ON THE DECIDABILITY OF HOMOMORPHISM EQUIVALENCE FOR LANGUAGES*

K. Culik II
Department of Computer Science
University of Waterloo
Waterloo, Ontario, Canada

Arto Salomaa
Mathematics Department
University of Turku
Turku, Finland

CS-77-26
September 1977

* This research was supported by the National Research Council of Canada, Grant No. A7403.

Abstract

We consider decision problems of the following type. Given a language L and two homomorphisms h_1 and h_2, one has to determine to what extent h_1 and h_2 agree on L. For instance, we say that h_1 and h_2 are equivalent on L if $h_1(w) = h_2(w)$ holds for each $w \in L$. In our main theorem we present an algorithm for deciding whether two given homomorphisms are equivalent on a given context-free language. This result also gives an algorithm for deciding whether the translations defined by two deterministic gsm mappings agree on a given context-free language.

1. Introduction

Although homomorphism is a very simple and, at least from the point of view of mathematics, the most important operation defined for languages, some of the very basic questions concerning homomorphisms have turned out to be very difficult or are still unanswered. The best example of the former is the DOL equivalence problem, cf. [1] and [2], which was open for a long time. This paper investigates problems of the latter type.

The basic set-up is as follows. We are given a language L (belonging to some specified family of languages) over an alphabet Σ and two homomorphisms h_1 and h_2 mapping Σ^* into Σ_1^*, where Σ_1 is a possibly different alphabet. We want to know to what extent h_1 and h_2 "agree" on L. More specifically, we want to know whether or not the equation

$$h_1(w) = h_2(w)$$

holds (i) for some $w \in L$, (ii) for infinitely many $w \in L$, (iii) for all $w \in L$, (iv) for all $w \in L$ with a finite number of exceptions. Questions (i)-(iv) give rise to four decision problems for each particular family of languages we are considering. It is easy to see that the HDOL (sequence) equivalence problem is simply problem (iii) stated for the family of DOL languages. We feel that solutions to problems of the kind described often give important information concerning the structure of the languages considered.

A brief outline of the contents of this paper follows. After the basic definitions and preliminary results presented in Section 2 we consider in Section 3 the problems (i)-(iv) for regular and context-sensitive languages. Section 4 deals with the same problems for the family of context-free languages. In particular, we show that problem (iii) is decidable for this family. This main result of our paper,we feel,is rather surprising because several related problems are undecidable, as will be pointed out. The results in the final Section 5 concern problems slightly different from (i)-(iv) in that, in Section 5, iterated homomorphisms will be considered. However, they can be viewed as problems similar to (i)-(iv) for DTOL languages.

2. Preliminaries

We assume that the reader is familiar with the fundamental theory of formal languages including the basics of L systems, cf. [4]. However, L systems will be referred to only in some parts of the paper. For convenience, some of the definitions will be given here.

A DOL system is a triple $G = (\Sigma, h, w)$, where Σ is an alphabet, h is a homomorphism on Σ^* and w is a nonempty word over Σ . The language (resp. sequence) generated by G is defined by

$$L(G) = \{h^i(w) \mid i \geqq 0\} \quad (\text{resp. } S(G) = w, h(w), h^2(w), \ldots).$$

An HDOL system G_1 consists of a DOL system G and another homomorphism h_1 mapping Σ^* into Σ_1^* , for some alphabet Σ_1 . The language and sequence defined by G_1 are obtained from $L(G)$ and $S(G)$ by an application of the homomorphism h_1 .

A DTOL system is a tuple

$$G = (\Sigma, h_1, \ldots, h_m, w) \quad , \quad m \geqq 1 \, ,$$

where (Σ, h_i, w) is a DOL system for each i . The language generated by the DTOL system G consists of all words of the form

$$h_{j_1} h_{j_2} \ldots h_{j_k}(w) \quad , \quad k \geqq 0 \quad , \quad 1 \leqq j_i \leqq m \quad .$$

The *length* of a word w is denoted by $|w|$. For the empty word λ , $|\lambda| = 0$.

We now introduce the basic notions of this paper.

Assume that L is a language over the alphabet Σ , and that h_1 and h_2 are homomorphisms on Σ^* . Then we say that

(i) h_1 and h_2 are *compatible* on L if, for some $w \in L$, $h_1(w) = h_2(w)$;

(ii) h_1 and h_2 are *strongly compatible* on L if, for infinitely many $w \in L$, $h_1(w) = h_2(w)$;

(iii) h_1 and h_2 are *equivalent* on L if, for all $w \in L$ $h_1(w) = h_2(w)$;

(iv) h_1 and h_2 are *ultimately equivalent* on L if there is only a finite number of words $w \in L$ such that $h_1(w) \neq h_2(w)$.

As an example, consider the alphabet $\Sigma = \{a,b,c,d\}$ and homomorphisms h_1 and h_2 defined by

$$h_1(a) = aba\ ,\quad h_1(b) = b\ ,\quad h_1(c) = dd\ ,\quad h_1(d) = ab$$
$$h_2(a) = h_2(b) = h_2(c) = ab\ ,\quad h_2(d) = cc\ .$$

Then h_1 and h_2 are ultimately equivalent (but not equivalent) on the language $L(G)$, where G is the DOL system $G = (\Sigma, h_1, abc)$.

Clearly, this implies that h_1 and h_2 are strongly compatible on $L(G)$, such an implication being valid with respect to any infinite language.

The four notions introduced above define in a natural way four decision problems with respect to every effectively specified language family. Thus, we may speak of the "homomorphism compatibility problem" for regular languages. If there is no danger of confusion, we may drop the word "homomorphism" when discussing these problems.

It should be emphasized already at this point that the problem of homomorphism equivalence is not the same as the problem of deciding whether or not $h_1(L) = h_2(L)$ holds for a language L in the family we are considering. Indeed, the latter problem is undecidable for context-free languages. (This can be shown as follows. Consider arbitrary context-free languages L_1 and L_2. By providing all letters in the terminal alphabet of L_2 with a bar, we construct the "barred version" $\bar{L}_2$ of L_2. We define now

$$L = L_1\bar{L}_2 \ , \quad h_1(a) = h_2(\bar{a}) = \lambda \ , \quad h_1(\bar{a}) = h_2(a) = a \ ,$$

for all letters a. Then $h_1(L) = h_2(L)$ if and only if $L_1 = L_2$.) However, in Section 4 we shall prove that the problem of homomorphism equivalence is decidable for context-free languages.

A very important tool in the proofs below will be the notion of balance defined as follows.

Consider two homomorphisms h_1 and h_2 defined on Σ^* and a word $w \in \Sigma^*$. Then the balance of w is defined by

$$\beta(w) = |h_1(w)| - |h_2(w)| \quad .$$

(Thus $\beta(w)$ is an integer depending, apart from w , also on h_1 and h_2 . However, we write it simply $\beta(w)$ because the homomorphisms, as well as their ordering, will always be clear from the context.) Note that the balance of w in [2] was defined as $|\beta(w)|$ in our notation.

It is an immediate consequence of the definition that

$$\beta(w_1w_2) = \beta(w_1) + \beta(w_2) \quad .$$

A repeated application of this equation shows that the balance of a word w depends only on the Parikh vector of w .

We say that the pair (h_1, h_2) has bounded balance on a given language L if there exists a constant C such that

$$|\beta(w)| \leq C$$

holds for all initial subwords w of the words in L .

The property of having bounded balance gives a method of deciding homomorphism equivalence. More specifically, we can state this as follows.

We call a family $\mathcal{L}$ of languages smooth if each of the following conditions (i)-(iii) is satisfied:

i) $\mathcal{L}$ is effectively closed under deterministic gsm mappings;

ii) The emptiness problem is decidable for languages in $\mathcal{L}$;

iii) For each language L in $\mathcal{L}$ and each pair of homomorphisms (h_1, h_2) , whenever h_1 and h_2 are equivalent on L then (h_1, h_2) has bounded balance on L .

(Certain fixed finite representation for languages from $\mathcal{L}$ is considered.)

An obvious modification of the proof of Theorem 2.1 in [1] gives now the following

Theorem 2.1 The problem of homomorphism equivalence is decidable for any smooth family $\mathcal{L}$.

As an example, consider the family of regular languages. That it is smooth follows directly from the proof of Theorem 5 in [2] . This can be established also by the following argument. Consider a regular language L and two homomorphisms h_1 and h_2 equivalent on L . We consider the minimal finite deterministic automaton accepting L . Any word w causing a loop in the automaton (i.e., mapping some state into itself) must satisfy

$$\beta(w) = 0 .$$

(Otherwise, we would have $\beta(w_1 w^n w_2) \neq 0$ and $w_1 w^n w_2 \in L$, for some words w_1 and w_2 and some sufficiently large number n . Hence, we would have

$$h_1(w_1 w^n w_2) \neq h_2(w_1 w^n w_2) ,$$

a contradiction.) Thus, an upper bound for the balance of initial subwords of the words in L can be computed by considering such words w only which cause a transition from the initial state to one of the final states without loops. Clearly, the number of such words w is finite.

On the other hand, the family of context-free languages is not smooth. A simple example showing this is provided by the language

$$L = \{a^n b^n \mid n \geqq 1\}$$

and homomorphisms h_1 and h_2 defined by

$$h_1(a) = h_2(b) = aa \quad , \quad h_2(a) = h_1(b) = a \quad .$$

Clearly, h_1 and h_2 are equivalent on L but the balance on initial subwords a^n is unbounded.

We will show in Section 4 that, in spite of the fact that the family is not smooth, the homomorphism equivalence problem is still decidable for the family of context-free languages. The argument will show that situations (like the one in the example above) caused by the Pumping Lemma are, in fact, the only ones where the balance may grow unbounded.

We note, finally, that it is still an open problem whether the family of DOL languages satisfies condition (iii) given in the definition of smoothness. (This is really the essential condition. The other two conditions can be modified in various ways without affecting the validity of

Theorem 2.1.) It is also an open problem whether or not homomorphism equivalence is decidable for the family of DOL languages. As regards this problem, we can give the following reduction result.

<u>Theorem 2.2</u> Homomorphism equivalence is decidable for the family of DOL languages if and only if sequence equivalence is decidable for HDOL systems.

<u>Proof</u> The "if" - part is obvious. To establish the "only if" - part, assume that homomorphism equivalence is decidable for the family of DOL languages. This enables us to decide whether two given HDOL sequences

$$h_3(h_1^n(w_1)) \quad \text{and} \quad h_4(h_2^n(w_2)) \ , \ n = 0,1,2,\ldots \ ,$$

coincide as follows. Assume without loss of generality that

$$h_1 : \Sigma_1^* \to \Sigma_1^* \ , \ h_2 : \Sigma_2^* \to \Sigma_2^* \ , \ h_3 : \Sigma_1^* \to \Delta^* \ , \ h_4 : \Sigma_2^* \to \Delta^*$$

where Σ_1 and Σ_2 are disjoint. Consider the DOL system $G = (\Sigma_1 \cup \Sigma_2, h, w_1w_2)$ with $h(a) = h_i(a)$ for $a \in \Sigma_i$, $i = 1,2$. Define two homomorphisms h_5 and h_6 by

$$h_5(a) = h_3(a) \quad \text{for} \quad a \in \Sigma_1 \ , \quad h_5(a) = \lambda \quad \text{for} \quad a \in \Sigma_2 \ ,$$
$$h_6(a) = \lambda \quad \text{for} \quad a \in \Sigma_1 \ , \quad h_6(a) = h_4(a) \quad \text{for} \quad a \in \Sigma_2 \ .$$

Then h_5 and h_6 are equivalent on $L(G)$ if and only if the original HDOL sequences are the same.

3. Decidability Results for Regular and Context-Sensitive Languages

The following two sections establish the decidability status of the four decision problems, mentioned in Section 2, for the language families in the Chomsky hierarchy. We consider the hierarchy up to deterministic context-sensitive languages only because already at this level all problems become undecidable. As corollaries we obtain also some related results, for instance, concerning the equivalence of two deterministic gsm mappings on a given language L.

Intuitively, decision problems concerning homomorphism compatibility are more difficult than those concerning homomorphism equivalence. Also deciding ultimate equivalence is harder than deciding equivalence. The results and proofs below show that this is indeed the case.

Theorem 3.1 The problems of homomorphism compatibility and strong compatibility are undecidable for the family of regular languages.

Proof The theorem is a direct consequence of the fact that an instance PCP of the Post Correspondence Problem

$$(\alpha_1, \ldots, \alpha_n) , \quad (\beta_1, \ldots, \beta_n)$$

can be viewed as a compatibility problem on the language $\{1, \ldots, n\}^*$. PCP has a solution exactly in case it has infinitely many of them.

□

Theorem 3.2 The problems of homomorphism equivalence and ultimate equivalence are undecidable for the family of deterministic context-sensitive languages.

Proof It is well known that the equivalence problem for deterministic linearly bounded transducers (the memory is linearly bounded by the length of the input) is undecidable. To show the undecidability of the homomorphic equivalence problem for the family of deterministic context-sensitive languages, we consider the following reduction of the equivalence problem for deterministic linearly bounded transducers.

Let M_1 and M_2 be two given deterministic linearly bounded transducers with input alphabet Σ and output alphabet Δ . We provide first the outputs of M_2 be primes, yielding the alphabet Δ' . (We assume that Σ , Δ and Δ' are pairwise disjoint.) Clearly, the language

$$L = \{xyz \mid x \in \Sigma^*, y \in \Delta^*, z \in (\Delta')^*, M_1(x) = y, M_2(x) = z\}$$

is deterministic context-sensitive. We now define two homomorphisms h_1 and h_2 by $h_1(a) = h_2(a) = a$ for $a \in \Sigma$,

$$h_1(a) = a \ , \ h_2(a) = \lambda \ , \ h_1(a') = \lambda \ , \ h_2(a') = a \quad \text{for} \quad a \in \Delta.$$

Then M_1 and M_2 are equivalent if and only if h_1 and h_2 are equivalent on L .

The undecidability of the ultimate equivalence problem is shown in the same way, beginning with the fact that the ultimate equivalence problem is undecidable for deterministic linearly bounded transducers.

□

<u>Theorem 3.3</u> The problems of homomorphism equivalence and ultimate equivalence are decidable for the family of regular languages.

<u>Proof</u> The statement concerning equivalence follows by Theorem 2.1. The decidability of ultimate equivalence is shown by the argument presented in Section 2 to show the smoothness of the family of regular languages. In fact, if h_1 and h_2 are ultimately equivalent on a regular language L then $\beta(w) = 0$ for all words w causing a loop in the automaton accepting L. Thus, h_1 and h_2 are ultimately equivalent on L if and only if they are equivalent on the regular language L_1 obtained from L by removing all words of a length smaller than the number of states in the automaton.

□

We conclude this section by a result showing how the decidability of homomorphic equivalence implies the decidability of deterministic gsm equivalence. More specifically, we say that two deterministic gsm's M_1 and M_2 are <u>equivalent</u> on a language L if $M_1(w) = M_2(w)$ holds for all $w \in L$.

Given a family $\mathcal{L}$ of languages, we can in a natural way speak about the deterministic gsm equivalence problem for $\mathcal{L}$.

<u>Theorem 3.4</u> Assume that $\mathcal{L}$ is a family of languages effectively closed under deterministic gsm mappings and that the homomorphism equivalence problem is decidable for $\mathcal{L}$. Then the deterministic gsm equivalence problem is decidable for $\mathcal{L}$.

<u>Proof</u> Consider an arbitrary $L \in \mathcal{L}$, $L \subseteq \Sigma^*$, and two deterministic gsm's M_1 and M_2 with input alphabet Σ and output alphabet Δ. We first provide the output letters of M_2 with primes, yielding the alphabet Δ', and assume without loss of generality that Σ, Δ and Δ' are pairwise disjoint. We then replace M_1 by the deterministic gsm M_1' obtained from M_1 as follows. Each instruction $(s_1, a;\ w, s_2)$ is replaced by $(s_1, a;\ aw, s_2)$. (The instruction $(s_1, a;\ w, s_2)$ means: in the state s_1 when scanning the input letter a, go to the state s_2 and output the word w.) Thus, the input alphabet of M_1' is Σ, output alphabet being $\Sigma \cup \Delta$. Finally, we replace M_2 by the deterministic gsm M_2' obtained from M_2 as follows. For each state s of M_2 and each letter a of Δ, the instruction $(s, a;\ a, s)$ is added. Thus, the input alphabet of M_2' is $\Sigma \cup \Delta$, output alphabet being $\Delta \cup \Delta'$.

Consider now the language $L_1 = M_2'(M_1'(L))$

Given a family $\mathcal{L}$ of languages, we can in a natural way speak about the deterministic gsm equivalence problem for $\mathcal{L}$.

<u>Theorem 3.4</u> Assume that $\mathcal{L}$ is a family of languages with the following properties:

(i) $\mathcal{L}$ is effectively closed under deterministic gsm mappings;

(ii) the emptiness problem for $\mathcal{L}$ is decidable;

(iii) the homomorphism equivalence problem for $\mathcal{L}$ is decidable.

Then the deterministic gsm equivalence problem is decidable for $\mathcal{L}$.

<u>Proof</u> Consider an arbitrary $L \in \mathcal{L}$, $L \subseteq \Sigma^*$ and two deterministic gsm's M_1 and M_2 with input alphabet Σ and output alphabet Δ . Let $R_i = \text{dom } M_i$, $i = 1,2$, and R_3 is the symmetric difference of R_1 and R_2 . Clearly, R_i is regular, $i = 1,2,3$. Now, M_1 and M_2 are equivalent on L iff they are equivalent on $L' = L \cap R_1 \cap R_2$ and $L \cap R_3 = \phi$. Since intersection of L with a regular set can be expressed as the result of a deterministic gsm mapping applied to L we have $L' \in \mathcal{L}$ and $L \cap R_3 \in \mathcal{L}$. Since the emptiness problem for $\mathcal{L}$ is decidable we can check whether $L \cap R_3 = 0$, if so we proceed to check the equivalence of M_1 and M_2 on L' . We remind that $L' \in \mathcal{L}$ and M_1, M_2 are defined $(M_i(w) \neq \phi$, $i = 1,2)$ for each $w \in L'$.

Now, we provide the output letters of M_2 with primes, yielding the alphabet Δ', and assume without loss of generality that Σ , Δ and Δ' are pairwise disjoint. We then replace M_1 by the deterministic gsm M_1' obtained from M_1 as follows.

Each instruction $(s_1,a;\ w,s_2)$ is replaced by $(s_1,a;\ aw,s_2)$. (The instruction $(s_1,a;\ w,s_2)$ means: in the state s_1 when scanning the input letter a, go to the state s_2 and output the word w.) Thus, the input alphabet of M_1' is Σ, output alphabet being $\Sigma \cup \Delta$. Finally, we replace M_2 by the deterministic gsm M_2' obtained from M_2 as follows. For each state s of M_2 and each letter a of Δ, the instruction $(s,a;\ a,s)$ is added. Thus, the input alphabet of M_2' is $\Sigma \cup \Delta$, output alphabet being $\Delta \cup \Delta'$.

Consider now the language $L_1 = M_2'(M_1'(L'))$ over the alphabet $\Delta \cup \Delta'$. By the assumption, L_1 is in the family L and can be effectively constructed. Define two homomorphisms h_1 and h_2 by

$$h_1(a) = h_2(a') = a\ ,\ h_1(a') = h_2(a) = \lambda \quad \text{for} \quad a \in \Delta\ .$$

Then M_1 and M_2 are equivalent on L' if and only if h_1 and h_2 are equivalent on L_1.

□

The following result is an immediate consequence of Theorems 3.3 and 3.4. It can be obtained also directly using the fact that the equivalence of deterministic gsm's is decidable.

<u>Theorem 3.5</u> The deterministic gsm equivalence problem is decidable for the family of regular languages.

Applying Theorem 3.5 to the language Σ^* we get another proof of the fact that the equivalence of deterministic gsm's is decidable.

former is obvious. The latter consists of checking, for $k = 0,1,2,\ldots$, whether or not h'_1 and h'_2 are equivalent on L' with balance bounded by k . This can be done by deciding the emptiness of the context-free language $M_k(L')$, where M_k is a deterministic gsm with a "buffer" of length k in its finite control.

We now begin the proof of Theorem 4.2. Without loss of generality, we assume that L is an infinite language, generated by a reduced context-free grammar G , where every nonterminal generates an infinite language and there are no productions of the form $A \to B$ with A and B nonterminals.

The following observation will be used throughout the proof without explicit mentioning. When analyzing G , if we meet a situation showing that $h_1(w) \neq h_2(w)$ for some $w \in L$, we may stop the construction immediately (and choose $L = L'$). Thus, we may assume that such situations do not arise during our construction.

The following simple lemma is of basic importance.

<u>Lemma 4.3</u> Assume that $B \Rightarrow^* vBx$ is a derivation according to G , where v and x are terminal words. Then $\beta(vx) = 0$, or else h_1 and h_2 are not equivalent on L .

Lemma 4.3 is established as follows. For some u, w, y, all words

$$P_n = uv^n wx^n y$$

are in L . Thus, $\beta(vx) \neq 0$ implies that $\beta(P_n) \neq 0$, for all sufficiently large n .

□

By Lemma 4.3 and the observation preceding it, we assume that in all situations encountered in our process we actually have $\beta(vx) = 0$.

Lemma 4.4 For every nonterminal B of G , $\beta(w)$ is constant for all terminal words w such that $B \Rightarrow^* w$ (or else h_1 and h_2 are not equivalent on L). This constant, say $\beta(B)$, can be computed from any terminal word generated from B .

The proof of Lemma 4.4 is obvious. Also the following lemma is easily established by the "shifting" argument used in [1] . (In fact, the situation here is much simpler than the one considered in the proof of Theorem 3.2 in [1].)

Lemma 4.5 Assume that $B \Rightarrow^* vBx$ and $\beta(v) \neq 0$. Denote by L_B the language generated by B . Then there is a word p (referred to as a period of B) such that

$$h_1(L_B) \subseteq p^* \quad , \quad h_2(L_B) \subseteq p'^* \; ,$$

where for some words p_1 and p_2 ,

$$p = p_1 p_2 \; , \quad p' = p_2 p_1$$

(or else h_1 and h_2 are not equivalent on L).

Using Lemma 4.5, we shall classify nonterminals of G as "periodic" or "nonperiodic". Lemma 4.5 shows that in recursive situations $B \Rightarrow^* vBx$ we can have $\beta(v) \neq 0$ (or

$\beta(x) \neq 0$, cf, Lemma 4.3) only in connection with periodic nonterminals B . In our algorithm we will test nonterminals for periodicity in all simple derivation loops. Nonterminals found to be periodic and consistent with the assumption that h_1 and h_2 are equivalent on L are finally replaced by their periods, yielding the language L' of Theorem 4.2.

To describe the algorithm more formally, we introduce some terminology. We say that the derivation

$$B \Rightarrow^* \alpha_1 B \alpha_2$$

is a simple recurrence situation (SRS in short) for the nonterminal B if (i) B does not occur in the intermediate steps (if any) in the derivation, and (ii) no nonterminal other than B occurs twice in any path in the derivation tree.

Thus, if $B \to aBABBA$ is a production of G then the derivation

$$B \Rightarrow aBABBA$$

is an SRS for B . In fact, we can interpret this derivation to be an SRS in three different ways because there are three possibilities for the choice of the pair (α_1, α_2) .

With each nonterminal B we associate the finite language F_B consisting of terminal words derived from B without loops, i.e., no path in the derivation tree contains two occurrences of the same nonterminal.

Consider now, for each nonterminal B , derivations

$$B =>^* vBx$$

such that the pair (v, x) is obtained from the pair (α_1, α_2) in some SRS for B by replacing every occurrence of a non-terminal A with some word in F_A . (Different occurrences may be replaced with different words.) Clearly, there is only a finite number of such derivations.

For each such derivation, we test whether there are words p and p' , p' obtained by shifting an initial subword of p to the end as in Lemma 4.5, such that

$$\{h_1(v^n wx^n) \mid n \geqq 0 , \quad w \in F_B\} \subseteq p^*$$

and

$$\{h_2(v^n wx^n) \mid n \geqq 0 , \quad w \in F_B\} \subseteq p'^* \quad .$$

If the answer is "yes", we choose the period p to be the shortest period and classify B as "periodic". If the answer is "no", we classify B as "nonperiodic". Periodic non-terminal B satisfying $\beta(v) \neq 0$ (in the derivation $B =>^* vBx$ we are considering) is referred to as "properly periodic".

We associate with each nonterminal B classified as "periodic" the pair (p, p') . For properly periodic non-terminals, this pair must be the same in all situations encountered, and they may not be classified as "nonperiodic" at later stages. (If we encounter a situation violating either one of these conditions, we may stop the process with the condition that h_1 and h_2 are not equivalent on L .) On the other hand, periodic nonterminals may be classified as "nonperiodic" at later stages. Thus, whenever we are about to classify a nonterminal as "nonperiodic", we have to check that this nonterminal has not been classified as "properly periodic" at previous stages. Once a nonterminal is classified as "nonperiodic", the classification cannot be changed at later stages. If B is properly periodic with periods (p, p') , we replace it at later stages of the procedure, when going from (α_1, α_2) to (v, x) and then taking the homomorphic images, simply by the periods p and p' .

After going through all of the derivations $B \Rightarrow^* vBx$ (as defined above) for all nonterminals B , we introduce for each nonterminal B classified as "properly periodic" with periods (p, p') a new terminal letter a_B . The homomorphisms h_1 and h_2 are extended (extensions being denoted by h_1' and h_2') to the larger alphabet Σ' obtained in this fashion by defining

$$h_1'(a) = h_1(a),\ h_2'(a) = h_2(a) \quad \text{for} \quad a \in \Sigma ,$$
$$h_1'(a_B) = p ,\quad h_2'(a_B) = p' \quad \text{for each} \quad a_B .$$

The language L' is now generated by the grammar G' obtained from G by replacing every properly periodic nonterminal B with a_B .

If h_1' and h_2' are equivalent on L' then the pair (h_1', h_2') has bounded balance on L'. This follows because we have eliminated all situations where the balance might grow unbounded; the balance is different from zero only in the finitely many situations essentially corresponding to derivations without loops. In more detail, if there are no properly periodic nonterminals, then there exists constants $p_1, p_2 > 0$ so that every string u in L' can be reduced to a string v in L', $|v| \le p_1$, by omitting substrings of balance zero and length no more than p_2. Therefore the balance $\beta(w)$ of every prefix w of L' is clearly bounded by $C_1 + C_2$, where C_i is the bound on the balance of the prefixes of the finite language $\{x \in \Sigma^* : |x| \le p_i\}$, for $i = 1,2$.

It follows from the notion of a properly periodic nonterminal and Lemma 4.5 that h_1 and h_2 are equivalent on L if and only if h_1' and h_2' are equivalent on L'.

□

We omit the proof of the following theorem. It is essentially the same as the proof above, the basic observation being that the discussions concerning properly periodic non-terminals remain unaltered. Thus, the finite number of exceptions to the equation $h_1(w) = h_2(w)$, $w \in L$, must occur for words w whose derivation does not involve properly periodic nonterminals. The only difference is now that we have to check the finiteness (instead of the emptiness) of a language obtained by applying a deterministic gsm to a context-free language.

Theorem 4.6 The problem of homomorphism ultimate equivalence is decidable for the family of context-free languages.

The following table summarizes our results.

	regular	context-free	deterministic context-sensitive
compatibility	undecidable	undecidable	undecidable
strong compatibility	"	"	"
equivalence	decidable	decidable	decidable
ultimate equivalence	"	"	"

Theorem 3.4 now yields immediately the following result.

Theroem 4.7 The deterministic gsm equivalence problem is decidable for the family of context-free languages.

Since deterministic gsm mappings can be viewed as translations, we have here a decidability result concerning the equivalence of such translations of context-free languages.

5. Iterated Homomorphisms

In this final section, the problems considered will be slightly different from those discussed above.

Consider two finite languages

$$f = \{w_1, \ldots, w_m\} , \qquad F' = \{w_1', \ldots, w_m'\}$$

of the same cardinality m and over the same alphabet Σ , and two n-tuples of homomorphisms

$$(h_1, \ldots, h_n) \quad , \quad \text{and} \qquad (h_1', \ldots, h_n')$$

defined on Σ^* . Thus, each element of F together with the n-tuple $(h_1, \ldots, h_n)$ defines a DTOL system. The whole set F together with this n-tuple defines a so-called DTOL system with finitely many axioms or, shortly, FDTOL system. Call the two FDTOL systems obtained in this fashion G and G' .

We call G and G' <u>compatible</u>, <u>strongly compatible</u>, <u>equivalent</u>, and <u>ultimately equivalent</u> if the equation

$$h_{i_1} h_{i_2} \ldots h_{i_k} (w_j) = h_{i_1}' h_{i_2}' \ldots h_{i_k}' (w_j')$$

holds for one choice of the sequence $i_1 i_2 \ldots i_k$ and number j , for infinitely many such choices, for all choices of the sequence $i_1 i_2 \ldots i_k$ and number j , and for all but finitely many choices, respectively. When we in this section speak of compatibility and equivalence problems, we mean problems in this set-up.

Note that, for $m = n = 1$, the equivalence and ultimate equivalence problem defined above coincide with the equivalence and ultimate equivalence problem for DOL sequences. For $m = 1$ and general n, the equivalence problem may be viewed as the equivalence problem for DTOL sequences. In this case the equivalence problem is also related to the homomorphism equivalence problem (in the sense of the previous sections of this paper) for the family of DTOL languages.

Our starting point above is two finite languages. We could also start, for instance, from two regular languages with a 1-to-1 correspondence between their words.

We shall prove that the compatibility and strong compatibility problems are undecidable, whereas the status of the equivalence and ultimate equivalence problems remains open; we only give some reduction results for these problems. We strongly suspect that the equivalence problem is decidable, at least if the homomorphisms are assumed to be nonerasing. One reason for this is that the techniques showing undecidability (such as those in [5]) all seem to fail because one is not able to keep track of the "matching" of the two sets of homomorphisms. Note also that the equivalence problem for DTOL languages is known to be undecidable but, thus, the equivalence problem for DTOL sequences remains open. In the case of DOL systems, we can reduce either one of these problem to the other, cf. [3]. This reduction does not work for DTOL systems.

Theorem 5.1 Compatibility and strong compatibility problems are undecidable, even when restricted to the case where $m = 1$ and $n = 2$.

Proof Consider an arbitrary instance PCP $(\alpha_1, \ldots, \alpha_k)$, $(\beta_1, \ldots, \beta_k)$ of the Post Correspondence Problem. We choose

$$w_1 = A \quad , \qquad w_1' = B$$

and define two pairs of homomorphisms as follows:

$$\begin{array}{llll} h_1(A) = A_1 & , \quad h_1'(B) = B_1 & , & \\ h_1(A_i) = A_{i+1} & , \quad h_1'(B_i) = B_{i+1} & , & 1 \leqq i \leqq k-1 \ , \\ h_1(A_i') = A_{i+1}' & , \quad h_1'(B_i') = B_{i+1}' & , & 1 \leqq i \leqq k-1 \ , \\ h_1(A_k) = A_k & , \quad h_1'(B_k) = B_k & , & \\ h_1(A_k') = \lambda & , \quad h_1'(B_k') = \lambda & , & \end{array}$$

$$h_2(A_i) = h_2(A_i') = \alpha_i A_1' \quad , \quad h_2'(B_i) = h_2'(B_i') = \beta_i B_1' \ ,$$

for $1 \leqq i \leqq k$. For all other letters x ,

$$h_1(x) = h_1'(x) = h_2(x) = h_2'(x) = x \quad .$$

(It is assumed that the alphabet of PCP does not contain any of the A's and B's introduced above.) Then PCP has a solution (resp. infinitely many solutions) if and only if the systems (w_1, h_1, k_2) and (w_1', h_1', k_2') are compatible (resp. strongly compatible).

□

Theorem 5.2 Equivalence problem is decidable if and only if it is decidable for systems with $m = 1$ and $n = 2$.

Proof The "only if" - part in obvious. To prove the "if" - part, we note first that assuming $m = 1$ does not restrict generality. If we have an algorithm for the case, where both of the systems have two homomorphisms, we can extend this algorithm to the general case by introducing "idle steps" as follows. Suppose we want to have an algorithm for deciding the equivalence of the systems

$$(w_1, h_1, h_2, h_3, h_4) \ , \quad (w_1, h_1', h_2', h_3', h_4') \ .$$

We then define two systems

$$(w_1, g_1, g_2) \ , \qquad (w_1, g_1', g_2')$$

as follows. For each letter a of the original alphabet, we introduce two new letters a' and a'' and define

$$g_1(a) = g_1'(a) = a' \ , \quad g_2(a) = g_2'(a) = a'' \ ,$$

$$g_1(a') = h_1(a) \ , \quad g_1(a'') = h_2(a) \ , \quad g_2(a') = h_3(a) \ ,$$

$$g_2(a'') = h_4(a) \ , \quad g_1'(a') = h_1'(a) \ , \quad g_1'(a'') = h_2'(a) \ ,$$

$$g_2'(a') = h_3'(a) \ , \quad g_2'(a'') = h_4'(a) \ .$$

Clearly, the original systems are equivalent if and only if the new ones are. An obvious inductive argument now generalizes the result to the case of arbitrary many homomorphisms.

□

<u>Theorem 5.3</u> The decidability of the equivalence problem implies the decidability of the HDOL sequence equivalence problem.

<u>Proof</u> Consider two HDOL sequences, given as in the proof of Theorem 2.2. Define two pairs of homomorphisms (g_1, g_2), (g_1', g_2') as follows:

$$g_1(a) = g_1'(a) = h_1(a) \quad \text{for } a \in \Sigma_1 ,$$

$$g_1(a) = g_1'(a) = h_2(a) \quad \text{for } a \in \Sigma_2 ,$$

$$g_1(a) = g_1'(a) = \lambda \quad \text{for } a \in \Delta ,$$

$$g_2(a) = h_3(a) , \quad \text{for } a \in \Sigma_1 ,$$

$$g_2(a) = \lambda , \quad \text{for } a \in \Sigma_2 \cup \Delta ,$$

$$g_2'(a) = h_4(a) , \quad \text{for } a \in \Sigma_2 ,$$

$$g_2'(a) = \lambda , \quad \text{for } a \in \Sigma_1 \cup \Delta .$$

Then the systems

$$(w_1w_2, g_1, g_2) , \quad (w_1w_2, g_1', g_2')$$

are equivalent if and only if the given HDOL sequences are.

□

6. Open Problems

The most interesting open problem in the discussions above is the decidability of the equivalence problem introduced in Section 5. We conjecture the problem to be decidable, at least if the homomorphisms are assumed to be nonearasing.

An interesting problem area is to study to what extent Theorem 4.7 remains valid for other language families. It might be still valid for EOL, ETOL, or even indexed languages.

References

[1] K. Culik II, On the decidability of the sequence equivalence problem for DOL-systems. Theoretical Computer Science 3 (1977), 75-84.

[2] K. Culik II and J. Fris, The decidability of the equivalence problem for DOL-systems. Information and Control, to appear.

[3] M. Nielsen, On the decidability of some equivalence problems for DOL-systems. Information and Control 25 (1974), 166-193.

[4] A. Salomaa, Formal Languages. Academic Press (1973).

[5] A. Salomaa, On sentential forms of context-free grammars. Acta Informatica 2 (1973), 40-49.