Acta Informatica 19, 357–367 (1983)

On Some Variants of Post's Correspondence Problem

Keijo Ruohonen

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

Summary. A variant of Post's Correspondence Problem is considered where two different index words are allowed provided that one of them can be obtained from the other by permuting a fixed number of subwords. It is shown that this variant is undecidable. Post's Correspondence Problem is also extended to circular words, doubly infinite words and doubly infinite powers of words, and shown to be undecidable in all these extensions.

1. Introduction

Post's Correspondence Problem (PCP) is one of the most useful undecidable problems in the sense that it is easily stated and easily reduced to a number of interesting problems, especially problems concerning context-free languages.

PCP can be formulated as the problem of finding out, given a pair of morphisms $f, g: \Sigma_1^* \to \Sigma_2^*$, whether or not there exists a nonempty word $A \in \Sigma_1^*$ such that $f(A)=g(A)$. Such a formulation offers numerous possibilities for variation. Two goals for investigating variants of PCP are the following: (I) investigating the "borderland" between decidable and undecidable and (II) obtaining undecidable variants for which the reduction to certain problems is easier and more natural than that of PCP itself.

Concerning (I), obvious ways of searching for decidable variants of PCP are the following:

(i) bounding the cardinalities of Σ_1 and Σ_2; we know that decidability is obtained if $|\Sigma_1|=1$ or $|\Sigma_2|=1$ (trivial) and if $|\Sigma_1|=2$ (highly nontrivial, see [4] or [13]); undecidability is obtained for $|\Sigma_2|=2$ (trivial) and for $|\Sigma_1| \geqq 9$ (see [2] and [12]); the cases $|\Sigma_1|=3, 4, \ldots, 8$ are open;

(ii) restricting the structure of f and g, i.e., only morphisms of a restricted type are considered; in this case undecidability is obtained for morphisms of very simple structure; a modification of a proof of the undecidability of PCP shows that undecidability is obtained for 2-bounded morphisms f and g, i.e., morphisms f and g such that $|f(a)| \leqq 2$ and $|g(a)| \leqq 2$ for all $a \in \Sigma_1$; on the other

hand, undecidability is obtained also for morphisms with (uniformly) bounded synchronization delays both from left to right and from right to left (for the definition see e.g. [14], for the result see [10] or [5]), even in the case where the so-called equality set $\{A \mid f(A)=g(A)\}$ is known to be of the simplest possible type B^* for a word $B \in \Sigma_1^*$ (the question then, of course, is whether or not B is empty);

(iii) restricting the structure of A; not much seems to be known of such restrictions (see [3], however); undecidability holds for the so-called modified PCP where it is required that A begins (or ends, or both) with a preassigned symbol (see e.g. [7]);

(iv) replacing PCP by the problem of finding out whether or not $f(A)=g(B)$ for some nonempty words $A, B \in \Sigma_1^*$ such that $A \setminus B$ for a given binary relation $\setminus$ on Σ_1^*, preferably computable, reflexive and symmetric; decidability is obtained in the case where $\setminus$ is defined by

$$A \setminus B \quad \text{if and only if } |A|=|B|$$

(see [6]) and also in the case where $\setminus$ is the Parikh equivalence relation (see [8], a generalization of this decidability result was recently given in [9]).

Conceivably, these variants are of interest also from the point of view of the second goal.

In this paper we give an undecidable variant of PCP of type (iv) above. The binary relation $\setminus_n$ we use is defined as follows:

$$A \setminus_n B \quad \text{if and only if } A=A_1 A_2 \ldots A_n \text{ and } B=A_{\sigma(1)} A_{\sigma(2)} \cdots A_{\sigma(n)}$$

for some words $A_1, A_2, \ldots, A_n$ (possibly empty) over Σ_1 and an n-permutation σ.

Here n is an arbitrary but fixed positive integer. (See also [1].) The variant of PCP thus obtained is called the *n-permutation PCP* (n-PPCP). Obviously 1-PPCP is PCP and thus undecidable. The undecidability of n-PPCP for $n=2, 3, \ldots$ is less immediate.

Our proof techniques produce the undecidability of PCP for circular words, doubly infinite words and doubly infinite powers of words as an easy "by-product". We give the proofs of these undecidabilities here, not having found them in the literature. It might be mentioned that PCP is known to be undecidable for singly infinite words (ω-words), too, even in the case where the morphisms f and g have bounded synchronization delays in both directions and there exists at most one ω-word A such that $f(A)=g(A)$ (this is easily deduced from the constructions in [10]; [5] contains more details). This goes for singly infinite powers of words, too.

For readers not familiar with the basics of formal language theory we recommend [7] and [14].

2. Preliminaries

We use the notation and terminology of [7].

One of the classic exercises concerning linear bounded automata (LBA) is to show that they can be replaced by equivalent LBA which always halt, see e.g.

Problem 9.5.4 in [7]. The idea is to use a k-ary counter, for sufficiently large k, which, operating in linear space, counts the number of steps of the "original" LBA while this is simulated by the "new" LBA. Counter overflow means that an instantaneous description (ID) of the original LBA is repeated and the computation halts in a nonaccepting state. The replacement preserves determinism. A straightforward elaboration of this idea proves the following lemma:

Lemma 2.1. *For any deterministic LBA M_1 another deterministic LBA M_2 can be effectively constructed, such that M_2 has a nonempty recurrent ID if and only if the language recognized by M_1 contains a nonempty word. (An ID α of M_2 is recurrent if the computation of M_2 starting from α eventually returns to α. The empty ID, corresponding to the case where the tape cells contain only the blank symbol, is excluded.)* ∎

We leave the proof of this lemma for the reader. There is an immediate undecidability result implied by the lemma:

Lemma 2.2. *It is undecidable whether or not a deterministic LBA has nonempty recurrent ID's.*

Proof. The emptiness problem for deterministic context-sensitive languages is undecidable. ∎

The deterministic LBA (DLBA) we shall consider are assumed to have left and right endmarkers. In fact, for technical reasons, we use double endmarkers on both sides and assume that the head of the DLBA can scan only the "inner" endmarkers which are not written on. We assume further that the DLBA halts when and only when it reaches an accepting state. Obviously these assumptions do not affect the validity of Lemmata 2.1 and 2.2. (Note also that the lemmata do not follow directly from these assumptions and the well-known closure of deterministic context-sensitive languages under complementation because a recurrent ID may not belong to any computation starting with and initial ID.)

Let $f, g\colon \Sigma_1^* \to \Sigma_2^*$ be morphisms. An *(f, g)-sequence of length n* is a sequence $A_1, A_2, \ldots, A_{n+1}$ of words over Σ_1 such that

$$f(A_i) = g(A_{i+1}) \quad \text{for} \quad i = 1, 2, \ldots, n.$$

By an *n-permutation of a word A* we mean any word of the form $A_{\sigma(1)}A_{\sigma(2)}\ldots A_{\sigma(n)}$ where $A = A_1A_2\ldots A_n$ and σ is an n-permutation. The possibility of any of the subwords $A_1, A_2, \ldots, A_n$ being empty is not excluded. The binary relation $\curlywedge_n$ defined by

$$A \curlywedge_n B \text{ if and only if } A \text{ is } n\text{-permutation of } B$$

is reflexive and symmetric, but it is transitive only for $n = 1, 2$. The equivalence classes of $\curlywedge_2$ are called *circular words*. An equivalence class of $\curlywedge_2$ represented by A is denoted by A^{Ω}.

The *n-permutation PCP* (n-PPCP) is the problem of deciding of arbitrary morphisms f and g whether or not there exist nonempty words A and B such

that

$$f(A)=g(B) \quad \text{and} \quad A \curlywedge_n B.$$

A morphism h is said to be k-bounded if $|f(a)| \leqq k$ for each symbol a in the domain alphabet of h. Simultaneous bounding of the morphisms and the sizes of the domain alphabets makes PCP and any variant of it trivially decidable. Therefore, to obtain undecidabilities, at least one of these parameters must be unbounded. The usual choice in proofs of the undecidability of PCP seems to be to let the morphisms be unbounded, see e.g. [7]. In our opinion, the other choice is the more flexible one and we adopted it here. In fact, our morphisms here will be 9-bounded.

3. Simulation of DLBA by (*f*, *g*)-Sequences with Regular Control

Let $M=(Q, \Sigma, \Gamma, \delta, q_0, F)$ be a DLBA with left endmarkers $¢_1, ¢_2 \in \Gamma$ and right endmarkers $\$_1, \$_2 \in \Gamma$. (Since endmarkers are used the blank symbol is omitted.) A nonempty ID of M reading a_i in state q can then be written in the form

$$a_{-1} a_0 a_1 \ldots a_{i-1} (q, a_i)\, a_{i+1} \ldots a_n a_{n+1} a_{n+2}$$

where $a_1, \ldots, a_n \in \Gamma - \{¢_1, ¢_2, \$_1, \$_2\}$, $q \in Q$, $a_{-1}=¢_1$, $a_0=¢_2$, $a_{n+1}=\$_1$, $a_{n+2}=\$_2$, $n \geqq 1$ and $0 \leqq i \leqq n+1$.

We shall now construct two morphisms $f_1, g_1 \colon \Sigma_1^* \to \Sigma_2^*$ such that a sequence of consecutive nonempty ID's given by a computation of M can be considered as an (f_1, g_1)-sequence. Such constructs actually appear already in [10, 11]. First we define the alphabets Σ_1 and Σ_2:

$$\begin{aligned}\Sigma_1 = \Gamma \cup & ((\Gamma - \{¢_1, \$_1, \$_2\}) \times (Q-F) \times (\Gamma - \{¢_1, ¢_2, \$_1, \$_2\}) \times (\Gamma - \{¢_1, ¢_2, \$_2\})) \\ & \cup (\{¢_1\} \times (Q-F) \times \{¢_2\} \times (\Gamma - \{¢_1, ¢_2, \$_1, \$_2\})) \\ & \cup ((\Gamma - \{¢_1, ¢_2, \$_1, \$_2\}) \times (Q-F) \times \{\$_1\} \times \{\$_2\}), \\ \Sigma_2 = \Gamma \cup & (Q \times (\Gamma - \{¢_1, \$_2\})).\end{aligned}$$

Then we define f_1 and g_1 by

$$f_1(a)=g_1(a)=a \quad \text{for} \quad a \in \Gamma$$

and

$$f_1((a_1, q, a_2, a_3)) = \begin{cases} a_1 (q', a_2')\, a_3, & \text{if } \delta(q, a_2)=(q', a_2', 0) \\ (q', a_1)\, a_2'\, a_3, & \text{if } \delta(q, a_2)=(q', a_2', -1) \\ a_1\, a_2' (q', a_3), & \text{if } \delta(q, a_2)=(q', a_2', +1), \end{cases}$$

$$g_1((a_1, q, a_2, a_3)) = a_1 (q, a_2)\, a_3.$$

It is easy to verify that if

$$a_{-1} a_0 a_1 \ldots a_{i-1} (q, a_i)\, a_{i+1} \ldots a_n a_{n-1} a_{n+2}$$

and

$$b_{-1} b_0 b_1 \ldots b_{j-1} (q', b_j)\, b_{j+1} \ldots b_n b_{n+1} b_{n+2}$$

are consecutive ID's of M and $q' \notin F$, then

$$f_1(a_{-1}a_0a_1 \dots (a_{i-1}, q, a_i, a_{i+1}) \dots a_n a_{n+1} a_{n+2})$$
$$= g_1(b_{-1}b_0b_1 \dots (b_{j-1}, q', b_j, b_{j+1}) \dots b_n b_{n+1} b_{n+2}).$$

Hence computations of M can be simulated by (f_1, g_1)-sequences. Note especially that a computational step of M leading to a halting state (i.e., a state in F) can be simulated only "half way" because the final application of g_1^{-1} produces the empty set. In the sequel we are interested only in simulating computations with recurring ID's and they cannot contain states in F, so no special measures are needed.

Before continuing our construction we note the following. This paper might be considered as an exercise in how to avoid the troublesome effects of limited random rearrangements (n-permutations) of the memory space of DLBA. The straightforward solution here is to take sufficiently many - $\ell \geqq n^n$, that is - "copies" of the computations and advance them in parallel using a common memory space divided in sections, one for each copy of computation. Due to the limitedness of the rearrangements, we know that at least one computation finishes "intact" for a sufficiently large ℓ.

For this purpose we take 2ℓ replicas of the alphabet Γ. The word obtained from $\alpha \in \Gamma^*$ by replacing symbols by their i^{th} replicas is denoted by $\alpha^{(i)}$. Now, let

$$\alpha_1 a_1(q_1, b_1) c_1 \beta_1, \alpha_2 a_2(q_2, b_2) c_2 \beta_2, \dots, \alpha_\ell a_\ell(q_\ell, b_\ell) c_\ell \beta_\ell$$

be nonempty ID's of M with $\alpha_1, \alpha_2, \dots, \alpha_\ell, \beta_1, \beta_2, \dots, \beta_\ell \in \Gamma^*$;

$$a_1, a_2, \dots, a_\ell, b_1, b_2, \dots, b_\ell, c_1, c_2, \dots, c_\ell \in \Gamma \quad \text{and} \quad q_1, q_2, \dots, q_\ell \in Q - F.$$

Consider the word

$$\alpha_1^{(1)}(a_1^{(1)}, q_1, b_1^{(2)}, c_1^{(2)}) \beta_1^{(2)} \alpha_2^{(3)}(a_2^{(3)}, q_2, b_2^{(4)}, c_2^{(4)}) \beta_2^{(4)}$$
$$\dots \alpha_\ell^{(2\ell-1)}(a_\ell^{(2\ell-1)}, q_\ell, b_\ell^{(2\ell)}, c_\ell^{(2\ell)}) \beta_\ell^{(2\ell)}. \tag{1}$$

The set of all words of this form is a regular set, denoted by R_1. For a word A of type (1), we call the subword corresponding to the i^{th} ID of M, i.e. $\alpha_i^{(2i-1)}(a_i^{(2i-1)}, q_i, b_i^{(2i)}, c_i^{(2i)}, \beta_i^{(2i)}$, the i^{th} *segment* of A and i is called the *index of the segment*. Similarly, we have the word

$$\alpha_1^{(1)} a_1^{(1)}(q_1, b_1^{(2)}) c_1^{(2)} \beta_1^{(2)} \alpha_2^{(3)} a_2^{(3)}(q_2, b_2^{(4)}) c_2^{(4)} \beta_2^{(4)}$$
$$\dots \alpha_\ell^{(2\ell-1)} a_\ell^{(2\ell-1)}(q_\ell, b_\ell^{(2\ell)}) c_\ell^{(2\ell)} \beta_\ell^{(2\ell)}.$$

The set of all words of this latter form is also a regular set, denoted by R_2. Let the set of all symbols appearing in words of R_1 (resp. R_2) be $\Sigma_{1\ell}$ (resp. $\Sigma_{2\ell}$).

Note that not only have we catenated ℓ ID's of M coded in as many copies of Σ_1 (and Σ_2) but, in each such ID, we have also separated symbols occurring to the left of the symbol under scan from those to the right of the scanned symbol by letting the former have odd indices and the latter even indices. (The symbol under scan has an even index.) This technical feature is needed for the regular control to be added in our construction.

We can represent ℓ (separate) computational steps of M simultaneously, in an obvious fashion, by giving two words of R_1, the i^{th} segments corresponding to the i^{th} computational step for $1 \leqq i \leqq \ell$. It is straightforward to add indices to the definitions of f_1 and g_1 so as to obtain a simulation of these ℓ computational steps. Let us denote the two morphisms $\Sigma_{1\ell}^* \to \Sigma_{2\ell}^*$ by f_2 and g_2. We have, for each $i \in \{1, \ldots, \ell\}$,

$$f_2(a^{(j)}) = g_2(a^{(j)}) = a^{(j)} \quad \text{for } a \in \Gamma \text{ and } j \in \{2i-1, 2i\},$$

$$f_2((a_1^{(2i-1)}, q, a_2^{(2i)}, a_3^{(2i)})) = \begin{cases} a_1^{(2i-1)}(q', b^{(2i)})\, a_3^{(2i)}, & \text{if } \delta(q, a_2) = (q', b, 0) \\ (q', a_1^{(2i)})\, b^{(2i)}\, a_3^{(2i)}, & \text{if } \delta(q, a_2) = (q', b, -1) \\ a_1^{(2i-1)}\, b^{(2i-1)}(q', a_3^{(2i)}), & \text{if } \delta(q, a_2) = (q', b, +1) \end{cases}$$

and

$$g_2((a_1^{(2i-1)}, q, a_2^{(2i)}, a_3^{(2i)})) = a_1^{(2i-1)}(q, a_2^{(2i)})\, a_3^{(2i)}.$$

The following lemma is then easily proved.

Lemma 3.1. *Let P and Q be words over $\Sigma_{1\ell}$ such that $f_2(P) = g_2(Q) \in R_2$. Then $P, Q \in R_1$ and the pair P, Q represents ℓ (separate) computational steps of M in the way described above.* ∎

There are, of course, other (f_2, g_2)-sequences than those obtained by termwise catenation of sequences simulating computations of M. To remedy this situation more structure is added in the morphisms f_2 and g_2. Let us first consider the situation in a somewhat more general setting.

We say that a deterministic finite automaton $G = (S, \Delta, \delta, s_0, H)$ is a *unique state automaton* if

(i) $s_0 \notin \delta(S, \Delta)$, $\quad \delta(s_0, \Delta) \cap H = \Phi$,

(ii) $\delta(H, \Delta) = \Phi$, $\quad \delta(s_0, \Delta) \neq \Phi$ and

(iii) there is a mapping $i\colon \Delta \to S - H$ such that $\delta(s, a)$ is defined if and only if $s = i(a)$.

Lemma 3.2. *For any unique state automaton G there exist effectively constructable morphisms $f, g\colon \Delta^* \to (\Delta \cup S)^*$ such that a nonempty word C is accepted by G only if it begins an (f, g)-sequence. Moreover, the Kleenean closure of the language recognized by G equals the set of first terms of (f, g)-sequences and $g^{-1}f$ and $f^{-1}g$ are identities on this set.*

Proof. Define f and g by

$$f(a) = \begin{cases} a\, \delta(s_0, a), & \text{if } i(a) = s_0 \\ i(a)\, a\, \delta(i(a), a), & \text{if } i(a) \neq s_0 \text{ and } \delta(i(a), a) \notin H \\ i(a)\, a, & \text{if } \delta(i(a), a) \in H \end{cases}$$

and

$$g(a) = \begin{cases} a, & \text{if } i(a) = s_0 \\ i(a)\, i(a)\, a, & \text{if } i(a) \neq s_0. \end{cases}$$

Then, if the word $a_1 a_2 \ldots a_n$ where $a_1, a_2, \ldots, a_n \in \Delta$ is accepted by G and $\delta(s_0, a_1 a_2 \ldots a_i) = s_i$, we have

$$f(a_1 a_2 \ldots a_n) = a_1 s_1 s_1 a_2 s_2 s_2 a_3 \ldots a_{n-1} s_{n-1} s_{n-1} a_n = g(a_1 a_2 \ldots a_n).$$

Thus words of the Kleenean closure of the language recognized by G start constant (f, g)-sequences and clearly this is the only way (f, g)-sequences can be formed. ∎

Thus a regular control can be realized by (f, g)-sequences provided that the control is no stronger than one given by a unique state automaton. This is sufficient because we have

Lemma 3.3. *The language R_2 can be recognized by a unique state automaton G'.*

Proof. A straightforward but somewhat tedious construction produces a unique state automaton, with at most $5\ell+1$ states, recognizing R_2. Details are left to the reader. ∎

By Lemmata 3.2 and 3.3 we have two morphisms f' and g', say, "recognizing" R_2^* through (f', g')-sequences. Define then

$$f_3 = f' f_2 \quad \text{and} \quad g_3 = g' g_2.$$

Making these compositions adds sufficient regular control in our simulation and we have

Lemma 3.4. *Let P and Q be nonempty words over $\Sigma_{1\ell}$ such that $f_3(P) = g_3(Q)$. Then there exist a natural number $k \geqq 1$ and words $P_1, \ldots, P_k, Q_1, \ldots, Q_k \in R_1$ such that $P = P_1 \ldots P_k$, $Q = Q_1 \ldots Q_k$ and $f_2(P_j) = g_2(Q_j) \in R_2$ for $j = 1, \ldots, k$.*

Proof. Since $f'(f_2(P)) = g'(g_2(Q))$, we have, by the choice of f' and g' and Lemma 3.2, $f_2(P) = g_2(Q) \in R_2^+$, say $f_2(P) = T_1 \ldots T_k$ where $T_1, \ldots, T_k \in R_2$. Since we have

$$f_2^{-1}(T_1 \ldots T_k) = f_2^{-1}(T_1) \ldots f_2^{-1}(T_k) \quad \text{and} \quad g_2^{-1}(T_1 \ldots T_k) = g_2^{-1}(T_1) \ldots g_2^{-1}(T_k)$$

(g_2 is, in fact, injective), the lemma follows by Lemma 3.1. ∎

4. The Main Result

We shall now prove that n-PPCP is undecidable for morphisms f_3 and g_3 of the type considered in the previous section, provided that the value of ℓ is at least n^n. Our construction does not cover the case $n = 1$, which is well-known anyway, so we shall assume that $n \geqq 2$.

Suppose that for some nonempty words P and Q over $\Sigma_{1\ell}$ we have

$$f_3(P) = g_3(Q) \quad \text{and} \quad P \not\sim_n Q.$$

We will show that the DLBA M has then a recurring (nonempty) ID.

By Lemma 3.4 we have $P = P_1 \ldots P_k$ and $Q = Q_1 \ldots Q_k$ where $k \geqq 1$ and $P_1, \ldots, P_k, Q_1, \ldots, Q_k \in R_1$ and furthermore

$$f_2(P_j) = g_2(Q_j) \in R_2 \quad \text{for } j = 1, \ldots, k$$

whence also $f_3(P_j) = g_3(Q_j)$ for $j = 1, \ldots, k$.

On the other hand, since $P \curlywedge_n Q$, we can write $P = A_1 \dots A_n$ and $Q = A_{\sigma(1)} \dots A_{\sigma(n)}$ for certain words $A_1, \dots, A_n$ over $\Sigma_{1\ell}$ and an n-permutation σ. There may be several possible choices for $A_1, \dots, A_n$ and σ. We choose one of them and consider it fixed.

Thus we get the words $A_1, \dots, A_n$ by making $n-1$ "cuts" in P. We say that the subword P_j of P is cut if at least one of these cuts takes place inside P_j. (Thus a cut occurring exactly between P_j and one of its neighbors is not considered as a cut of P_j.) Denote $J = \{j | P_j \text{ is cut}\}$. Similarly we get the words $A_{\sigma(1)}, \dots, A_{\sigma(n)}$ by making $n-1$ cuts in Q and we denote $J' = \{j | Q_j \text{ is cut}\}$.

We then reason as follows. Let m be an index in $\{1, \dots, k\}$. If $m \notin J'$, that is, if Q_m is not cut, then Q_m is one of the words $P_1, \dots, P_k$, say $Q_m = P_{j_{m1}}$, and $j_{m1} \notin J$. If now $j_{m1} \notin J'$, then, again, $Q_{j_{m1}}$ is one of the words $P_1, \dots, P_k$, say $Q_{j_{m1}} = P_{j_{m2}}$, and $j_{m2} \notin J$. We have two cases: either we can continue in this way indefinitely, or we end up eventually with an index $j_{m,t(m)}$, say, in J' (especially, if $m \in J'$, then $t(m) = 0$ and $j_{m,t(m)} = m$). In the former case there certainly exists a recurring (nonempty) ID of M, because we have $|Q_m| = |Q_{j_{m1}}| = |Q_{j_{m2}}| = \dots$. Let us then assume that, for all $m \in \{1, \dots, k\}$, the latter case is valid. In this case we can write

$$Q_{j_{m,t(m)}} = T_{m1} \dots T_{m,s(m)}$$

where $T_{m1}, \dots, T_{m,s(m)}$ are nonempty proper subwords of words in $\{P_j | j \in J\}$. Note that $2 \leqq s(m) \leqq n$ for $m = 1, \dots, k$.

Now, at least one of the words $T_1, \dots, T_{1,s(1)}$ contains at least $\ell / s(1) \geqq \ell / n \geqq n^{n-1}$ whole consecutive segments by Dirichlet's box principle, say segments with indices forming the set I. Let $T_{1,u(1)}$ be such a word, a subword of $P_{v(1)}$, say, where $v(1) \in J$. Then, by the same token, one of the words $T_{v(1),1}, \dots, T_{v(1),s(v(1))}$ contains at least n^{n-2} whole consecutive segments with indices coming from the set I, say the word $T_{v(1),u(2)}$, a subword of $P_{v(2)}$, say, where $v(2) \in J$.

Continuing in this way for n steps we finally find that a word $T_{v(n-1),u(n)}$, a subword of $P_{v(n)}$, where $v(n) \in J$, contains at least one whole segment with indice in I. Since $|J| \leqq n-1$, at least one ID of M occurs twice in the sequence $T_{1,u(1)}, \dots, T_{v(n-1),u(n)}$ and is thus recurring. Note that each such occurrence of the ID, properly coded as a segment of a word of R_1, must have the same index.

So, existence of nonempty words P and Q such that $f_3(P) = g_3(Q)$ implies that M has a recurring (nonempty) ID. Suppose then, conversely, that M has a recurrent nonempty ID γ_1, i.e., that a sequence

$$\gamma_1, \gamma_2, \dots, \gamma_p, \gamma_1$$

is a sequence of consecutive ID's of M. Taking replicas of each of these ℓ times and a proper coding produces $p+1$ words of R_1: $B_1, B_2, \dots, B_p, B_1$. Then

$$f_3(B_1 B_2 \dots B_p) = g_3(B_2 B_3 \dots B_p B_1)$$

and clearly $B_1 B_2 \dots B_p \curlywedge_n B_2 B_3 \dots B_p B_1$ for any $n \geqq 2$.

We have proved, by Lemma 2.2,

Theorem 4.1. *For any fixed $n \geqq 1$, n-PPCP is undecidable.* ∎

5. PCP for Circular Words and Doubly Infinite words

We recall that *circular words* are equivalence classes of the relation $\curlywedge_2$. Any morphism f induces a mapping f^Ω of circular words into circular words:

$$f^\Omega(A^\Omega)=(f(A))^\Omega.$$

Note that f^Ω is well-defined. Circular words may be visualized as circular strings or "necklaces" of symbols.

We may thus formulate *PCP for circular words*, using the mappings induced by morphisms, as the problem of deciding of arbitrary morphisms f and g whether or not there exists a nonempty circular word A^Ω such that $f^\Omega(A^\Omega)=g^\Omega(A^\Omega)$.

Considering morphisms of the type of f_3 and g_3 of Sect. 3, as was done for 2-PPCP, does not prove PCP for circular words undecidable. In fact, there frequently exists a nonempty word A such that $(f_3(A))^\Omega=(g_3(A))^\Omega$ with no connection whatsoever to computations of the DLBA M.

Thus an additional construct is needed. We take as our starting point the result of Sects. 2 and 3 that it is undecidable of arbitrary morphisms $f,g\colon \Sigma_1^*\to\Sigma_2^*$ whether or not there exists a nonempty word A starting and ending an (f,g)-sequence. This follows immediately from Lemmata 2.2, 3.1 and 3.4 (here we may take $\ell=1$). We take two replicas of Σ_1 and Σ_2, and define the morphisms f_4 and g_4 on $(\Sigma_1^{(1)}\cup\Sigma_1^{(2)})^*$ by

$$f_4(a^{(1)})=(f(a))^{(2)},\quad f_4(a^{(2)})=(f(a))^{(1)},$$
$$g_4(a^{(1)})=(g(a))^{(1)},\quad g_4(a^{(2)})=(g(a))^{(2)}.$$

It is immediate that

$$A_0,A_1,\ldots,A_{2n+1},A_0 \tag{2}$$

is an (f,g)-sequence only if

$$f_4^\Omega((A_0^{(1)}A_1^{(2)}A_2^{(1)}\ldots A_{2n+1}^{(2)})^\Omega)=g_4^\Omega((A_0^{(1)}A_1^{(2)}A_2^{(1)}\ldots A_{2n+1}^{(2)})^\Omega). \tag{3}$$

On the other hand, if (3) holds true, then a subsequence of the sequence obtained by repeating (2) $2n+3$ times is a recurring (f,g)-sequence. Note that any nonempty circular word A^Ω for which $f_4^\Omega(A^\Omega)=g_4^\Omega(A^\Omega)$ necessarily contains symbols of both $\Sigma_1^{(1)}$ and $\Sigma_1^{(2)}$ and has a representative beginning with symbols of $\Sigma_1^{(1)}$ and ending with symbols of $\Sigma_1^{(2)}$, and that we may restrict ourselves to consider only recurring (f,g)-sequences of even length by repeating a sequence twice if needed.

We have thus

Theorem 5.1. *PCP for circular words is undecidable.* ∎

The same construction shows PCP to be undecidable for doubly infinite words ($\omega^*+\omega$-words) and doubly infinite powers of words ($\omega^*+\omega$-powers of words). A doubly infinite word over Σ is an equivalence class of the binary relation $\curlywedge_\infty$ on the set of mappings $\phi\colon\mathbb{Z}\to\Sigma$ ($\mathbb{Z}$ denotes the set of integers) defined by

$$\phi\curlywedge_\infty\psi \quad \text{if and only if there exists a } j\in\mathbb{Z} \text{ such that } \phi(n)=\psi(n+j)$$
$$\text{for all } n\in\mathbb{Z}.$$

The equivalence class represented by ϕ is denoted by ϕ^∞ or by

$$\ldots \phi(-2)\,\phi(-1)\,\phi(0)\,\phi(1)\,\phi(2)\ldots.$$

A morphism f on Σ^* induces a mapping f^∞ of doubly infinite words into doubly infinite words as follows:

$$f^\infty(\phi^\infty)=\ldots f(\phi(-2))f(\phi(-1))f(\phi(0))f(\phi(1))f(\phi(2))\ldots$$

(assuming, of course, that ϕ^∞ contains infinitely many occurrences of symbols not erased by f).

Thus we can define *PCP for doubly infinite words* as the problem of deciding of arbitrary morphisms f and g whether or not there exists a doubly infinite word ϕ^∞ such that $f^\infty(\phi^\infty)=g^\infty(\phi^\infty)$.

A doubly infinite word ϕ^∞ where ϕ is periodic is called a *doubly infinite power* of the periodic part. If the periodic part is A we can then write

$$\phi^\infty=\ldots AAA\ldots=A^{\omega^*+\omega}.$$

We can define doubly infinite powers of words alternatively as equivalence classes of the following binary relation σ:

$A\sigma B$ if and only if $A^n \curlywedge_2 B^m$ for some positive integers n and m.

Since the image of a doubly infinite power under f^∞ is again a doubly infinite power, i.e.,

$$f^\infty(A^{\omega^*+\omega})=(f(A))^{\omega^*+\omega},$$

we can define *PCP for doubly infinite powers of words* in the natural way.

From Sects. 2 and 3 we know that it is undecidable of arbitrary nonerasing morphisms $f,g\colon \Sigma_1^*\to\Sigma_2^*$, such that no (f,g)-sequence contains words of differing length, whether or not there exists a nonempty word A starting and ending an (f,g)-sequence. Since, for such f and g, there exists a doubly infinite word ϕ^∞ such that $f_4^\infty(\phi^\infty)=g_4^\infty(\phi^\infty)$ if and only if there is a recurring (f,g)-sequence containing only nonempty words, we have

Theorem 5.2. *PCP for doubly infinite words is undecidable.* ∎

On the other hand, existence of a ϕ^∞ such that $f_4^\infty(\phi^\infty)=g_4^\infty(\phi^\infty)$ implies the existence of a doubly infinite power ψ^∞ of a word obtained by catenating finitely many subwords of ϕ^∞, such that $f_4^\infty(\psi^\infty)=g_4^\infty(\psi^\infty)$. Hence we have also

Theorem 5.3. *PCP for doubly infinite powers of words is undecidable.* ∎

In the context of PCP it is much more natural to define doubly infinite words as generalizations (to infinite length) of circular words than of "ordinary" words, i.e., to use equivalence classes of mappings under shift rather than the mappings as such. For the latter, it is not even clear how to define PCP without getting two instances of PCP for singly infinite words.

Remark. A generalization of PCP for circular words is the following which might be called the (m,n)-permutation PCP ((m,n)-PPCP): decide of arbitrary

morphisms f and g whether or not there exist nonempty words A and B such that $A \curlywedge_m B$ and $f(A) \curlywedge_n g(B)$. Obviously (2,2)-PPCP is PCP for circular words and $(m, 1)$-PPCP is m-PPCP.

It is tempting to conjecture that (m, n)-PPCP is undecidable for all m and n. A proof of this along the lines of this paper would seem to require much more elaborate versions of the constructions in Sect. 3, and is pursued no further here.

Acknowledgement. The author wishes to thank the referees for their useful suggestions.

References

1. Brandstadt, A.: Closure properties of certain families of formal languages with respect to a generalization of cyclic closure. R.A.I.R.O. Informatique théorique **15**, 233–252 (1981)
2. Claus, V.: Die Grenze zwischen Entscheidbarkeit und Nichtentscheidbarkeit. Fernstudienkurs für die Fernuniversität Hagen. Open University of Hagen, Hagen 1979
3. Čulik II, K.: Homomorphisms: Decidability, equality and test sets. In: Formal Language Theory. Perspectives and Open Problems. Book, R.V. (ed.), pp. 167–194. New York: Academic Press 1980
4. Ehrenfeucht, A., Karhumäki, J., Rozenberg, G.: The (generalized) Post correspondence problem with lists consisting of two words is decidable. Theoret. Comput. Sci. **21**, 119–144 (1982)
5. Ehrenfeucht, A., Rozenberg, G., Ruohonen, K.: Structurally restricted maximal solutions of language equations involving morphisms. Report **42**. Tampere University of Technology, Department of Electrical Engineering, Mathematics, Tampere 1983
6. Greibach, S.: A remark on code sets and context-free languages. IEEE Trans. Comput. **C-24**, 741–742 (1975)
7. Harrison, M.: Introduction to Formal Language Theory. Reading, MA: Addison-Wesley 1978
8. Ibarra, O., Kim, C.: A useful device for showing the solvability of some decision problems. Proc. 8th Ann. Symp. on Theory of Computing, pp. 135–140, 1976
9. Karhumäki, J.: Generalized Parikh mappings and homomorphisms. Information Control **47**, 155–165 (1980)
10. Lecerf, Y.: Récursive insolubilité de l'équation générale de diagonalisation de deux monomorphismes de monoïdes libres $\phi x = \psi x$. Comptes Rendus **257**, 2940–2943 (1963)
11. Lecerf, Y.: Machines de Turing réversibles. Récursive insolubilité en $n \in N$ de l'équation $u = \theta^n u$, où θ est un "isomorphisme de codes". Ibid., pp. 2597–2600, 1963
12. Pansiot, J.J.: A note on Post's Correspondence Problem. Information Processing Lett. **12**, 233 (1981)
13. Pavlenko, V.A.: The combinatorial problem of Post with two word pairs (In Russian). Dokl. Akad. Nauk Ukr. SSR, Ser. A, **7**, 9–11 (1981)
14. Salomaa, A.: Jewels of Formal Language Theory. Potomac: Computer Science Press 1981

Received February 8, 1982/January 17, 1983