A Polynomial Time Algorithm for the Equivalence of two Morphisms on $\omega$-Regular Languages

Stefano Varricchio

Dipartimento di Matematica Università di Catania, Italy.

Abstract. Let $L \subseteq A^\omega$ be an $\omega$-regular language given by means of a non-deterministic Büchi automaton $M$. Let $f, g : A^\omega \rightarrow B^\omega$ be two morphisms. We give an algorithm to decide whether $f$ and $g$ are equivalent (word by word) on $L$. This algorithm has time complexity $O(mn^3)$, where $n$ is the number of arcs of $M$ and $m$ is the size of $f$ and $g$. This result improves the only known algorithm for this problem which is exponential time [3].

1 Introduction and preliminaries

Infinite words and sets of them ($\omega$-languages) played a central role in Computer Science and Formal Languages theory since the beginning. The theory of $\omega$-regular languages, started with [2], studies the behavior of finite automata on infinite words. Morphisms of free monoids have always had great importance in this context, since they can express many undecidable problems. The following statement proved in [1] has been an open problem for many years (Ehrenfeucht's conjecture): Any language $L$ of a finitely generated free monoid has a finite subset $L'$ such that whenever two morphisms are equivalent on $L'$ (word by word) they are equivalent on $L$ (the set $L'$ is also called a test set for $L$). This property has been generalized to $\omega$-languages in [7].

In principle the existence of finite test sets might assure the decidability of the equivalence of two morphism on a language $L$; unfortunately this is not the case, since the proof of Ehrenfeucht's conjecture is purely existential and in general no algorithm may be given to construct a finite test set. Any way for some families of formal languages (regular, context-free, DOL languages) an effective procedure can be given. In a recent paper [6] was proved that for a regular language $L$ a finite test set can be found by an algorithm which is polynomial time with respect to the number of arcs of a non deterministic finite automaton which recognizes $L$. This also gives a polynomial time algorithm for the equivalence of two morphism on a regular language.

The equivalence problem for morphisms on $\omega$-regular languages has been proved to be decidable in [3], anyway the algorithm proposed is exponential time. In this paper we give an algorithm to decide the equivalence of two morphism $g, h$ on an $\omega$-regular language $L$. The algorithm is polynomial time with respect to the size of $g, h$ and the number of arcs of a Büchi automaton which recognizes $L$.

Let $A$ be a finite non-empty set, or alphabet, and $A^*$ (resp. $A^+$) the free semigroup (resp. free monoid) over $A$. The elements of $A$ are called letters and those of $A^*$ words. The identity element of $A^*$ is called empty word and denoted by $\lambda$. For any word $w$, $|w|$ denotes its length. A word $u$ is a factor (resp. prefix) of the word $w$ if $w \in A^*uA^*$ (resp. $w \in uA^*$). For any $w \in A^*$, $F(w)$ (resp. $P(w)$) denotes the set of all its factors (resp prefixes). A language $L$ over the alphabet $A$ is any subset of $A^*$. By $F(L)$ (resp. $P(L)$) we denote the set of all factors (resp. prefixes) of the words of $L$. A word $w \in A^*$ is called primitive if $w \neq u^h$ with $h > 1$ and $u \neq \lambda$. For any $w \in A^+$ the primitive root of $w$ is defined as the unique primitive word $z$ such that $w = z^h$ for some $h \geq 1$. Let $x, y \in$
We say that \( x, y \) are *conjugate* if there exist \( u, v \in A^* \) such that \( x = uv \) and \( y = vu \). For any \( x \in A^* \), \( x = a_1 \ldots a_n \), we call \( C(x) \) the cyclic permutation of \( x \) defined by \( C(x) = a_2 \ldots a_n a_1 \). For any integer \( p \) we will denote by \( C_p \) the cyclic permutation \( C^p \). It is easy to see that \( p \equiv q \pmod{\ell_x} \) implies \( C_p(x) = C_p(y) \) and if \( x \) is primitive the reverse holds, indeed any primitive word \( x \) has exactly \( \ell_x \) different conjugates (cf. [5]).

Let \( \mathbb{N}_+ \) be the set of positive integers. A one-sided (from left to right) infinite word is any map \( w : \mathbb{N}_+ \to A \). For each \( n > 0 \) the factor \( w[1, n] = w_1 \ldots w_n \) of length \( n \) is called the *prefix* of \( w \) of length \( n \) and will be simply denoted by \( w[n] \). The set of all infinite words \( w : \mathbb{N}_+ \to A \) will be denoted by \( A^\omega \), moreover we set \( A^\omega = A^* \cup A^\omega \). An application \( f : A^\omega \to B^\omega \) is a *morphism* if for any \( a \in A \), \( f(a) \in B^* \) and for \( x = a_1 \ldots a_n \) (resp. \( x = a_1 \ldots a_n \ldots \)) one has \( f(x) = f(a_1) \ldots f(a_n) \) (resp. \( f(x) = f(a_1) \ldots f(a_n) \ldots \)).

Given two morphisms \( f, g : A^\omega \to B^\omega \), we set \( E(f, g) = \{ y \in A^\omega \mid f(y) = g(y) \} \) and \( \text{size}(f) = \max\{ |f(x)| \mid x \in A \} \). For any word \( u \in A^+ \) we denote by \( u^\omega \) the infinite word \( uu \ldots \) obtained by the infinite concatenation of \( u \) with itself.

We write a (non deterministic) Büchi automaton \( M = (Q, A, \delta, \varepsilon, F) \), where \( Q \) is a finite set of states, \( A \) is the input alphabet, \( \delta : Q \times A \to 2^Q \) is the transition function, \( \varepsilon \) is the initial state and \( F \) is the set of final states. The set of the edges of \( M \) is defined by \( E_M = \{(p,a,q) \mid p, a, q \in Q \times A \times Q \} \). Let \( w = a_0 \ldots a_n \ldots \), we say that an infinite sequence \( p = p_0 \ldots p_n \ldots \) of states is a run of \( w \) if \( p_0 = \varepsilon \) and for any \( i \geq 1 \) \( p_i \in \delta(p_{i-1}, a_i) \). We denote by \( \text{Inf}(p) \) the set of states that are infinitely many times repeated in \( p \) and we say that an infinite word \( w \) is accepted by \( M \) if there exists a run \( p \) of \( w \) such that \( \text{Inf}(p) \) contains at least an element of \( F \). The set of all infinite words accepted by \( M \) is denoted by \( L^\omega(M) \).

**2 Preliminary results**

We start this section with some technical lemmas that will be useful in the sequel.

**Lemma 2.1.** Let \( x \in A^+ \), \( w \in A^\omega \). If \( w = xw \) then \( w = x^\omega \).

**Proof.** Since \( w = xw \), by substituting one has \( w = xxw = xxxw = \ldots \) and by iteration one obtains \( w = x^\omega \). Q.E.D.

**Lemma 2.2.** Let \( \gamma \in A^\omega \) and \( x, y \in A^+ \) be primitive words such that \( \gamma = \lambda x^\omega = \mu y^\omega \) for suitable \( \lambda, \mu \in A^* \). Then \( x, y \) are conjugate and moreover \( x = y \) if and only if \( \lambda \equiv \mu \pmod{\ell_x} \).

**Proof.** By hypothesis we can write \( \gamma = uy \gamma \) where \( \gamma \) is an infinite word having periods \( \ell_x \) and \( \ell_y \). By the Fine and Wilf theorem (cf. [5]) \( \gamma \) has period \( \text{M.C.D.}(\ell_x, \ell_y) \). Since \( x \) and \( y \) are primitive, this implies that \( |x| = |y| \). An easy computation shows that \( x, y \) are conjugate and \( x = C_{|x| - |\mu|}(y) \). Since \( x, y \) are primitive, one has \( x = y \) if and only if \( |x| = |\mu| \pmod{\ell_x} \). Q.E.D.

Let \( M = (Q, A, \delta, \varepsilon, F) \) be a Büchi automaton. A path of \( M \) is a sequence \( \pi = (p_0, a_1, p_1)(p_2, a_2, p_2) \ldots (p_{n-1}, a_n, p_n) \) such that \( (p_{i-1}, a_i, p_i) \in E_M \) for \( 1 \leq i \leq n \), and the label of \( \pi \) is the word \( \ell_{\pi} = a_1 a_2 \ldots a_n \). Similarly an infinite path is an infinite sequence \( \pi \)