A MORPHIC REPRESENTATION OF COMPLEMENTS OF RECURSIVELY ENUMERABLE SETS

by

A. Ehrenfeucht
G. Rozenberg
K. Ruohonen

CU-CS-193-80

A. Ehrenfeucht
Dept. of Computer Science
University of Colorado at Boulder
Boulder, Colorado 80309

G. Rozenberg
Institute of Applied Mathematics
and Computer Science
University of Leiden
2300 RA Leiden
The Netherlands

K. Ruohonen
Institute of Mathematics
Tampere University of Technology
SF-33101, Tampere 10, Finland

All Correspondence to G. Rozenberg

Abstract

After extending two word morphisms f and g to languages, an equation $f(X) = g(X)$ can be written and its language solutions investigated. An elementary characterization of the family of all solutions of the equation is given and it is used to investigate the maximal solution which is the main subject of this paper. It turns out that going through all propagating morphisms f and g the family of maximal solutions obtained equals the family of complements of recursively enumerable languages after intersecting with regular languages and mapping with propagating morphisms. In the general case (of arbitrary morphisms f and g) the corresponding family is larger and includes the full-AFL closure of the family of complements of recursively enumerable languages.

Key words

complements of recursively enumerable languages, language equations

CR categories

5.23, 5.26, 5.27

0. Introduction

Any word morphism $A^* \to B^*$ determines in the natural way a morphism $2^{A^*} \to 2^{B^*}$ (not all morphisms $2^{A^*} \to 2^{B^*}$ are obtained in this way, of course). Therefore, given two word morphisms $f,g : A^* \to B^*$, we may write the equation

$$f(X) = g(X) \tag{1}$$

the solutions of which are languages over A.

We give first some elementary algebraic properties of the family of solutions of equations like (1) and then use them to investigate the main subject of this paper, viz. maximal elements in these families. Our principal result is the following fact: Starting with maximal elements of families of solutions of equations like (1) with f and g propagating and applying first intersections with regular languages and then propagating morphisms one gets exactly the family of complements of recursively enumerable languages. This is mainly due to (I) the fact that the maximal solution of (1) can be obtained by iteration if f and g are propagating, i.e. it equals $\bigcap_{n=-\infty}^{\infty} (g^{-1}f)^n(A^*)$, and (II) the remarkable ability of morphisms to simulate intermediate steps in all kinds of effective processes.

Classically (II) is justified by various proofs of the undecidability of Post's Correspondence Problem, cf. e.g. [S1], and more recently by results concerning equality sets of morphisms, i.e. languages of the form $\{P \mid f(P) = g(P)\}$; see [S2], [ČS], [ER1] and [ER2]. In fact, we have here a "complementary" situation: recursively enumerable languages can be characterized using equality sets and complements of recursively enumerable languages can be characterized using maximal solutions of equations like (1).

Thus both recursively enumerable languages and their complements can be given a morphic characterization!

The above mentioned principal result does not hold true for general (nonpropagating) morphisms f and g. In fact, the family of languages obtained in this general case includes the full-AFL closure of the family of complements of recursively enumerable languages. The strictness of this inclusion, though, remains an open problem.

It may be mentioned that a well-known connection exists between maximal fixed points and infinite computations, see e.g. [B], [dB], [dR], [M] and [T], which bears resemblance to our constructions. However, it is perhaps surprising to have such a simple and concrete connection as is obtained here.

1. Notation

— the empty word is denoted by Λ

— the complement of a language L is denoted by $\overline{L}$

— the maximal solution of the equation $f(X) = g(X)$ is denoted by maxeq(f,g)

— the family of solutions of the equation $f(X) = g(X)$ is denoted by $EQ(f,g)$; following standard category theoretic terminology it might be called the equalizer of f and g

— $MAX(HOM) = \{maxeq(f,g) \mid f$ and g are morphisms$\}$

— $MAX(PHOM) = \{maxeq(f,g) \mid f$ and g are propagating morphisms$\}$

— the closure of a family F of languages under

(a) intersections with regular languages is denoted by $F \wedge R$

(b) morphisms is denoted by $\hat{H}(F)$

(c) propagating morphisms is denoted by $H(F)$

— the set of integers is denoted by $\mathbb{Z}$ and the set of natural numbers is denoted by $\mathbb{N}$

— the families of recursively enumerable languages and their complements are denoted by RE and $\overline{RE}$, respectively

— the necessary requisite in formal language theory and theory of recursive functions can be found e.g. in [S1] or [HU] and [R].

Let f and g be morphisms. A doubly infinite sequence of words $(\omega_n)_{n=-\infty}^{\infty}$ is called an (f,g)-sequence if

$$f(\omega_n) = g(\omega_{n+1}) \quad \text{for} \quad n \in \mathbb{Z}$$

(see Fig. 1). A language L is called an (f,g)-language if L equals $\{\omega_n | n \in \mathbb{Z}\}$ for an (f,g)-sequence $(\omega_n)_{n=-\infty}^{\infty}$.

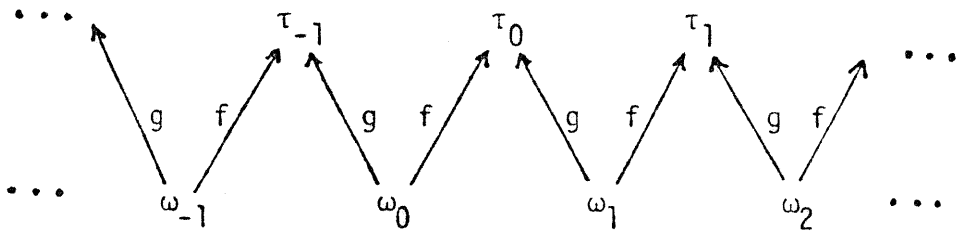

Fig. 1. It is frequently illustrative to depict an (f,g)-sequence in the above manner. Here $\tau_n = f(\omega_n)$.

We give some algebraic properties of $EQ(f,g)$.

THEOREM 2.1. Let f and g be morphisms. Then $EQ(f,g)$ is a complete semilattice with respect to union generated by the (f,g)-languages and with maximal element $\text{maxeq}(f,g)$. Especially, if f and g are monomorphisms, then $EQ(f,g)$ is a complete Boolean algebra with respect to set-theoretic operations.

Proof. Clearly $EQ(f,g)$ is closed with respect to all kinds of unions. To prove the first claim of the theorem we note first that each (f,g)-language belongs to $EQ(f,g)$. Let then $L \neq \Phi$ belong to $EQ(f,g)$ and $\omega_0 \in L$. We show that a certain (f,g)-language $\{\omega_n | n \in \mathbb{Z}\}$ is included in L . Since $\omega_0 \in L$, there are words $\omega_{-1}, \omega_1 \in L$ such that

$$f(\omega_{-1}) = g(\omega_0) \ , \quad f(\omega_0) = g(\omega_1) \ .$$

Since $\omega_{-1},\omega_1 \in L$, there are words $\omega_{-2},\omega_2 \in L$ such that

$$f(\omega_{-2}) = g(\omega_{-1}) \ , \quad f(\omega_1) = g(\omega_2) \ .$$

Continuing in this fashion we see that an (f,g)-language $L_1 = \{\omega_n | n \in \mathbb{Z}\}$ is included in L . If $L_1 = L$, then there is nothing more to prove. If $L_1 \subsetneq L$ and $\omega_0' \in L - L_1$, then, as above, it is seen that $\omega_0' \in L_2$ for some (f,g)-language $L_2 \subset L$. In this fashion we get a sequence of (f,g)-languages $L_1,L_2,L_3,\ldots$ such that $L = L_1 \cup L_2 \cup L_3 \cup \cdots$

The second claim of the theorem follows because the (f,g)-languages are pairwise disjoint if f and g are injective. □

THEOREM 2.2. Let f and g be morphisms, Then $EQ(f,g)$ is closed under catenation of languages.

Proof. If $f(X) = g(X)$ and $f(Y) = g(Y)$, then $f(XY) = f(X)f(Y) = g(X)g(Y) = g(XY)$. □

COROLLARY 2.1. Let f and g be morphisms, Then $maxeq(f,g)$ is a star language, i.e. a monoid. □

In the sequel we consider the families MAX(HOM) and MAX(PHOM) . The languages of these families are star languages by Corollary 2.1. There are, however, very simple star languages which are not in MAX(HOM) , e.g. the language $(a^2)^*$ is one. On the other hand, MAX(PHOM) contains languages which are not even recursively enumerable, as we shall see. Thus the position of MAX(HOM) and MAX(PHOM) is curiously transverse with respect to the Chomsky hierarchy of language families.

We conclude this section by some remarks concerning sequences closely connected with (f,g)-sequences. Let f and g be morphisms. We call a sequence of words $(\tau_n)_{n=-\infty}^{\infty}$ an <u>(f^{-1},g^{-1})-sequence</u> if

$$\tau_n \in gf^{-1}(\tau_{n+1}) \quad \text{for} \quad n \in \mathbb{Z} .$$

Thus, for every (f^{-1},g^{-1})-sequence $(\tau_n)_{n=-\infty}^{\infty}$, there exists an (f,g)-sequence $(\omega_n)_{n=-\infty}^{\infty}$ such that $\tau_n = f(\omega_n)$ for $n \in \mathbb{Z}$, and conversely, every sequence $(f(\omega_n))_{n=-\infty}^{\infty}$, where $(\omega_n)_{n=-\infty}^{\infty}$ is an (f,g)-sequence, is an (f^{-1},g^{-1})-sequence; see Fig. 1. The concept of an <u>(f^{-1},g^{-1})-language</u> is defined in an obvious way.

The union of all (f^{-1},g^{-1})-languages is denoted by $\mathrm{maxeq}(f^{-1},g^{-1})$. (Note that if $f,g : A^* \to B^*$ are monomorphisms, then $\mathrm{maxeq}(f^{-1},g^{-1})$ is the maximal solution of the equation $f^{-1}(X) = g^{-1}(X)$ in $2^{f(A^*)\cap g(A^*)}$.)

THEOREM 2.3. Let f and g be morphisms $A^* \to B^*$.

(i) If L is an (f,g)-language, then $f(L)$ is an (f^{-1},g^{-1})-language, and conversely, for any (f^{-1},g^{-1})-language L there exists an (f,g)-language L_1 such that $L = f(L_1)$. Consequently $f(\mathrm{maxeq}(f,g)) = \mathrm{maxeq}(f^{-1},g^{-1})$.

(ii) There exist alphabets A_1 and $B_1 \supset A$ and morphisms $f_1,g_1 : A_1^* \to B_1^*$ (which are propagating if f and g are so) such that each (f,g)-language equals $L \cap A^*$ for an (f_1^{-1},g_1^{-1})-language L. Furthermore, $\mathrm{maxeq}(f,g) = \mathrm{maxeq}(f_1^{-1},g_1^{-1}) \cap A^*$.

(iii) There exist alphabets $A_2 \supset B$ and B_2 and morphisms $f_2,g_2 : A_2^* \to B_2^*$ (which are propagating if f and g are so) such that each (f^{-1},g^{-1})-language equals $L \cap B^*$ for some (f_2,g_2)-language L. Furthermore, $\mathrm{maxeq}(f^{-1},g^{-1}) = \mathrm{maxeq}(f_2,g_2) \cap B^*$.

(iv) $\mathrm{maxeq}(f,g) = f^{-1}(\mathrm{maxeq}(f^{-1},g^{-1})) \cap g^{-1}(\mathrm{maxeq}(f^{-1},g^{-1}))$

Proof. (i) Obvious.

(ii) Define $A_1 = A \cup \{\bar{a} \mid a \in A\}$, $B_1 = A \cup \{\bar{b} \mid b \in B\}$ and

$$f_1(a) = a \quad , \quad f_1(\bar{a}) = \overline{f(a)} \ ,$$

$$g_1(a) = \overline{g(a)} \ , \quad g_1(\bar{a}) = a \quad \text{for} \quad a \in A$$

(where $\overline{P}$ stands for the word obtained from P by "barring" its symbols). Then $(\omega_n)_{n=-\infty}^{\infty}$ is an (f,g)-sequence if and only if

$$\ldots, \omega_{-2} \ , \ \overline{f(\omega_{-2})} \ , \ \omega_{-1} \ , \ \overline{f(\omega_{-1})} \ , \ \omega_0 \ , \ \overline{f(\omega_0)} \ , \ \omega_1 \ , \ \overline{f(\omega_1)} \ , \ldots$$

is an (f_1^{-1},g_1^{-1})-sequence. N.B. the fact that it suffices to consider (f_1^{-1},g_1^{-1})-sequences of this "alternating" type only and that erasing by f_1 and/or g_1 does not cause any troubles although it does have the effect of mixing "barred" and "unbarred" symbols.

(iii) Analogous to (ii).

(iv) The left hand side of the equation is clearly included in the right hand side. To prove the reverse inclusion let $(\omega_n)_{n=-\infty}^{\infty}$ and $(\tau_n)_{n=-\infty}^{\infty}$ be (f,g)-sequences and $\omega \in g^{-1}f(\omega_{n_1}) \cap f^{-1}g(\tau_{n_2})$, say. Then the "recombination" of $(\omega_n)_{n=-\infty}^{\infty}$ and $(\tau_n)_{n=-\infty}^{\infty}$

$$\ldots \ , \ \omega_{n_1-2} \ , \ \omega_{n_1-1} \ , \ \omega_{n_1} \ , \ \omega \ , \ \tau_{n_2} \ , \ \tau_{n_2+1} \ , \ \ldots$$

is an (f,g)-sequence, too. See Fig. 2. Thus $\omega \in \mathrm{maxeq}(f,g)$. □

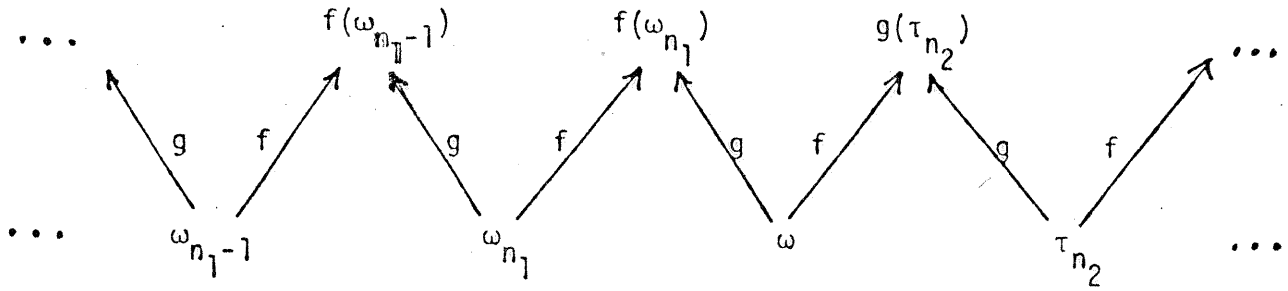

Fig. 2. "Recombination" of two (f,g)-sequences.

There is thus a close connection between maxeq(f,g) and maxeq(f^{-1},g^{-1}) (and the associated notions) via simple language operations. For this reason they are widely interchangable in the sequel. We shall, however, confine ourselves to maxeq(f,g) .

COROLLARY 2.2. Let f and g be morphisms A* → B* . Then maxeq(f^{-1},g^{-1}) is a star language. If one of the alphabets A and B is unary, then maxeq(f,g) and maxeq(f^{-1},g^{-1}) are both regular.

Proof. Corollary 2.1 and Theorem 2.3. (Unary star languages are well-known to be regular.) □

It is interesting to compare the latter results with the corresponding results for equality sets: A unary equality set is regular (being a star language); equality sets of morphisms with unary range alphabets are context-free but not all of them are regular; see [S2].

3. Complements of recursively enumerable languages

Although there probably is no simple characterization of MAX(PHOM) or MAX(HOM) using "traditional characterization mechanisms" in formal language theory, it turns out that $H(\text{MAX(PHOM)} \wedge R)$ has a surprisingly simple characterization: it equals $\overline{RE}$. This fact is proved in this section. We begin with

LEMMA 3.1. Let f and g be morphisms on A^*. Then the language $L = \cap_{n=-\infty}^{\infty} (f^{-1}g)^n(A^*)$ is in $\overline{RE}$.

Proof. The lemma follows because $\overline{L} = \cup_{n=-\infty}^{\infty} \overline{(f^{-1}g)^n(A^*)}$ is an effectively denumerable union of effectively obtainable regular languages and hence is recursively enumerable. □

THEOREM 3.1. $H(\text{MAX(PHOM)} \wedge R) = \overline{RE}$

Proof. (I) Let us prove first that, for any propagating morphisms f and g, $\text{maxeq}(f,g) \in \overline{RE}$. It suffices to prove (by Lemma 3.1) that

$$\text{maxeq}(f,g) = \cap_{n=-\infty}^{\infty} (f^{-1}g)^n(A^*) =_{\text{def}} L ,$$

where A is the domain alphabet of f and g, i.e. the maximal solution of $f(X) = g(X)$ can be obtained by iteration. Since each word of $\text{maxeq}(f,g)$ is a term of an (f,g)-sequence, it follows that $\text{maxeq}(f,g) \subset L$.

To prove the reverse inclusion let $P \in L$ be arbitrary. Now each of the languages $(f^{-1}g)^n(P)$, $n \in \mathbb{Z}$, is finite because f and g are propagating. An application of König's Lemma (on infinite graphs) then shows that P is a term of an (f,g)-sequence and is thus in $\text{maxeq}(f,g)$.

Since $\overline{RE}$ is an AFL (see [U], Theorem 2.2 and its proof), it follows that $H(\text{MAX(PHOM)} \wedge R) \subset \overline{RE}$.

(II) To show that $\overline{RE} \subset H(MAX(PHOM) \wedge R)$ we simulate computations of deterministic Turing machines using (f,g)-sequences. Consider the following variant of 1-tape deterministic Turing machine. Let A be a finite alphabet and denote

$$A_{left} = \{a_{left} | a \in A\} \ , \quad A_{right} = \{a_{right} | a \in A\} \ .$$

The machine is an ordered quintuple $M = \langle B,S,X,\lambda,s_0 \rangle$ where

(i) $B = A \cup A_{left} \cup A_{right}$ is the input alphabet

(ii) S is the state alphabet and $s_0 \in S$ is the initial state

(iii) $X = Y \cup Y_{left} \cup Y_{right}$ where Y is the auxiliary tape alphabet and Y_{left} (leftmost symbols) and Y_{right} (rightmost symbols) are defined as A_{left} and A_{right} above; it is assumed that $X \cap B = \Phi$

(iv) $\lambda : (B \cup X) \times S \to (B \cup X \cup (A \cup Y)(A_{right} \cup Y_{right})) \times S \times$ $\times \{left,right,stay\}$ is the partial transition function satisfying the following conditions:

(a) $\lambda((A_{left} \cup Y_{left}) \times S) \subset (A_{left} \cup Y_{left}) \times S \times \{right,stay\}$

(b) $\lambda((A_{right} \cup Y_{right}) \times S) \subset ((A_{right} \cup Y_{right}) \times S \times \{left,stay\}) \cup$
$\cup ((A \cup Y)(A_{right} \cup Y_{right}) \times S \times \{stay\})$

(c) $\lambda((A \cup Y) \times S) \subset (A \cup Y) \times S \times \{left,right,stay\}$

The input $aa_1 \cdots a_n b$ ($a,a_1,\ldots,a_n,b \in A$ and $n \geq 0$) is given to the machine in the form $a_{left}a_1 \cdots a_n b_{right}$ and the scanning of this input starts from a_{left} in state s_0. When reading a symbol x in state s the next configuration is determined as follows:

(i) If $x \in A \cup A_{left} \cup Y \cup Y_{left}$ and $\lambda(x,s) = (y,s',z)$ then the machine writes y on the scanned tape cell, changes its state from s to

s' and moves its head one cell to the left or to the right or does not move its head, according to whether z is "left", "right" or "stay", respectively.

(ii) If $x \in A_{right} \cup Y_{right}$ and $\lambda(x,s) = (y,s',z)$, where $y \in A_{right} \cup Y_{right}$, then the machine writes y on the scanned tape cell, changes its state from s to s' and moves its head to the left or does not move its head, according to whether z is "left" or "stay", respectively.

(iii) If $x \in A_{right} \cup Y_{right}$ and $\lambda(x,s) = (yu,s',stay)$, where $y \in A \cup Y$ and $u \in A_{right} \cup Y_{right}$, then the machine writes y on the scanned tape cell, adds one cell to the right where it writes u, changes its state from s to s' and keeps its head on the originally scanned tape cell.

(iv) If $\lambda(x,s)$ is undefined, then the machine halts.

An input is accepted if and only if the machine halts in some finite number of steps after receiving it.

It should be clear that although our machines have some unconventional features they are equivalent to more standard Turing machines (e.g. in [HU]) and recognize exactly all recursively enumerable languages which do not contain the empty word or words of length 1. (We have not incorporated in our machines any additional mechanisms for accepting words of length less than 2 since such words can be dealt with separately, if necessary.)

We now define the new alphabets

$$C_1 = B \cup X \cup \{\bar{a} \mid a \in A\} \cup \{F\} \cup$$

$$\cup \{(x,s,y) \mid x \in A_{left} \cup Y_{left},\ y \in B \cup X,\ s \in S \text{ and } \lambda(x,s) \text{ is defined}\}$$

$$\cup \{(x,y,s) \mid x \in B \cup X,\ y \in A_{right} \cup Y_{right},\ s \in S \text{ and } \lambda(y,s) \text{ is defined}\}$$

$$\cup \{(x,y,s,u) \mid x,y,u \in B \cup X,\ s \in S \text{ and } \lambda(y,s) \text{ is defined}\},$$

$$C_2 = B \cup X \cup ((B \cup X) \times S) \cup \{F\}$$

and morphisms $f,g : C_1^* \to C_2^*$ as follows:

$$f(x) = g(x) = x \quad \text{if} \quad x \in B \cup X \cup \{F\} ,$$

$$f(\bar{a}) = (a_{left},s_0) \quad \text{and} \quad g(\bar{a}) = F \quad \text{if} \quad a \in A ,$$

$$g((x,s,y)) = (x,s)y \ , \ g((x,y,s)) = x(y,s) ,$$

$$g((x,y,s,u)) = x(y,s)u ,$$

$$f((x,s,y)) = \begin{cases} (x',s')y & \text{if} \quad \lambda(x,s) = (x',s',stay) \\ x'(y,s') & \text{if} \quad \lambda(x,s) = (x',s',right) , \end{cases}$$

$$f((x,y,s)) = \begin{cases} x(y',s') & \text{if} \quad \lambda(y,s) = (y',s',stay) \\ (x,s')y' & \text{if} \quad \lambda(y,s) = (y',s',left) \\ x(y',s')u & \text{if} \quad \lambda(y,s) = (y'u,s',stay) , \end{cases}$$

$$f((x,y,s,u)) = \begin{cases} (x,s')y'u & \text{if} \quad \lambda(y,s) = (y',s',left) \\ x(y',s')u & \text{if} \quad \lambda(y,s) = (y',s',stay) \\ xy'(u,s') & \text{if} \quad \lambda(y,s) = (y',s',right) . \end{cases}$$

By inspection one sees easily that the language not accepted by M is $h(maxeq(f,g) \cap \{\bar{a} | a \in A\}A^*A_{right})$ where $h(\bar{a}) = h(a_{right}) = h(a) = a$. See Fig. 3.

Let then $L \in \overline{RE}$ be a language over A . By the above $L - \{\Lambda\} - A = h(K \cap R)$ for a propagating morphism h , $K \in MAX(PHOM)$ and a regular language R . We may assume that the intersection of A and the alphabet A_1 of K and R is empty. Let $K = maxeq(f,g)$. We may assume further that the intersection of A and the range alphabet of f and g is empty. Extend then f , g and h onto $(A \cup A_1)^*$ by

$$f_1(a) = g_1(a) = h_1(a) = a \quad \text{if} \quad a \in A ,$$

$$f_1(a) = f(a) \ , \ g_1(a) = g(a) \ , \ h_1(a) = h(a) \quad \text{if} \quad a \in A_1 .$$

Then $L = h_1(K_1 \cap R_1)$ where $K_1 = \text{maxeq}(f_1, g_1)$ and $R_1 = R \cup ((A \cup \{\Lambda\}) \cap L)$. □

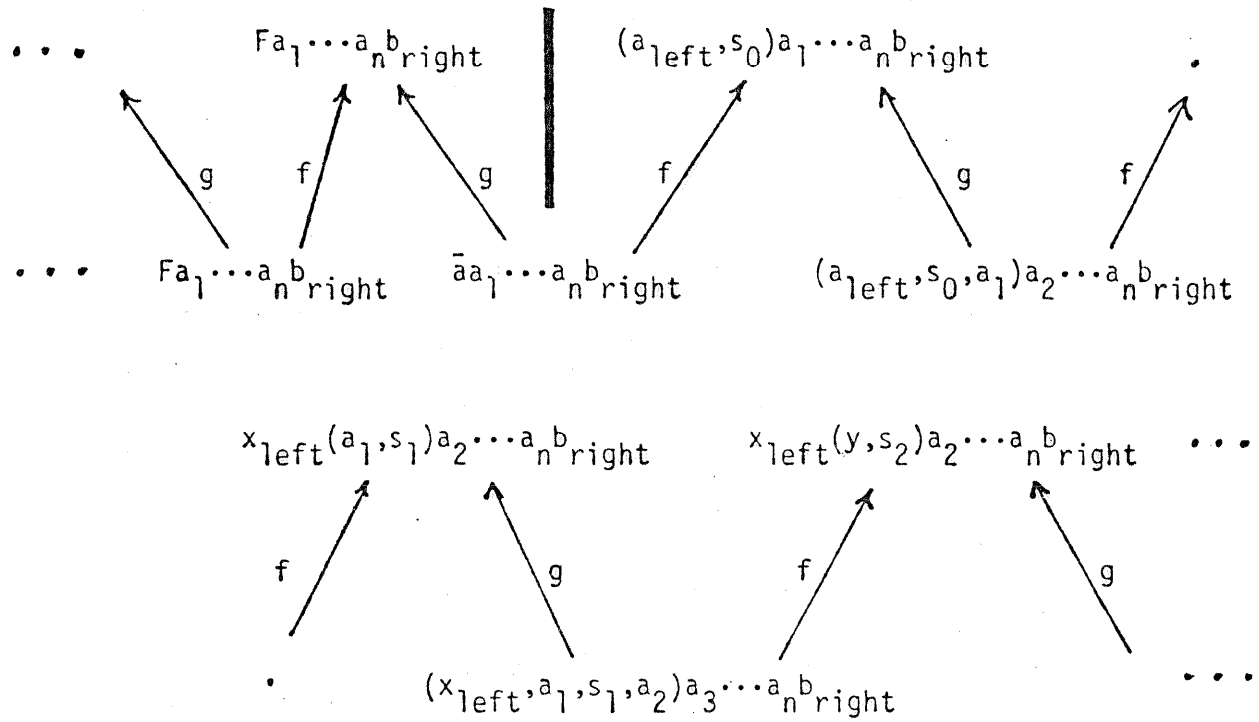

Fig. 3. Simulation of a computation of M using (f,g)-sequences. The lower sequence is an (f,g)-sequence whereas the upper (f^{-1}, g^{-1})-sequence (to the right of the vertical bar) can be considered as a sequence of consecutive configurations of M . A symbol x under scanning in state s is given as (x,s) . (It is assumed in the picture that $\lambda(a_{left}, s_0) = (x_{left}, s_1, right)$ and $\lambda(a_1, s_1) = (y, s_2, stay)$.)

4. The role of erasing

Since $H(\mathrm{MAX}(\mathrm{PHOM})\wedge R) = \overline{\mathrm{RE}}$, then, of course, $\hat{H}(\mathrm{MAX}(\mathrm{PHOM})\wedge R)$ equals $\hat{H}(\overline{\mathrm{RE}})$, i.e. the full-AFL closure of $\overline{\mathrm{RE}}$ (see e.g. Theorem 3.4.3(b) in [G]), and hence strictly includes $\mathrm{RE} \cup \overline{\mathrm{RE}}$.

We will show next that

$$\hat{H}(\overline{\mathrm{RE}}) = \hat{H}(\mathrm{MAX}(\mathrm{PHOM})\wedge R) \subset H(\mathrm{MAX}(\mathrm{HOM})\wedge R) .$$

Thus Theorem 3.1 does not hold true for nonpropagating morphisms any more. Consequently, for nonpropagating f and/or g, the maximal solution of $f(X) = g(X)$ cannot be obtained by iteration in the general case, see the proof of Theorem 3.1. We do not know whether the inclusion $\hat{H}(\overline{\mathrm{RE}}) \subset H(\mathrm{MAX}(\mathrm{HOM})\wedge R)$ is strict or not.

LEMMA 4.1. Let $L \in \overline{\mathrm{RE}}$ be a language over $A_1 \cup A_2$, where $A_1 \cap A_2 = \Phi$, and let e be defined by $e(a) = a$, for $a \in A_1$, and $e(a) = \Lambda$, for $a \in A_2$. Let c be a symbol of A_2 and $\$$ a symbol not in $A_1 \cup A_2$ and extend e onto $(A_1 \cup A_2 \cup \{\$\})^*$ by $e(\$) = \Lambda$. Then there exists a language $K \subset A_1^* c^+ \$$ such that $K \in \overline{\mathrm{RE}}$ and $e(L) = e(K)$.

Proof. An easy modification of a deterministic Turing machine M recognizing $\overline{L}$ produces a deterministic Turing machine M' recognizing $\overline{K}$, where K is obtained from L by moving all symbols of A_2 to the final block and at the same time changing them to c and finally adding $c\$$ to the right. M' first tests whether the input word P is in $A_1^* c^+ \$$ and halts in the negative case. In the positive case M' lists all "candidates" for a word of L which could be transformed to P by moving the symbols of A_2 to the final block and changing them to c and adding $c\$$; there will, of course, be a finite number of these candidates. After

that M' simulates deterministically computations of M on each of these candidates and halts if and only if all these simulations end in acceptance. Further details are straightforward to insert and are left to the reader. (Note that an idea of recognizing $\overline{e(L)}$ in a similar fashion fails – as it must, since $\overline{RE}$ is not closed under erasing – because the number of candidates is then infinite.) □

THEOREM 4.1. $\hat{H}(MAX(PHOM)\wedge R) \subset H(MAX(HOM)\wedge R)$

Proof. Let A_1, A_2, c, $\$$, e and K be as in the proof of Lemma 4.1. Since $K \in \overline{RE}$, the construction in the proof of Theorem 3.1 is applicable to K. Thus, for each word $aa_1\cdots a_m bc^{n+1}\$ \in K$, where $a,a_1,\ldots,a_m,b \in A_1$ and $m \geq 2$, the word $\bar{a}a_1\cdots a_m bc^{n+1}\$_{right}$ belongs to an (f,g)-sequence where f and g are the propagating morphisms given by the proof of Theorem 3.1; see Fig. 3. (We employ here the notation of the proof, now $A = A_1 \cup A_2 \cup \{\$\}$.) Define new alphabets

$$A_3 = \{\tilde{a} \mid a \in A_1\} \cup \{c_1,c_2,p,q\} \quad \text{and} \quad A_4 = \{x,y,p,q\},$$

which are disjoint from $C_1 \cup C_2$, and extend f and g onto $(C_1 \cup A_3)^*$ (with range $(C_2 \cup A_4)^*$) by

$$f(\tilde{a}) = apq\$_{right}, \quad f(p) = y, \quad f(q) = x,$$

$$f(c_1) = xy, \quad f(c_2) = c$$

and

$$g(\tilde{a}) = a_{right}, \quad g(p) = p, \quad g(q) = q,$$

$$g(c_1) = \Lambda, \quad g(c_2) = yx.$$

Then an (f,g)-sequence "emanating" from $\bar{a}a_1 \cdots a_m \tilde{b}$ is always as in Fig. 4 (see also Fig. 3).

It is straightforward to verify that

$$K_1 =_{def} h(\text{maxeq}(f,g) \cap \{\bar{a} \mid a \in A_1\} A A^+ \{\tilde{a} \mid a \in A_1\}) = e(K) - \{\Lambda\} - A_1 - A_1^2 - A_1^3 ,$$

where $h(a) = h(\bar{a}) = h(\tilde{a}) = a$ for $a \in A_1$. The "missing" words in $e(K) \cap (\{\Lambda\} \cup A_1 \cup A_1^2 \cup A_1^3)$, if any, can be added to K_1 as in the proof of Theorem 3.1. □

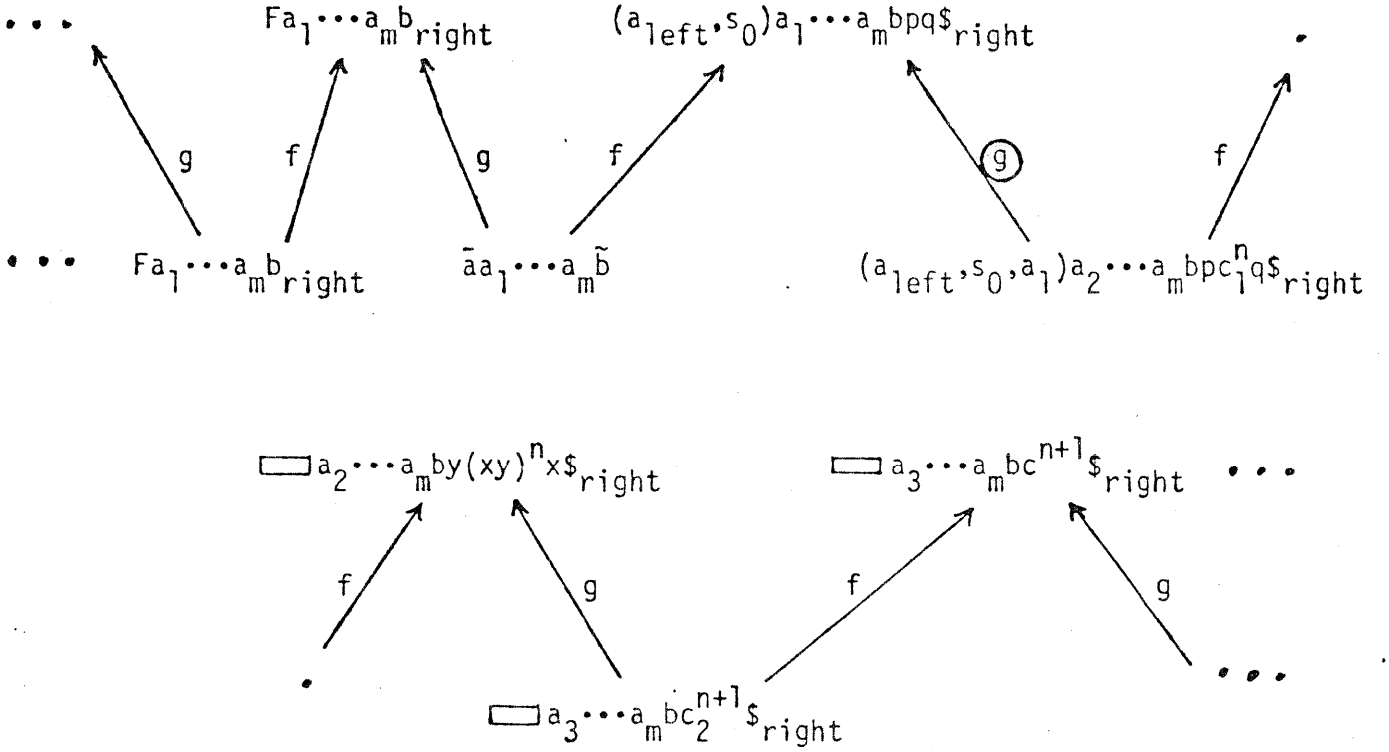

Fig. 4. Here the rectangular boxes denote those initial blocks of words which are (or may be) affected by the computation of the Turing machine simulated. To the right the sequence continues as in Fig. 3. Without restricting the case, it may be assumed that during the computation simulated here the rightmost symbol is always $\$_{right}$, cf. the proof of Lemma 4.1. Thus a symbol with a tilde can occur only once in the (f,g)-sequence. N.B. the fact that g can erase only in one step (which is circled out) and only in between p and q. In this way it becomes possible to simulate in one step the erasing of all c's.

REMARK. One can even extend the ideas of the above proof and show that $\hat{H}(\mathrm{MAX}(\mathrm{HOM})\wedge R) = H(\mathrm{MAX}(\mathrm{HOM})\wedge R)$. This goes, however, beyond the scope of the present paper.

ACKNOWLEDGEMENTS. The authors would like to thank the referees for their useful comments. The first two authors gratefully acknowledge the financial support of NSF grant number MCS 79-03838.

References

[B] Blikle, A.: Proving programs by sets of computations, in: Mathematical Foundations of Computer Science. Lecture Notes in Computer Science 28. Springer-Verlag, Berlin (1975), 333-358.

[ČS] Čulik II, K. & Salomaa, A.: On the decidability of homomorphism equivalence for languages. J. Comput. System Sci. 17 (1978), 163-175.

[dB] de Bakker, J.W.: Semantics of infinite processes using generalized trees, in: Mathematical Foundations of Computer Science 1977. Lecture Notes in Computer Science 53. Springer-Verlag, Berlin (1977), 240-246.

[dR] de Roever, W.P.: On backtracking and greatest fixpoints, in: Automata, Languages and Programming. Lecture Notes in Computer Science 52. Springer-Verlag, Berlin (1977), 412-429.

[ER1] Engelfriet, J. & Rozenberg, G.: Equality languages and fixed point languages. Inform. Control 43 (1979), 20-49.

[ER2] Engelfriet, J. & Rozenberg, G.: Fixed point languages, equality languages and representation of recursively enumerable languages. JACM, to appear.

[G] Ginsburg, S.: Algebraic and Automata-Theoretic Properties of Formal Languages. North-Holland Publ. Co., Amsterdam (1975).

[HU] Hopcroft, J.E. & Ullman, J.D.: Formal Languages and Their Relation to Automata. Addison-Wesley Publ. Co., Reading, Mass. (1969).

[M] Mazurkiewicz, A.: Proving properties of processes. CC PAS Report 134, Warsaw (1973).

[R] Rogers Jr., H.: Theory of Recursive Functions and Effective Computability. McGraw-Hill Book Co., New York (1967).

[S1] Salomaa, A.: Formal Languages. Academic Press, New York (1973).

[S2] Salomaa, A.: Equality sets for homomorphisms of free monoids. Acta Cybernet. 4 (1978), 127-140.

[T] Takahashi, H.: The maximum invariant set of an automaton system. Inform. Control 32 (1976), 307-354.

[U] Ullian, J.S.: Three theorems concerning principal AFL's. J. Comput. System Sci. 5 (1971), 304-314.