Theoretical Computer Science 62 (1988) 235-249
North-Holland

A REPRESENTATION OF RECURSIVELY ENUMERABLE LANGUAGES BY TWO HOMOMORPHISMS AND A QUOTIENT*

Viliam GEFFERT
Department of Computer Science, University of P.J. Šafárik, Jesenná 5, 04154 Košice, Czechoslovakia

Communicated by A. Salomaa
Received February 1987
Revised October 1987

Abstract. A new representation for recursively enumerable languages is presented. It uses a pair of homomorphisms and the left (or right) quotient: For each recursively enumerable language L one can find homomorphisms $h_1, h_2: \Sigma_A^* \to \Sigma_B^*$, such that $w \in \Sigma_L^*$ is a word in L if and only if $w = h_1(\alpha)\backslash h_2(\alpha)$ for some $\alpha \in \Sigma_A^+$. (Or, each recursively enumerable language can be given by $L = O(h_1\backslash h_2) \cap \Sigma_L^*$, where $O(h_1\backslash h_2)$ is the so-called right overflow language defined as $O(h_1\backslash h_2) = \{h_1(x)\backslash h_2(x); x \in \Sigma_A^+\}$.)

1. Introduction

There exist many different representations for recursively enumerable sets. We should mention Turing machines, phrase-structure grammars, and many other different models of algorithms, which are capable of representing the recursively enumerable sets. (See [5, 6, 7, 11] for the definitions of mathematical and language notions used in this paper.) It is of interest to have also a representation based on some simple algebraic operations. We shall present the representation using two homomorphisms, intersection with Σ_L^*, and a quotient (an operation inverse to concatenation): For each recursively enumerable language L there exists a pair of homomorphisms $h_1, h_2: \Sigma_A^* \to \Sigma_B^*$ (where Σ_A, Σ_B are some alphabets, $\Sigma_B \supseteq \Sigma_L$) such that

$$\begin{aligned} L &= O(h_1\backslash h_2) \cap \Sigma_L^* \\ &= \{w \in \Sigma_L^*; w = h_1(\alpha)\backslash h_2(\alpha) \text{ for some } \alpha \in \Sigma_A^+\}, \end{aligned}$$

where $O(h_1\backslash h_2)$ is the so-called right overflow language of two homomorphisms $h_1, h_2: \Sigma_A^* \to \Sigma_B^*$, defined as $O(h_1\backslash h_2) = \{h_1(x)\backslash h_2(x); x \in \Sigma_A^+\}$.

Section 2 deals with the proof of this main result. The proof is based on the notion of a g-system, originally introduced by Rovan [8] in order to unify the theory of grammars. Section 3 concerns the complexity of this form of representation, and

* This research was supported in part by Research Grant SPZV I-1-5/08.

Section 4 discusses some extensions, for example, a modification of the above representation for context-sensitive languages. The results of the paper should be compared with some older similar characterizations; for example, the result of [1] is that for every recursively enumerable language L there exists an erasing h_0 (a homomorphism either preserving or erasing any symbol), and a pair of homomorphisms h_1, h_2 such that $L = h_0(e(h_1, h_2))$, where $e(h_1, h_2)$ is so-called minimal equality set: $e(h_1, h_2) = \{w \in \Sigma_A^+; h_1(w) = h_2(w)$ and $h_1(u) \neq h_2(u)$ for each proper nonempty prefix u of $w\}$. The time and space complexity questions concerning this characterization (similar to those in Section 3) can be found in [2]. In [3, 10] some other problems are presented concerning homomorphism equivalence and equality sets (the sets of words on which h_1 and h_2 agree, i.e., $E(h_1, h_2) = \{w \in \Sigma_A^*; h_1(w) = h_2(w)\}$).

2. The homomorphic representation

Let us begin with the definition of a g-system [8], which is a generalization of the notion of a grammar. (The definitions of grammar, sentential form, rewriting relation, etc. can be found in [6, 7, 11].) Rewriting of the sentential form is performed by a 1-a-transducer [5]. The resulting word is obtained by an iterative rewriting by this transducer, starting from an initial symbol. Most of the known types of grammars can be naturally defined as special cases of g-systems.

Definition. *A generative system* (*g-system*, for short) is a quadruple $G = (N, T, P, S)$, where N and T are finite alphabets of nonterminal and terminal symbols (not necessarily disjoint), S in N is an initial symbol, and P is a finitely specified binary relation over $V^+ \times V^*$ (where $V = N \cup T$).

P (the rewriting relation) is given in the form of a 1-a-transducer [5] (from V^+ to V^*), i.e., $P = (K, V, V, H, q_I, q_F)$, where K is a finite set of states, q_I, q_F in K are initial and final states respectively, and H is a finite subset of $K \times V \times V^* \times K$ (the set of *transitions*, or *edges*).

We shall use the following notation:

$u \Rightarrow v$ means that P is able to rewrite u to v, i.e., there exists a path of transitions

$$(q_I, s_1, v_1, q_1)(q_1, s_2, v_2, q_2)\ldots(q_{n-1}, s_n, v_n, q_F) \in H^+,$$

such that $s_1 s_2 \ldots s_n = u$, and $v_1 v_2 \ldots v_n = v$.

$\Rightarrow^*$ denotes the reflexive and transitive closure of $\Rightarrow$.

Finally, *a language generated by G is*

$$L(G) = \{w \in T^*; S \Rightarrow^* w\}.$$

Now we shall show informally that g-systems are capable of generating any recursively enumerable language.

Theorem 2.1 (Rovan [8]). *Let G be a phrase-structure grammar, $G=(N, T, P, S)$. Then there exists a g-system G' such that $L(G)=L(G')$.*

Proof. We need to show, how a 1-a-transducer can imitate a rewriting step in G by the rule $A_1 \ldots A_n \to v \in P$.

The transitions shown in Fig. 1 are needed in H (they are given graphically, in the form usual in the theory of automata). (We use some new distinct states $q_1, \ldots, q_{n-1}$ for each rule $A_1 \ldots A_n \to v \in P$, so only the initial and final states are shared by different paths for different rules.)

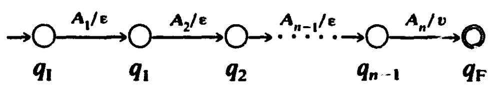

Fig. 1.

It is easily seen that this set of transitions rewrites $A_1 \ldots A_n$ to v. Moreover, we shall add so-called copying cycles to H, i.e., transitions $(q, x, x, q) \in H$, for each $x \in V = N \cup T$, and $q \in \{q_I, q_F\}$. This gives us $u_1 A_1 \ldots A_n u_2 \Rightarrow_{G'} u_1 v u_2$, for each $u_1, u_2 \in V^*$. □

It is easy to see that for each g-system G there exists an equivalent g-system G' (i.e., generating the same language) such that the initial and final states of its 1-a-transducer are distinct. In what follows we shall therefore assume (for technical reasons) that in each g-system $q_I \neq q_F$. We are now ready to establish the main result, namely, the representation of recursively enumerable languages by a pair of homomorphisms and the left quotient. As will be shown later, we can use the right quotient as well. *The quotients* are understood as operations inverse to concatenation, i.e., $u \backslash uv = v$, $uv/v = u$, for each u, v. This implies that $v \backslash w$ is defined only if v is a prefix of w; an analogous condition holds for v/w. These operations can be extended to languages, for example, $L_1/L_2 = \{u/v; u \in L_1, v \in L_2\}$. Now we can define *overflow languages of the homomorphisms* h_1, h_2:

$$O(h_1 \backslash h_2) = \{h_1(x) \backslash h_2(x); x \in \Sigma_A^+\}, \qquad O(h_1/h_2) = \{h_1(x)/h_2(x); x \in \Sigma_A^+\}.$$

The proof of the following theorem is relatively long and technical. But, to obtain an idea how the mechanism works, it is sufficient to read only part (i) at first reading.

Theorem 2.2. *For each effectively given recursively enumerable language L, one can effectively construct a pair of homomorphisms $h_1, h_2 : \Sigma_A^* \to \Sigma_B^*$ such that*

$$\begin{aligned} L &= O(h_1 \backslash h_2) \cap \Sigma_L^* \\ &= \{w \in \Sigma_L^*; w = h_1(\alpha) \backslash h_2(\alpha) \quad \text{for some } \alpha \in \Sigma_A^+\}. \end{aligned}$$

(Here Σ_A, Σ_B are some alphabets, $\Sigma_L \subseteq \Sigma_B$.)

Thus, $w \in \Sigma_L^*$ is a word in L if and only if there exists some $\alpha \in \Sigma_A^+$ such that $h_2(\alpha) = h_1(\alpha) w$

Proof. We may use Theorem 2.1 and assume that the recursively enumerable language L is given by some g-system. Let $L=L(G)$ for some g-system $G=(N, T, P, S)$, where P is a 1-a-transducer, i.e., $P=(K, V, V, H, q_I, q_F)$. ($V=N\cup T$, and $T=\Sigma_L$.)

As mentioned above, we shall assume without loss of generality that $q_I \neq q_F$. (The proof could be done even without this assumption, but it would become more complicated.) We can also assume that H, K, V, and $K\times V$ are pairwise disjoint sets. Define

$$\Sigma_A=\{a_0, a_1, a_2, a_3\}\cup K\times V\cup H,$$

$$\Sigma_B=\{b_0, b_1, b_2\}\cup K\cup V\cup K\times V,$$

where $a_0, a_1, a_2, a_3, b_0, b_1, b_2$ are new symbols. We now define h_1 and h_2 as shown in Table 1.

Table 1

$x\in\Sigma_A$	$h_1(x)$	$h_2(x)$	Remark
a_0	b_0	b_0b_1S	
a_1	b_1	b_2	
a_2	b_2q_I	q_Fb_1	
a_3	b_1	ε	
(q, A)	A	$q(q, A)$	for each $x\in K\times V$
(q, A, v, q')	$(q, A)q'$	v	for each $x\in H$

(i) First, we are going to prove that if $w\in L(G)$, then $w=h_1(\alpha)\backslash h_2(\alpha)$ for some $\alpha\in\Sigma_A^+$. This will also give us an idea how this system of homomorphisms can imitate the derivation in g-system G.

Claim 1. *If $S\Rightarrow_G^* w$, then there exists an $\alpha\in\Sigma_A^+$ such that*

$$h_2(\alpha)=h_1(\alpha)b_1w. \tag{2.1}$$

Proof. We proceed by induction on the length of a derivation in G:

(A) If $w=S$ (the length of derivation is zero), then (2.1) holds for $\alpha=a_0$:

$$\begin{aligned} h_1(\alpha)&: \quad b_0,\\ h_2(\alpha)&: \quad b_0b_1S.\end{aligned}$$

(B) Now assume Claim 1 holds for derivation of length k. Let $S\Rightarrow_G^* u\Rightarrow_G w$ be a derivation of length $k+1$. We have, by the induction hypothesis, $\alpha'\in\Sigma_A^+$ such that

$$h_2(\alpha')=h_1(\alpha')b_1u. \tag{2.2}$$

Since $u\Rightarrow_G w$, there exists a sequence of transitions

$$(k_1, s_1, v_1, k_1'), \ldots, (k_n, s_n, v_n, k_n')$$

in H^+ such that

$$\begin{aligned} &u=s_1\dots s_n, && n>0,\ s_i\in V \text{ for } i=1,\dots,n,\\ &w=v_1\dots v_n, && v_i\in V^* \text{ for } i=1,\dots,n,\\ &k_1=q_1, && \\ &k_i'=k_{i+1} && \text{for } i=1,\dots,n-1,\\ &k_n'=q_F. && \end{aligned} \tag{2.3}$$

We shall now construct α by successively appending certain letters to α', thus obtaining a sequence of words α_1, α_2, α_3, α_4, and $\alpha_5=\alpha$. First, let $\alpha_1=\alpha'$. By (2.2), the string $h_1(\alpha_1)$ is a prefix of $h_2(\alpha_1)$, and $h_1(\alpha_1)\backslash h_2(\alpha_1)=b_1u$:

$$\alpha_1=\alpha'$$

$$\begin{aligned} &h_1(\alpha_1): && h_1(\alpha'),\\ &h_2(\alpha_1): && h_1(\alpha')b_1\underbrace{s_1\dots s_n}_{u}. \end{aligned}$$

Let us append a_1 to α_1, i.e., $\alpha_2=\alpha' a_1$. This is illustrated as follows:

$$\alpha_2=\alpha' a_1$$

$$\begin{aligned} &h_1(\alpha_2): && h_1(\alpha')b_1,\\ &h_2(\alpha_2): && h_1(\alpha')b_1s_1\dots s_nb_2. \end{aligned}$$

Now we extend α_2 by $(k_1,s_1)\dots(k_n,s_n)$:

$$\alpha_3=\alpha' a_1(k_1,s_1)\dots(k_n,s_n)$$

$$\begin{aligned} &h_1(\alpha_3): && h_1(\alpha')b_1s_1\dots s_n,\\ &h_2(\alpha_3): && h_1(\alpha')b_1s_1\dots s_nb_2k_1(k_1,s_1)\dots k_n(k_n,s_n). \end{aligned}$$

Next, we append the symbol a_2:

$$\alpha_4=\alpha' a_1(k_1,s_1)\dots(k_n,s_n)a_2$$

$$\begin{aligned} &h_1(\alpha_4): && \dots b_1s_1\dots s_nb_2q_1,\\ &h_2(\alpha_4): && \dots b_1s_1\dots s_nb_2k_1(k_1,s_1)\dots k_n(k_n,s_n)q_Fb_1. \end{aligned}$$

$h_1(\alpha_4)$ still remains a prefix of $h_2(\alpha_4)$, because $q_1=k_1$ by (2.3). Finally, let us append $(k_1,s_1,v_1,k_1')\dots(k_n,s_n,v_n,k_n')$:

$$\alpha_5=\alpha' a_1(k_1,s_1)\dots(k_n,s_n)a_2(k_1,s_1,v_1,k_1')\dots(k_n,s_n,v_n,k_n')$$

$$\begin{aligned} &h_1(\alpha_5): && \dots b_2q_1(k_1,s_1)k_1'(k_2,s_2)k_2'\dots(k_n,s_n)k_n'\\ &h_2(\alpha_5): && \dots b_2k_1(k_1,s_1)k_2(k_2,s_2)\dots k_n(k_n,s_n)q_Fb_1v_1v_2\dots v_n. \end{aligned}$$

By (2.3) we have $k_i'=k_{i+1}$ for $i=1,\dots,n-1$, $k_n'=q_F$, and also $v_1\dots v_n=w$. Then $h_1(\alpha_5)$ is still a prefix of $h_2(\alpha_5)$. Note that the strings $\alpha=\alpha_5$ and $w=v_1\dots v_n$ again satisfy (2.1) and Claim 1 is verified. $\square$

We are now ready to show that for each $w \in L(G)$ there exists an $\alpha \in \Sigma_A^+$ such that $w = h_1(\alpha)\backslash h_2(\alpha)$: The existence of $\alpha' \in \Sigma_A^+$ such that $h_2(\alpha') = h_1(\alpha')b_1 w$ follows from Claim 1. Let $\alpha = \alpha' a_3$. Clearly, $h_2(\alpha) = h_1(\alpha)w$, i.e., $w = h_1(\alpha)\backslash h_2(\alpha)$.

Moreover, we have also proved that having a derivation

$$S = w_0 \Rightarrow_G w_1 \Rightarrow_G \cdots \Rightarrow_G w_m \Rightarrow_G w,$$

α can always be chosen in the form

$$a_0\left(\prod_{i=1}^{m} (a_1(K \times V)^{n_i} a_2 H^{n_i})\right)a_3, \tag{2.4}$$

where $n_i = |w_i|$. This proves (i).

(ii) (*Only-if*): Now we need to show that the *only* way of forming an α with the desired properties is as shown in (i), i.e., if $h_1(\alpha)\backslash h_2(\alpha) = w \in V^*$, then $S \Rightarrow_G^* w$. More exactly, we shall show that

- if $h_1(\alpha)$ is a prefix of $h_2(\alpha)$, then α has to be a prefix of some word in $a_0(a_1(K \times V)^n a_2 H^n)^* a_3$;
- moreover, if $\alpha \in a_0(a_1(K \times V)^n a_2 H^n)^*$, then $h_1(\alpha)\backslash h_2(\alpha) = b_1 w$ for some w derivable from S;
- if α terminates in a_3, then $h_1(\alpha)\backslash h_2(\alpha) = w$, for some w derivable from S. Otherwise, $h_1(\alpha)\backslash h_2(\alpha) \notin \Sigma_L^*$.

We have to prove some claims first.

Claim 2. *If $h_1(\alpha)$ is a prefix of $h_2(\alpha)$, $\alpha \in \Sigma_A^+$, then α begins with a_0.*

Proof. Since $h_1(\alpha)$ is a prefix of $h_2(\alpha)$, we have $h_2(\alpha) = h_1(\alpha)w$ for some $w \in \Sigma_B^*$. We prove Claim 2 by elimination of all other possibilities (see also Table 1).

Case 1: α does not begin with a_1, a_2, or $x \in K \times V$. In this case, $h_2(\alpha)$ and $h_1(\alpha)w$ begin with different symbols.

Case 2: α cannot begin with $x \in H$; $h_1((q, A, v, q')) = (q, A)q'$. Since in this case $h_1(\alpha)w$ begins with $(q, A) \in K \times V$, the string $h_2(\alpha)$ must also begin with $(q, A) \in K \times V$, which is a contradiction since $(q, A) \in K \times V$ must be preceded by $q \in K$ in $h_2(\alpha)$ (see Table 1).

Case 3: Similarly, α cannot begin with a_3: $h_1(a_3) = b_1$, so $h_2(\alpha)$ must also begin with b_1, a contradiction with Table 1.

This proves Claim 2. □

Claim 3. *If $\alpha)\backslash h_2(\alpha) = w$, then there is no loss of generality in assuming that the only occurrence of a_0 in α is at the beginning, i.e., $\alpha = a_0\alpha'$, α' does not contain any a_0.*

Proof. Given an $\bar{\alpha}$ such that $h_2(\bar{\alpha}) = h_1(\bar{\alpha})w$, one can effectively find the shortest $\alpha \in \Sigma_A^+$ satisfying $h_2(\alpha) = h_1(\alpha)w$ (simply by testing all $\alpha \in \Sigma_A^+$ such that $|\alpha| \leq |\bar{\alpha}|$).

Suppose that α contains the symbol a_0 at least twice, that is, $\alpha = a_0\beta a_0\gamma$. (By Claim 2, the first a_0 must occur at the beginning.) Suppose the two leftmost a_0's are displayed. Thus β does not contain a_0 and hence neither of $h_1(\beta)$, $h_2(\beta)$ contains the symbol b_0:

$$\begin{array}{ll}
h_1(\alpha): & b_0|\underline{\quad h_1(\beta) \quad}|b_0|\underline{\quad h_1(\gamma) \quad}| \\
h_2(\alpha): & b_0b_1S|\underline{\quad h_2(\beta) \quad}|b_0b_1S|\underline{\quad h_2(\gamma) \quad}| \\
 & \uparrow \qquad\qquad\qquad\quad \uparrow
\end{array}$$

The two leftmost occurrences of b_0 in both $h_1(\alpha)$ and $h_2(\alpha)$

Then, also, $h_2(a_0\gamma) = h_1(a_0\gamma)w$, which is a contradiction since α is the shortest string with this property. □

Claim 4. *If $h_1(\alpha)\backslash h_2(\alpha) = w$ for some $w \in \Sigma_L^*$, then α is terminated by a_3.*

Proof. Clearly,

$$h_2(\alpha) = h_1(\alpha)w. \tag{2.5}$$

By the same reasoning as in Claim 2, we are going to prove Claim 4 by elimination of all other possibilities:

Case 1: α is not terminated by a_0: Let $\alpha = \alpha' a_0$. Thus, by Claim 2 and 3, $\alpha' = \varepsilon$ since the only a_0 in α is at the beginning. Then $\alpha = a_0$ and $b_0b_1S = b_0w$. Then $w = b_1S \notin \Sigma_L^*$, a contradiction.

Case 2: α is not terminated by a_1: By substitution into (2.5), we obtain $\ldots b_2 = \ldots b_1w$. Now, there are two possibilities: Either $w \neq \varepsilon$; then the right-hand side is terminated by some $s \in \Sigma_L$, or $w = \varepsilon$ and then it is terminated by b_1. In either case, it terminates with a symbol different from b_2, which is a contradiction.

Case 3: The proof for α terminated by a_2 is similar: Using (2.5), we get $\ldots q_F b_1 = \ldots b_2 q_1 w$.

Case 4: α is not terminated by $(q, A) \in K \times V$: $\ldots q(q, A) = \ldots Aw$. Since $w \in \Sigma_L^*$ and $A \in \Sigma_L$, the left- and right-hand sides end by different symbols, a contradiction.

Case 5: α cannot be terminated by $(q, A, v, q') \in H$: By substitution into (2.5), we have $\ldots v = \ldots (q, A)q'w$. Thus, either $w = \varepsilon$ and then the rightmost symbol is $q' \in K$, or $w \neq \varepsilon$ and then the right-hand side contains a substring $q's$ for some $s \in V$. This gives a contradiction in either case since $q' \in K$ must always be followed by $(q', B) \in K \times V$, or by b_1. (For the left-hand side of the equation is $h_2(\alpha)$.)

This proves Claim 4. □

Claim 5. *If $h_1(\alpha)\backslash h_2(\alpha) = w$ for some $w \in \Sigma_L^*$, then α cannot contain a_3 more than once.*

Proof. Define $N_b(w) = N_{b_1}(w) + N_{b_2}(w) + N_{b_0}(w)$. ($N_b(w)$ denotes the number of b's in w.) Now, let us count $N_b(w)$:

$$\begin{aligned}
N_b(w) &= N_b(h_1(\alpha)\backslash h_2(\alpha)) = N_b(h_2(\alpha)) - N_b(h_1(\alpha)) \\
&= N_{a_0}(\alpha) - N_{a_3}(\alpha)
\end{aligned}$$

since $N_b(h_2(x)) - N_b(h_1(x)) = 0$ for each x different from a_0, a_3, and $N_b(h_2(a_0)) - N_b(h_1(a_0)) = 1$ and $N_b(h_2(a_3)) - N_b(h_1(a_3)) = -1$. By Claims 2 and 3, α contains exactly one a_0; therefore,

$$N_b(w) = 1 - N_{a_3}(\alpha).$$

Since the number of b's in w must be a nonnegative integer, we have

$$N_{a_3}(\alpha) \leq 1.$$

In addition, we have also proved that if α does not contain any a_3, then $w = h_1(\alpha)\backslash h_2(\alpha) \notin V^*$ since w contains exactly one b-symbol. □

Now, let us present a brief summary of the claims:
- α begins with a_0 and ends by a_3.
- No more a_0's, a_3's are contained in α.
- If α does not contain a_3, then $h_1(\alpha)\backslash h_2(\alpha) \notin \Sigma_L^*$.

Let us now consider the form of $\alpha \in \Sigma_A^+$, such that $h_1(\alpha)$ is a prefix of $h_2(\alpha)$ and $h_1(\alpha)\backslash h_2(\alpha) \in \Sigma_L^*$, in a more detailed way. We shall do so by considering the form of all possible prefixes of such α and determining all possible ways of extending these prefixes to obtain again a prefix of α. In doing so we shall show that there exists a prefix φ satisfying

$$\begin{aligned} &h_1(\varphi)\backslash h_2(\varphi) = b_1 s_1 \ldots s_n \quad \text{and} \\ &S \Rightarrow_G^* s_1 \ldots s_n \end{aligned} \tag{2.6}$$

for some $s_1 \ldots s_n \in V$, $n \geq 0$. Then we shall show that having any prefix φ satisfying (2.6) (for some $s_1 \ldots s_n \in V^*$ derivable from S), it can only be extended so that another prefix φ' satisfying (2.6) is obtained (for some $s'_1 \ldots s'_{n'} \in V^*$ derivable from S).

(A) We know that α must begin with a_0, so there is a prefix satisfying (2.6), namely $\varphi = a_0$, since

$$\begin{aligned} h_1(\varphi)&: \quad b_0, \\ h_2(\varphi)&: \quad b_0 b_1 S. \end{aligned}$$

(Clearly, $h_1(\varphi)\backslash h_2(\varphi) = b_1 S$, and $S \Rightarrow_G^* S$.)

(B) Let now φ be an arbitrary prefix of α satisfying condition (2.6). Then

$$\begin{aligned} h_1(\varphi)&: \quad y, \\ h_2(\varphi)&: \quad y b_1 s_1 \ldots s_n \end{aligned}$$

(for some $y \in \Sigma_B^*$). There are two possible ways of extending φ such that $h_1(\varphi)$ will remain a prefix of $h_2(\varphi)$:

$$\varphi_1 = \varphi a_1 \quad \text{or} \quad \varphi_1 = \varphi a_3$$

because only $h_1(a_1)$ and $h_1(a_3)$ begin with symbol b_1.

(B1) Let us try to append a_3 first, i.e., $\varphi_1 = \varphi a_3$. Then

$$\begin{aligned} h_1(\varphi_1)&: \quad y b_1, \\ h_2(\varphi_1)&: \quad y b_1 s_1 \ldots s_n. \end{aligned}$$

In this case, we have $h_1(\varphi_1)\backslash h_2(\varphi_1)=s_1\ldots s_n\in V^*$. No further extension of φ_1 is possible by Claims 4 and 5. a_3 can be used only as the rightmost terminating character, which implies that $\alpha=\varphi_1$.

But, using (2.6), we have also $S\Rightarrow_G^* s_1\ldots s_n$, and we are done.

(B2) Another possibility is to append a_1, that is, $\varphi_1=\varphi a_1$:

$$\begin{aligned} h_1(\varphi_1)&: \quad yb_1,\\ h_2(\varphi_1)&: \quad yb_1s_1\ldots s_nb_2. \end{aligned}$$

There are now two cases.

(B2.1) If $n=0$, then we have

$$\begin{aligned} h_1(\varphi_1)&: \quad yb_1,\\ h_2(\varphi_1)&: \quad yb_1b_2. \end{aligned}$$

Now we are forced to append a_2, i.e., $\varphi_2=\varphi_1a_2$:

$$\begin{aligned} h_1(\varphi_2)&: \quad yb_1b_2q_1,\\ h_2(\varphi_2)&: \quad yb_1b_2q_Fb_1. \end{aligned}$$

But $h_1(\alpha)$ will remain a prefix of $h_2(\alpha)$ only if $q_1=q_F$, which contradicts one of the main initial assumptions of the theorem. Thus, this case has led up to a dead end, and we cannot extend φ by a_1 if $n=0$.

(B2.2) Let $n>0$. Since for each $x\in\Sigma_A$ (except $x\in K\times V$) we have $h_1(x)\notin V^*$, the only possible extension is $\varphi_2=\varphi_1(q_1,s_1)\ldots(q_n,s_n)$ for some $q_1\ldots q_n\in K$. Then,

$$\begin{aligned} h_1(\varphi_2)&: \quad yb_1s_1\ldots s_n,\\ h_2(\varphi_2)&: \quad yb_1s_1\ldots s_nb_2q_1(q_1,s_1)\ldots q_n(q_n,s_n). \end{aligned}$$

a_2 is the only symbol for which $h_1(x)$ begins with b_2; $h_1(a_2)=b_2q_1$. Clearly, further extension of φ_2 is possible only if

$$q_1=q_1. \tag{2.7}$$

For $\varphi_3=\varphi_2a_2$ we obtain

$$\begin{aligned} h_1(\varphi_3)&: \quad yb_1s_1\ldots s_nb_2q_1,\\ h_2(\varphi_3)&: \quad yb_1s_1\ldots s_nb_2q_1(q_1,s_1)q_2\ldots(q_n,s_n)q_Ft \end{aligned}$$

Because $h_1(\varphi_3)\backslash h_2(\varphi_3)\in(K\times V.K)^+b_1$, we must extend φ_3 by symbols from H. (If $x\notin H$, then $h_1(x)$ contains some symbols which are neither in $K\times V$, nor in K.)

So let

$$\varphi_4=\varphi_3(q_1,s_1,w_1,q_1')\ldots(q_n,s_n,w_n,q_n')$$

for some $w_i\in V^*$, $q_i'\in K$, such that $(q_i,s_i,w_i,q_i')\in H$. This gives

$$\begin{aligned} h_1(\varphi_4)&: \quad \ldots b_2q_1(q_1,s_1)q_1'\ldots(q_i,s_i)q_i'\ldots(q_n,s_n)q_n',\\ h_2(\varphi_4)&: \quad \ldots b_2q_1(q_1,s_1)q_2\ldots(q_i,s_i)q_{i+1}\ldots(q_n,s_n)q_Fb_1w_1\ldots w_i\ldots w_n. \end{aligned}$$

But $h_1(\varphi_4)$ will remain a prefix of $h_2(\varphi_4)$ only if we are able to pick and choose $(q_i, s_i, w_i, q_i') \in H$ such that

$$q_i' = q_{i+1} \quad \text{for } i = 1, \ldots, n-1, \tag{2.8}$$

$$q_n' = q_F. \tag{2.9}$$

Only under these conditions we have $h_1(\varphi_4)\backslash h_2(\varphi_4)$ defined. Then

$$h_1(\varphi_4)\backslash h_2(\varphi_4) = b_1 w_1 \ldots w_n = b_1 s_1' \ldots s_{n'}'$$

for some $s_1' \ldots s_{n'}' \in V$, $n' \geq 0$.

Now we need to show that $S \Rightarrow_G^* s_1' \ldots s_{n'}'$, and then $\varphi' = \varphi_4$ will also satisfy (2.6). But we have

$$s_1' \ldots s_{n'}' = w_1 \ldots w_n,$$

for some $(q_1, s_1, w_1, q_1') \ldots (q_n, s_n, w_n, q_n') \in H$. Moreover, by (2.7), (2.8), and (2.9), we have obtained

$$q_1 = q_1,$$

$$q_i' = q_{i+1} \qquad \text{for } i = 1, \ldots, n-1,$$

$$q_n' = q_F.$$

It is, as a matter of fact, just a slightly different formulation of the notion of the derivation step in g-systems. Thus,

$$s_1 \ldots s_n \Rightarrow_G s_1' \ldots s_{n'}'.$$

Recall that φ satisfies (2.6), hence $S \Rightarrow_G^* s_1 \ldots s_n$. Combining these results gives $S \Rightarrow_G^* s_1' \ldots s_{n'}'$, hence it follows that $\varphi' = \varphi_4$ again satisfies (2.6). (And we can continue our reasoning again from (B).)

Here we give a brief summary of (A) and (B).

(A) α must begin with a_0. (Otherwise, $h_1(\alpha)$ will not be a prefix of $h_2(\alpha)$.) For $\varphi = a_0$ condition (2.6) holds, i.e.,

$$\begin{aligned} &h_1(\varphi)\backslash h_2(\varphi) = b_1 s_1 \ldots s_n \quad \text{and} \\ &S \Rightarrow_G^* s_1 \ldots s_n \end{aligned} \tag{2.6}$$

(for some $s_1 \ldots s_n \in V^*$, $n \geq 0$).

(B) There are only two possibilities to extend φ satisfying (2.6):

(B1) $\varphi' = \varphi a_3$; then $h_1(\varphi')\backslash h_2(\varphi') = s_1 \ldots s_n$, and $S \Rightarrow_G^* s_1 \ldots s_n$. No further extension is possible in this case, which implies that $\varphi' = \alpha$.

(B2) $\varphi' = \varphi\beta$, and φ' again satisfies (2.6) for some string $s_1' \ldots s_{n'}' \in V^*$ derivable from S. (β must be of the form $a_1(K \times V)^+ a_2 H^+$.)

Moreover, using Claim 5, $h_1(\alpha)\backslash h_2(\alpha) \in V^*$ only if α is terminated by a_3.

Thus, if $h_1(\alpha)\backslash h_2(\alpha) = w \in \Sigma_L^* \subseteq V^*$, then $S \Rightarrow_G^* w$. Since $w \in \Sigma_L^*$, we have $w \in L(G)$, which proves Theorem 2.2. $\square$

If we used Σ_A^* instead of Σ_A^+, then the pair of homomorphisms would characterize a language $L \cup \{\varepsilon\}$ since the empty word can always be expressed in the form $\varepsilon = h_1(\varepsilon)\backslash h_2(\varepsilon)$. (No matter how h_1, h_2 are defined.)

3. Complexity of the representation

The construction that we have given allows us to draw some further conclusions. We have shown not only a pair of homomorphisms such that $w \in \Sigma_L^*$ is a word in L if and only if $w = h_1(\alpha)\backslash h_2(\alpha)$ for some $\alpha \in \Sigma_A^+$, but also, by (2.4), that α is of a special form: Let

$$S = w_0 \Rightarrow_G w_1 \Rightarrow_G \cdots \Rightarrow_G w_m \Rightarrow_G w$$

be a derivation of w in g-system G, then

$$\alpha \in a_0\left(\prod_{i=1}^{m} (a_1(K \times V)^{n_i} a_2 H^{n_i}) \right) a_3,$$

where $n_i = |w_i|$ (for $i = 0, \ldots, m$). Now we shall introduce a measure of complexity for this homomorphic representation (which corresponds to measuring the time and/or space complexity of the known types of accepting and/or generating devices). The efficiency of representation will be characterized by the length of α for which $h_1(\alpha)\backslash h_2(\alpha) = w$.

Definition. For each pair of homomorphisms representing a recursively enumerable language L we define

$$\mathrm{TS}_{h_1 h_2}(w) = \min_{\alpha \in \Sigma_A^+} \{|\alpha|; h_1(\alpha)\backslash h_2(\alpha) = w\}.$$

By Theorem 2.2, $\mathrm{TS}_{h_1 h_2}(w)$ is defined for each $w \in L$. Next, we define

$$\mathrm{TS}_{h_1 h_2}(n) = \max\{\mathrm{TS}_{h_1 h_2}(w); w \in L, |w| \leq n\}.$$

A pair of homomorphisms h_1, h_2 is said to be of complexity $\mathrm{TS}_{h_1 h_2}(n)$, if, for each word $w \in L$ of length n, there exists an $\alpha \in \Sigma_A^+$ of length at most $\mathrm{TS}_{h_1 h_2}(n)$ which satisfies $w = h_1(\alpha)\backslash h_2(\alpha)$.

We have the following relationship between the time and space complexity of the g-systems and the complexity of the homomorphic representation: If $w \in L(G)$ is generated by the derivation

$$S = w_0 \Rightarrow_G w_1 \Rightarrow_G \cdots \Rightarrow_G w_m \Rightarrow_G w,$$

then one can find an $\alpha \in \Sigma_A^+$ satisfying $h_1(\alpha)\backslash h_2(\alpha) = w$ such that

$$|\alpha| \leq 2 + \sum_{i=0}^{m} (2 + 2|w_i|) \leq 2 + 4 \sum_{i=0}^{m} |w_i|$$

(using (2.4)). Thus, the complexity measure TS corresponds to $\sum |w_i|$ of the derivation in the g-system. Our next development shows that constant factors do not matter in computing TS (the analogous result for Turing machines is known as "the speed-up theorem").

Theorem 3.1. *Let h_1, h_2 be a pair of homomorphisms representing a language L with complexity $\mathrm{TS}_{h_1h_2}(n)$. Then, for each $k>0$, there is a pair of homomorphisms h'_1, h'_2 representing the same language with complexity*

$$\mathrm{TS}_{h'_1h'_2}(n) \leq \lceil \mathrm{TS}_{h_1h_2}(n)/k \rceil.$$

Proof. Let h_1, h_2 be homomorphisms from Σ_A^* to Σ_B^*, and $\Sigma_L \subseteq \Sigma_B$. Define

$$\Sigma'_B = \Sigma_B, \qquad \Sigma'_A = \bigcup_{i=1}^{k} \Sigma_A^i.$$

Since strings in Σ'^*_A are composed of characters, which can be also viewed as strings in Σ_A^* (of length at most k), we shall use the following notation: "$\underline{abc}\ \underline{de}\ \underline{f}$" is a string in Σ'^*_A consisting of three symbols; "$\underline{abc}$", "$\underline{de}$", and "$\underline{f}$". The corresponding strings in Σ_A^* will be denoted by "abc", "de", and "f" respectively.

Now, define an auxiliary homomorphism g from Σ'^*_A to Σ_A^*:

$$g(\underline{x_1 \ldots x_j}) = x_1 \ldots x_j \quad \text{for each } j = 0, \ldots, k, \text{ and } x_1, \ldots, x_j \in \Sigma_A.$$

Finally, we define h'_1, h'_2:

$$h'_i(\alpha') = h_i(g(\alpha')) \qquad \text{for } i = 1, 2 \text{ and each } \alpha' \in \Sigma'^*_A.$$

Let $w = h'_1(\alpha') \backslash h'_2(\alpha')$, for some $\alpha' \in \Sigma'^+_A$. Then

$$w = h'_1(\alpha') \backslash h'_2(\alpha') = h_1(g(\alpha')) \backslash h_2(g(\alpha')),$$

so we were able to find $\alpha = g(\alpha') \in \Sigma_A^+$ such that $w = h_1(\alpha) \backslash h_2(\alpha)$.

Conversely, let $w = h_1(\alpha) \backslash h_2(\alpha)$ for some $\alpha \in \Sigma_A^+$, $\alpha = x_1 \ldots x_t$. Define

$$\alpha' = \underline{x_1 \ldots x_k}\,\underline{x_{k+1} \ldots x_{2k}} \ldots \underline{x_{\lfloor t/k \rfloor k-k+1} \ldots x_{\lfloor t/k \rfloor k}}\,\underline{x_{\lfloor t/k \rfloor k+1} \ldots x_t}.$$

Clearly, $|\alpha'| = \lceil |\alpha|/k \rceil$, and also $g(\alpha') = \alpha$. Now

$$w = h_1(\alpha) \backslash h_2(\alpha) = h_1(g(\alpha')) \backslash h_2(g(\alpha')) = h'_1(\alpha') \backslash h'_2(\alpha'),$$

and we are done. □

Example. We shall show that each regular set can be represented by a pair of homomorphisms with linear complexity. The idea of the proof is to construct a g-system G generating a regular set L "very fast", i.e., there exists a $c>0$ such that, for each $w \in L$, we have

$$S = w_0 \Rightarrow_G w_1 \Rightarrow_G \cdots \Rightarrow_G w_m \Rightarrow_G w \quad \text{and}$$

$$\sum_{i=0}^{m} |w_i| \leq c|w|.$$

The derivation will be of the form

$$S \Rightarrow AXB \Rightarrow AXXB \Rightarrow \cdots \Rightarrow AX^{2^i}B \Rightarrow AX^{2^{i+1}}B \Rightarrow \cdots$$
$$\Rightarrow AX^{2^{\lfloor \log n \rfloor}}B \Rightarrow AX^nB \Rightarrow w$$

(where $n = |w|$, and S, A, B, X are nonterminals).

We shall now design a g-system for L. Let L be given by a finite-state automaton $M = (K, \Sigma_L, \delta, q_\mathrm{I}, F)$, where K is a finite set of states, Σ_L an alphabet, δ a transition function, q_I in K an initial state, and $F \subseteq K$ a subset of final states. Then we define $G = (\{S, A, B, X\}, \Sigma_L, P, S)$, where P (1-a-transducer for rewriting relation) is given by $P = (K \cup \{q'_\mathrm{I}, q'_\mathrm{F}, q'\}, V, V, H, q'_\mathrm{I}, q'_\mathrm{F})$. q'_I, q'_F, and q' are some new states, and H (the set of transitions) will be defined as follows:

(i) The first step of a derivation $S \Rightarrow AXB$ will be done by $(q'_\mathrm{I}, S, AXB, q'_\mathrm{F}) \in H$.

(ii) Rewriting $AX^iB \Rightarrow AX^jB$ for $i \leqslant j \leqslant 2i$ will be performed by edges $(q'_\mathrm{I}, A, A, q')$, (q', X, XX, q'), (q', X, X, q'), and $(q', B, B, q'_\mathrm{F}) \in H$.

(iii) For the last step of a derivation $AX^nB \Rightarrow w$, the following transitions are needed:

$$(q'_\mathrm{I}, A, \varepsilon, q_\mathrm{I}) \in H,$$

$$(q_1, X, a, q_2) \in H \text{ iff } \delta(q_1, a) = q_2 \quad \text{for each } q_1, q_2 \in K,\ a \in \Sigma_L,$$

$$(q, B, \varepsilon, q'_\mathrm{F}) \in H \quad \text{for each } q \in F.$$

The following will hold for the most efficient derivation of w in G:

$$\sum_{i=0}^{m} |w_i| = 1 + \sum_{i=0}^{\lfloor \log n \rfloor} (2 + 2^i) + (2 + n)$$
$$\leqslant 3n + 2\log n + 4 \leqslant 9n.$$

Thus, by Theorems 2.2 and 3.1, for each regular set L and arbitrarily large $k > 0$, we can construct a pair of homomorphisms h_1, h_2 such that $w \in \Sigma_L^*$ is a word in L if and only if there exists an α of length at most $\lceil |w|/k \rceil$ satisfying $w = h_1(\alpha) \backslash h_2(\alpha)$.

4. Some extensions

To characterize languages by homomorphisms we can use the right quotient as well, as expressed in the following theorem.

Theorem 4.1. *For each recursively enumerable language L there exists a pair of homomorphisms $h_1, h_2 : \Sigma_A^* \to \Sigma_B^*$ such that*

$$L = \{w \in \Sigma_L^*;\ w = h_2(\alpha)/h_1(\alpha) \text{ for some } \alpha \in \Sigma_A^+\} = O(h_2/h_1) \cap \Sigma_L^*.$$

Table 2

$x \in \Sigma_A$	$h_1(x)$	$h_2(x)$	Remark
a_0	b_0	Sb_1b_0	
a_1	b_1	b_2	
a_2	q_Fb_2	b_1q_1	
a_3	b_1	ε	
(q, A)	A	$(A, q)q$	for each $x \in K \times V$
(q, A, v, q')	$q(A, q')$	v	for each $x \in H$

(Similarly as in Theorem 2.2, Σ_A and Σ_B are some alphabets, $\Sigma_L \subseteq \Sigma_B$.) The proofs of Theorems 2.2 and 4.1 are so similar that we will present only a modification of Table 1 (definition of h_1, h_2) (cf. Table 2). All the rest of the proof is merely a "mirror image" of the proof of Theorem 2.2.

A more important modification of Theorem 2.2 concerns the family of context-sensitive languages.

Theorem 4.2. *For each context-sensitive language L there exists a pair of homomorphisms $h_1, h_2: \Sigma_A^* \to \Sigma_B^*$ such that*

$$L = \{w \in \Sigma_L^*;\ w = h_1(\alpha)\backslash h_2(\alpha) \text{ for some } \alpha \in \Sigma_A^+\} = O(h_1\backslash h_2) \cap \Sigma_L^*$$

and

$$|h_1(x)| \leq |h_2(x)| \quad \text{for each } x \in \Sigma_A.$$

(The last condition plays the same role as monotony for the phrase-structure grammars.)

The proof is based on the fact (Rovan [8]) that the context-sensitive languages correspond to so-called ε-free, nonerasing g-systems (i.e., for each $(q, s, v, q') \in H$ we have $v \neq \varepsilon$). Some additional packing together of symbols in Σ_A and Σ_B is needed (by the method similar to that in Theorem 3.1).

The converse is also true since one can easily construct a nondeterministic linear-bounded Turing machine to a given "monotonic" pair of homomorphisms.

We can make one step further in this analogy and consider pairs of homomorphisms such that $|h_1(x)| = 1$ for each x. However, here is a difference with the Chomsky hierarchy since this class of pairs of homomorphisms is capable of representing an arbitrary PE0L language. (See [9] for the definition of PE0L.) The exact identification of the corresponding language family is an open problem.

References

[1] K. Culik II, A purely homomorphic representation of recursively enumerable sets, *J. ACM* **26** (1979) 345-350.

[2] K. Culik II and N.D. Diamond, A homomorphic characterization of time and space complexity classes of languages, *Internat. J. Comput. Math.* **8** (1980) 207-222.
[3] J. Engelfriet and G. Rozenberg, Fixed point languages, equality languages and representation of recursively enumerable languages, *J. ACM* **27** (1980) 499-518.
[4] V. Geffert, Grammars with context dependency restricted to synchronization, in: *Proc. MFCS'86*, Lecture Notes in Computer Science **233** (Springer, Berlin, 1986) 370-378.
[5] S. Ginsburg, *Algebraic and Automata-theoretic Properties of Formal Languages* (North-Holland, Amsterdam, 1975).
[6] M. Harrison, *Introduction to Formal Language Theory* (Addison-Wesley, Reading, MA, 1978).
[7] J.E. Hopcroft and J.D. Ullman, *Formal Languages and their Relation to Automata* (Addison-Wesley, Reading, MA, 1969).
[8] B. Rovan, A framework for study grammars in: *Proc. MFCS'81*, Lecture Notes in Computer Science **118** (Springer, Berlin, 1981) 473-482.
[9] G. Rozenberg and A. Salomaa, *The Mathematical Theory of L Systems* (Academic Press, New York, 1980).
[10] A. Salomaa, Equality sets of homomorphisms of free monoids, *Acta Cybernet.* **4** (1978) 127-139.
[11] A. Salomaa, *Formal Languages* (Academic Press, New York, 1973).