any IL language under a Λ-free homomorphism. Hence Theorem 5 holds.

However, if we allow weak codings, rather than codings only, the situation becomes very different.

Before we prove our main result concerning weak codings of IL languages we need some auxiliary results concerning Turing machines. From now on we assume the reader to be familiar with constructions of Turing machines, for example, in the scope of [Davis] and [Hopcroft and Ullman]. In particular, we will not define the notion of a Turing machine here and we shall use standard terminology concerning Turing machines (as given, for example, in the above references).

A language is recursively enumerable if, and only if, it can be generated by a Turing machine. We shall adopt here the following definition of the generation of languages by Turing machines. (In the following, if n is a natural number then $\hat{n}$ denotes its unary tape representation for a Turing machine, it is $n = |^{n+1}$, where $|$ is a special distinguished symbol).

<u>Definition 17</u>. An infinite recursively enumerable language $\{w_1, w_2, \ldots\}$ is said to be <u>generated by a Turing machine</u> T if, and only if, for every n in N^+, whenever T starts on the input of the form $\#\hat{n}\#$ (where # is an end marker) it halts with the input tape storing the word $\hat{n} * w_n$ only (where * is a distinguished symbol different from $|$).

The proof of the following result requires a rather standard construction so we leave it to the reader.

<u>Lemma 13</u>. For every recursively enumerable language W there exists a Turing machine T which generates W without repetitions, meaning that, if T starts on $\# \hat{n} \#$ and then halts storing $\hat{n} * w_n$, and $m \neq n$, then if T starts on $\# \hat{m} \#$ and then halts storing $\hat{m} * w_m$ then $w_n \neq w_m$.

<u>Lemma 14</u>. For every recursively enumerable language W (over some finite alphabet Σ) there exists a Turing machine T with two distinguished tape symbols S_0, S_f (which are not in Σ) such that whenever T starts on tape containing the word $\# S_0 \#$ only (where # is an end marker) then

(i) it will go through the infinite sequence τ of tape descriptions such that $S_0, S_f w_1, S_0 w_1, S_f w_2, \ldots$ is the subsequence of τ, where $w = \{w_1, w_2, \ldots\}$, and

(ii) T produces a tape description of the form $S_0 w$ or $S_f w$ only if w is in W.

Proof.

Let W be a recursively enumerable language. By Lemma 13 we may assume that there exists a Turing machine T' such that T' generates W without repetitions. Let $W = \{w_1, w_2, \ldots\}$ where the ordering on W corresponds to its generation by T' (hence if $i \neq j$ then $w_i \neq w_j$).

Now we construct a Turing machine T satisfying the statement of the Lemma as follows.

T starts in its initial state q_0 with tape description S_0 and then produces a tape description $S_f w_1$ (this obviously can be done). Then it changes S_f to S_0 and we get a tape description $S_0 w_1$ and an instantaneous description $q_0\ S_0\ w_1$.

Now, if the instantaneous description of T at the given moment is $q_0\ S_0\ w_k$ for some $k \geq 1$, then T does the following. It puts a special marker ¢ next to the rightmost symbol of w_k and then to the right of this marker it produces (employing T') consecutively words w_1, w_2, w_3, etc. After w_j is produced (for $j \geq 1$) then it is compared with w_k which is to the left of the marker ¢. If a comparison is successful, then T first produces $S_f\ n_{k+1}$ (using T' to produce the next word in the sequence, which is now w_{k+1}) and then $S_0\ w_{k+1}$, ending in an instantaneous description $q_0\ S_0\ w_{k+1}$.

Again, T is ready for the next iteration.

It is obvious that T satisfies the conditions from the statement of Lemma 14 and hence the result holds.

For our next result we assume the reader to be familiar with simulation of Turing machine by IL systems, say $\langle 1, 1 \rangle$-systems, (see, e.g., [von Dalen] and [Herman]) and with the so-called "firing squad synchronomization problem" (see, e.g., [Balzer] and [Herman, Liu, Rowland and Walker]). We are assuming this, and we present the proof of the next result rather informally, as oterhwise the size of the formal proof of the next result would by itself exceed the size of this paper. The idea of the proof is, however, simple and clear, so we do feel that an informal proof will suffice.

Theorem 6. For every EIL language L there exist an IL language K and a weak coding h such that $h(K) = L$.

Proof.

Let L be an EIL language.

If L is a finite language then the result is trivially true. Thus we may assume that L is an infinite language.

By Theorem 4, L is a recursively enumerable language and (see Lemma 14) it can be generated by a Turing machine T satisfying conditions of Lemma 14.

Now let G' be a $\langle 1, 1\rangle$-system simulating T step-by-step. Obviously, we may assume that G' has a production $\langle g, S_f, x\rangle \to S_0$ for every x in the alphabet of G'.

We construct another $\langle 1, 1\rangle$-system G which performs as follows. Productions of the form $\langle g, S_f, x\rangle \to S_0$ for x in the alphabet of G', are replaced now (in G) by production $\langle g, S_f, s\rangle \to \overline{S}_f$, where $\overline{S}_f$ is a new distinguished symbol. $\overline{S}_f$ is "sending a messenger" towards the right end of a string in order to change successively each symbol x in the string into the new "two-component" symbol of the form $[x, \sigma]$ where σ is a fixed symbol. After the whole string $x_1 \ldots x_f$ is changed into the string $[x_1, \sigma] \ldots [x_f, \sigma]$ the messenger "goes back" and we obtain the string of the form $\overline{\overline{S}}_f\ [x_1, \sigma] \ldots [x_f, \sigma]$ where $\overline{\overline{S}}_f$ is a new distinguished symbol. Now the string of the second components (σ) of symbols $[x_i, \sigma]$ together with $\overline{\overline{S}}_f$ on the left end form the "firing squad" with $\overline{\overline{S}}_f$ being the general and σ's being the soldiers.

G acts now as to achieve the synchronization of this firing squad in such a way that after a finite number of steps the string $\overline{\overline{S}}_f\ [x_1, \sigma]\ldots[x_f, \sigma]$ will turn into the string $\overline{x}_1, \overline{x}_2, \ldots, \overline{x}_f$, where $\overline{x}_1, \overline{x}_2, \ldots, \overline{x}_f$ are new letters being in one-to-one correspondence with $x_1, \ldots, x_f$, and all intermediate strings contain a two-component symbols only. Now G (using productions $\langle g, \overline{x}, y\rangle \rightarrow S_0\ x$ and $\langle z, \overline{x}, y\rangle \rightarrow x$ for every $z \neq g$ and for every y in the alphabet of G) changes the string $\overline{x}_1\ldots\overline{x}_f$ in one step into the string $S_0\ x_1\ldots x_f$.

Thus a simple step $S_f\ x_1\ldots x_f \underset{G'}{\Longrightarrow} S_0\ x_1\ldots x_f$ is replaced in G by the whole whole sequence of steps $S_f\ x_1\ldots x_f \underset{G}{\Longrightarrow} \ldots \underset{G}{\Longrightarrow} \overline{x}_1\ldots\overline{x}_f \underset{G}{\Longrightarrow} S_0\ x_1\ldots x_f$ where (because of the synchronization) no intermediate string in this derivation contains an occurrence of a "barred" symbol $(\overline{x})$.

Because of this property of derivations, in G it is obvious that if h is a weak coding such that

$h(\overline{a}) = \overline{a}$, for each barred symbol $\overline{a}$, and

$h(x) = \Lambda$, otherwise

then $h(L(G)) - \{\Lambda\} = L$.

The last thing to be done is to modify G to $\overline{G}$ in such a way that $h(L(\overline{G})) = L$. Note that even if Λ is not in L, then Λ occurs in h(L(G)) because G generates strings consisting of "nonbarred" letters only.

To remove this obstacle one may apply a very simple trick indeed. Let $z = b_1\ldots b_f$ be a nonempty word from L (where $b_1, \ldots, b_f$ are symbols from the alphabet of L). A $\langle 1, 1\rangle$-system $\overline{G}$ will contain new "extra" markers $*_1, *_2, \ldots, *_f$ which will be used in such a way that they are erased whenever the general $\overline{\overline{S}}_f$ is erased, and whenever $\overline{\overline{S}}_f$ is generated in G then the string $*_1 *_2\ldots *_f\ \overline{\overline{S}}_f$ is generated in $\overline{G}$.

Now if h is extended to the weak coding $\overline{h}$ defined also for symbols $*_1, \ldots, *_f$ in such a way that $\overline{h}(*_i) = b_i$ for $1 \leq i \leq f$, then it is obvious that $h(L(\overline{G})) = L$. (Note that whenever the nonempty string y generated in G is such that $h(y) = \Lambda$ then a string $*_1 \ldots *_f\ y$ is generated in $\overline{G}$ and $\overline{h}(*_1 \ldots *_f\ y) = z$).

Thus the Theorem holds.

Now we can prove the main result of this section.

Theorem 7. $F_{CIL} \subset F_{EIL} = F_{WIL} = F_{HIL}$.

Proof.

This result follows directly from Corollary 2, Theorem 5, Theorem 6 and from the fact that each weak coding is also a homomorphism.

Again one may easily check the effective character of the equalities from Theorem 7 (we leave this to the reader). Thus we have, for example, the following result (which is a sample of three possible results).

Proposition 2.

(i) There exists an algorithm which given an arbitrary IL system G and a weak coding h will produce an EIL system H such that $L(H) = h(L(G))$.

(ii) There exists an algorithm which given an arbitrary EIL system G will produce an IL system H and a weak coding h such that $L(G) = h(L(H))$.

So far our results sum up into the following informal thesis:

(i) In the class of developmental systems with "context-free" rewritings (EOL and ETOL systems) the use of nonterminal symbols is equivalent to the use of coding tables.

(ii) In the class of developmental systems with "context-dependent" rewritings (EIL systems) the use of nonterminal symbols cannot be replaced by coding tables, whereas the use of coding table (even "homomorphic" tables) can be replaced by nonterminal symbols. However, the use of nonterminal symbols in this class of developmental systems is equivalent to the use of weak coding tables.

In the next section we discuss the trade-off between coding (homomorphic) tables and nonterminals in the class of context-free and context-sensitive grammar from the classical Chomsky hierarchy.

7. HOMOMORPHIC IMAGES OF SENTENTIAL FORMS OF CONTEXT-FREE AND CONTEXT-SENSITIVE GRAMMARS.

In this section we discuss the trade-off between the use of nonterminal symbols and coding (homomorphic) tables in the classes of grammars of the Chomsky hierarchy. We shall adopt here the notation concerning context-free and context-sensitive grammars as given in [Hopcroft and Ullman] with the following modification.

As it is almost always required that the axiom of a context-free or context-sensitive grammar is a single symbol, if one takes codings of the languages of sentential forms of such grammars, then each such language contains a word of length one. In this way one could immediately produce a trivial proof of the fact that the class of codings of, say, sentential forms of context-free languages is not identical to the class of context-free languages (not every context-free language contains a word of length one). To avoid trivialities of this kind we adopt here two conventions:

(i) <u>the axiom of a context-free or a context-sensitive grammar may be an arbitrary nonempty string over the total alphabet</u> (terminals and nonterminals) <u>of the given grammar</u>, and

(ii) <u>neither the empty set nor $\{\Lambda\}$ is a context-free or a context-sensitive language</u>.

We shall use symbols $\mathcal{F}_{CF}$, $\mathcal{F}_{SCF}$, $\mathcal{F}_{CS}$ and $\mathcal{F}_{SCS}$ to denote the classes of context-free languages, sentential forms of context-free grammars, context-sensitive languages and sentential forms of context-sensitive grammars.

The first class to be investigated is the class of context-free grammars (with the modification explained above).

The following result is well-known (see, e.g., [Hopcroft and Ullman, Theorem 9.7]).

Lemma 15. If L is a context-free language over an alphabet Σ and h is a homomorphism from Σ, then h(L) is also a context-free language.

The easy proof of the next result we leave to the reader.

Lemma 16. If L is a set of sentential forms of a context-free or context-sensitive grammar, then it is a context-free or context-sensitive language respectively.

The following is an interesting result on its own.

Theorem 8. There exist finite language which are not codings of sentential forms of context-free grammars.

Proof.

Let $L = \{ab, cd\}$

Let us assume that there exist a context-free grammar $G = \langle V_N, V_T, P, \omega \rangle$ and a coding h such that h(Sent(G)) = L. Then Sent(G) must consist of two words, say AB and CD, such that h(A) = a, h(B) = b, h(C) = c, h(D) = d (thus all the letters A, B, C, D are different) and either $AB \underset{G}{\Longrightarrow} CD$ or $CD \underset{G}{\Longrightarrow} AB$. This would mean however that in one direct derivation step in G two symbols must be rewritten; a contradiction.

Theorem 9. For each context-free language L there exist a context-free grammar G and a Λ-free homomorphism h such that L = h(Sent(G)).

Proof.

Let L be a context-free language.

Thus L is generated by a context-free grammar $G = \langle V_N, V_T, P, S \rangle$ which is such that either $\Lambda \notin L$ and no production in P is of the form $A \to \Lambda$, or $\Lambda \in L$

and the only production in P with Λ on its right-hand side is the production $S \to \Lambda$, where S does not appear on the right-hand side of any production. It is well-known (see, e.g., [Hopcroft and Ullman, Theorem 4.3]) that we may assume that for each A in V_N there exists a string w in V_T^+ such that $A \overset{*}{\underset{G}{\Longrightarrow}} w$. Let, for each A in V_N, w_A be an arbitrary but fixed string over V_T^+ such that $A \overset{*}{\underset{G}{\Longrightarrow}} w_A$.

We leave to the reader the easy proof of the fact that if h is a Λ-free homomorphism from $(V_N \cup V_T)^*$ into V_T^* defined by:

$h(A) = w_A$, for every A in V_N, and

$h(a) = a$, for every a in V_T,

then $h(\mathrm{Sent}(G)) = L(G) = L$.

Thus Theorem 9 holds.

Theorem 10.

(i) $\mathcal{F}_{CSCF} \subset \mathcal{F}_{CF}$.

(ii) $\mathcal{F}_{PHSCF} = \mathcal{F}_{HSCF} = \mathcal{F}_{CF}$.

Proof.

(i) follows from Lemmata 15, 16 and Theorem 8, and (ii) follows from Lemmata 15, 16, and Theorem 9.

Lemma 17. If L is an infinite language of sentential forms of a context-sensitive grammar and $\{w_1, w_2, \ldots\}$ is any ordering of elements of L such that $|w_j| \leq |w_i|$ for $j > i$, then, for every $k > 1$, $|w_k| - |w_{k-1}| < F_L$ where F_L is a positive integer constant dependent on L only.

Theorem 11. There exist a context-sensitive language K for which there does not exist a context-sensitive grammar G and a Λ-free homomorphism h such that $K = h(L(G))$.

Proof.

Let $K = \{a^{2^n} : n \geq 0\}$. Obviously, K is a context-sensitive language.

Let us assume that there exist a context-sensitive grammar G and a Λ-free homomorphism h such that $h(Sent(G)) = K$. Let $U = \max\{|h(a)|:$ a is in the alphabet of G$\}$ and let $F_{Sent(G)}$ be a constant from the statement of Lemma 17. But if we order K according to the (increasing) length of words, $K = \{a, a^2, a^4, \ldots\}$, then from Lemma 17 we have that, for every $j > 1$, $2^j - 2^{j-1} < U \cdot F_{Sent(G)}$; a contradiction.

Thus Theorem 11 holds.

The following is a well-known result (see, e.g., [Hopcroft and Ullman, Theorems 9.9 and 9.11]).

Theorem 12.

(i) The class of context-sensitive languages is closed under Λ-free homomorphic mappings.

(ii) The class of context-sensitive languages is not closed under arbitrary homomorphic mappings.

Theorem 13.

(i) $\mathcal{F}_{PHSCS} \subset \mathcal{F}_{CS}$.

(ii) Neither $\mathcal{F}_{HSCS} \subset \mathcal{F}_{CS}$ nor $\mathcal{F}_{CS} \subset \mathcal{F}_{HSCS}$.

Proof.

This result follows from Theorems 11, 12 and Lemma 16.

GR:cah

REFERENCES.

Balzer, R., The firing squad synchronization problem, Information and Control, 1967, v. 10, 22-42.

Chomsky, N., Three models for the description of language, IRE Trans. Inform. Theory, 1956, v. IT2, pp. 113-124.

Chomsky, N., Syntactic Structures, Mouton and Company, The Hague, 1957.

Davis, M., Computability and Unsolvability, McGraw-Hill Book Company, New York, 1958.

van Dalen, D., A note on some systems of Lindenmayer, Mathematical Systems Theory, 1971, v. 5, 128-140.

Ehrenfeucht, A., and Rozenberg, G., Codings of OL languages, submitted for publication.

Ginsburg, S., and Greibach, S., Abstract families of languages, IEEE Conf. Record on 8th Annual Symp. on Switching and Automata Theory, 1967.

Herman, G., The computing ability of a developmental model for filamentous organisms, Journal of Theoret. Biology, 1969, v. 25, 421-435.

Herman, G., Closure properties of some families of languages associated with biological systems, Information and Control, to appear.

Herman, G., Lindenmayer, A., and Rozenberg, G., Description of developmental languages using recurrence systems, submitted for publication.

Herman, G., and Rozenberg, G., Developmental systems and languages, to be published by North-Holland Publishing Company.

Herman, G., Liu, W., Rowland, S., and Walker, A., Synchronization of growing cellular arrays, Quarterly Bulletin of the Center for Theoretical Biology, SUNY at Buffalo, 1972, v. 5, No. 2, 143-196.

Hopcroft, J., and Ullman, J., Formal languages and their relation to automata, Addison-Wesley, 1969.

Lindenmayer, A., Mathematical models for cellular interactions in development, Journal of Theoret. Biology, 1968, v. 18, 300-315.

Lindenmayer, A., and Rozenberg, G., Developmental systems and languages, Proc. of the 4th ACM Symp. of Th. of Computing, 1972.

Paz, A., and Salomaa, A., Integral sequential word function and growth equivalence of Lindenmayer systems, Information and Control, to appear.

Rozenberg, G., (a), Extension of tabled OL systems and languages, International Journal of Comp. and Inf. Sciences, to appear.

Rozenberg, G., (b), TOL systems and languages, Information and Control, to appear.

Rozenberg, G., and Doucet, P., On OL languages, Information and Control, 1971, v. 19, 302-318.

ACKNOWLEDGEMENT

The authors are indebted to Dr. H. Paul Zeiger for his comments on the original manuscript.