Available online at www.sciencedirect.com
ScienceDirect
Information Processing Letters 106 (2008) 115–119
Information Processing Letters
www.elsevier.com/locate/ipl

More decidable instances of Post's correspondence problem: beyond counting

Mirko Rahn

Universität Karlsruhe (TH), IAKS Vollmar, D-76128 Karlsruhe, Germany

Received 5 February 2007; received in revised form 12 October 2007; accepted 5 November 2007

Available online 12 November 2007

Communicated by F. Meyer auf der Heide

Abstract

We present a new technique to decide or reduce instances of Post's correspondence problem. It generalizes balance arguments to a consideration about some special context-free languages which allow the combination with other decision (or reduction) techniques. In spite of its simplicity, the new technique is able to decide more instances than known techniques.

Keywords: Post correspondence problem; Context-free language; Decidability; Combinatorial problems

1. Introduction

Let Σ, Δ be two finite alphabets and h, g two morphisms $h, g : \Sigma^* \to \Delta^*$. The *Post correspondence problem* (PCP) is to determine whether the equality set $E(h, g) = \{u \in \Sigma^* \mid h(u) = g(u)\}$ of h and g contains more than the empty word ε. In 1946 Emil L. Post has showed in [9] that the problem is undecidable in general. Since then, PCP serves as a common reference for undecidability proofs that is easily reduced to a number of interesting problems.

Some classes of decidable instances are known, namely binary instances [1,3], instances with periodic morphisms [1,8], instances with marked morphisms [5,3], and the so-called unique block instances [4] which form the largest class known so far. To learn more about the border line between decidability and undecidability, it is important to identify such classes.

In this paper, we present a new technique that allows to decide more instances of PCP. In a way, it is similar to the technique that is used to decide periodic instances: Just calculate a superset $S \supseteq E(h, g)$ of solutions, including as much information as possible, and ensure that S belongs to a class of languages for which finiteness is decidable. If S is finite, then the instance is unsolvable. Indeed, $E(h, g)$ is either $\{\varepsilon\}$ or infinite. As for periodic instances, the intersection of a regular superset of solutions with the kernels of some morphisms (which reflect word lengths) is used. These kernels can be imagined as generalized balances, and with their help we open symbol counting for enrichment with non-counting information.

It should be mentioned that similar techniques have been used to attack a related problem, namely to decide whether or not the set $P_k(h, g) = \{u \mid \exists v\colon\ u \equiv_k v \wedge h(u) = g(v)\}$ is empty. Here $u \equiv_k v$ is the case if u and v have the same number of occurrences of segments of length k. See [7] for the details. The relation to the

E-mail address: rahn@ira.uka.de.

0020-0190/$ – see front matter
doi:10.1016/j.ipl.2007.11.002

PCP is given by the equation $E(h,g) = \bigcap_{k \geqslant 0} P_k(h,g)$. This problem is decidable for non-erasing morphisms, as shown by Greibach in [2] for the case $k = 0$ (that is u and v have the same length), by Ibarra and Kim in [6] for the case $k = 1$ (that is u is a permutation of v) and by Karhumäki in [7] for $k \geqslant 2$. The proofs utilize some decidability properties of simple k-head pushdown automata, namely the fact that emptiness is decidable for them.

However, in this paper we are not going to modify the problem, but we will show how to use some basic results from language theory in a new way to decide some instances of PCP. In spite of its simplicity, the new technique is able to decide instances which cannot be decided by any of the known techniques. Some of the example instances in this paper are of this kind: They are neither binary, nor periodic, nor marked, nor unique block instances. The examples are chosen as instructive as possible and as small as possible at the same time. Nevertheless, the calculations are performed with the help of computer programs and may not be obvious at first glance.

We shall fix some notation. If $h : A \to B$ is a morphism, and $A' \subseteq A$ and $B' \subseteq B$ are subsets of A or B, respectively, then we denote by $h(A') \subseteq B$ the set $h(A') = \{h(x) \mid x \in A'\}$ and by $h^{-1}(B') \subseteq A$ the set $h^{-1}(B') = \{x \mid h(x) \in B'\}$. For alphabets $A = \{a_0, \ldots, a_{k-1}\}$ the *Parikh mapping* $\psi_A : A^* \to \mathbb{N}^k$ is given by $\psi_A(w) = (|w|_{a_0}, \ldots, |w|_{a_{k-1}})$. Here $|w|_a$ denotes the number of occurrences of the symbol a in the word w. For vectors $x, y \in \mathbb{Z}^m$ the standard scalar product $\sum_{i=0}^{m-1} x_i y_i$ is denoted by $\langle x|y\rangle$. For languages $L \subseteq A^*$ we denote by $S_A(L)$ the smallest alphabet $B \subseteq A$ where $L \subseteq B^*$. If $L \subseteq A^*$ is context-free, then the set $S_A(L)$ is computable, since $L \subseteq B^*$ if and only if $L \cap (A^* \setminus B^*) = \emptyset$. Indeed, context-free languages are closed under intersection with regular languages, and emptiness is decidable for them.

2. Generalized balances

We are now going to define infinitely many context-free languages, each of which is a superset of the equality set of two given morphisms. These languages are kernels of some symbol counting morphisms. The idea behind is simple: For each element u in the equality set, we have that the Parikh vectors of $h(u)$ and $g(u)$ are equal, and if one builds the scalar product with the same vector, the results are equal as well.

Let $h, g : \Sigma^* \to \Delta^*$ be an instance of PCP with $|\Delta| = n$. Define for a vector $v \in \mathbb{Z}^n$ the mapping $\varrho_v : \Sigma^* \to \mathbb{Z}$ by $\varrho_v(u) = \langle v|\psi_\Delta(h(u))\rangle - \langle v|\psi_\Delta(g(u))\rangle$. It is easy to see that $\varrho_v(\varepsilon) = 0$ and $\varrho_v(xy) = \varrho_v(x) + \varrho_v(y)$. Thus, ϱ_v is a monoid morphism from Σ^* into the additive group of integers and we have $\varrho_v(u) = \sum_{\sigma \in \Sigma} \varrho_v(\sigma)|u|_\sigma$. This definition for ϱ_v covers the most interesting cases, such as the calculation of length difference or the calculation of the difference of the number of occurrences of some specific symbol.

The equality set is a subset of the context-free kernels of these mappings:

Lemma 1. $E(h,g) \subseteq \bigcap_{v \in \mathbb{Z}^n} \varrho_v^{-1}(0)$.

Proof. Assume $u \notin \varrho_v^{-1}(0)$ for some $v \in \mathbb{Z}^n$. Then we have $\langle v|\psi_\Delta(h(u))\rangle \neq \langle v|\psi_\Delta(g(u))\rangle$ which implies $\psi_\Delta(h(u)) \neq \psi_\Delta(g(u))$ and $h(u) \neq g(u)$. □

Lemma 2. *The language $\varrho_v^{-1}(0)$ is context-free for arbitrary $v \in \mathbb{Z}^n$.*

Proof. We use the equality

$$\varrho_v^{-1}(0) = \left\{ u \mid \sum_{\sigma \in \Sigma} \varrho_v(\sigma)|u|_\sigma = 0 \right\}.$$

It is easy to construct a pushdown automaton that accepts the language on the right-hand side: The PDA uses its stack as an unary counter, and whenever the symbol σ is read from the input, the value $\varrho_v(\sigma)$ is added to this counter. A word is accepted, if and only if the stack counter contains the value 0 after the complete word was read. □

Furthermore, the subset relation holds even after calculating the smallest alphabets on both sides:

Lemma 3. $S_\Sigma(E(h,g)) \subseteq \bigcap_{v \in \mathbb{Z}^n} S_\Sigma(\varrho_v^{-1}(0))$.

Proof. Assume $\sigma \notin \bigcap_{v \in \mathbb{Z}^n} S_\Sigma(\varrho_v^{-1}(0))$. Then there is a $z \in \mathbb{Z}^n$ with $\sigma \notin S_\Sigma(\varrho_z^{-1}(0))$. Therefore, no word in $\varrho_z^{-1}(0)$ contains the symbol σ. By Lemma 1, no word in $E(h,g)$ contains the symbol σ. □

For context-free languages finiteness is decidable, and if there is a vector v where $\varrho_v^{-1}(0)$ is finite, one can already conclude that $E(h,g) = \{\varepsilon\}$. Unfortunately, context-free languages are not closed under intersection, so Lemma 1 cannot be used directly. But Lemma 3 helps: One is able to calculate the smallest alphabets for many vectors and to use their intersection as the largest alphabet for possible solutions.

Lemma 3 already generalizes the so-called *length balance* and *element balance*:

Example 4. Let $\Sigma = \{a, b, c\}$, $\Delta = \{0, 1\}$ and h, g be:

	a	b	c
h	00	01	10
g	001	00	101
$\varrho_{(1,1)}$	-1	0	-1

All images of g are at least as long as the images of h. Therefore, one can delete all rules that have strictly longer g-images. In terms of generalized balances, one has $\varrho_{(1,1)}^{-1}(0) = \{u \mid |u|_a + |u|_c = 0\}$. This implies $|u|_a = |u|_c = 0$ and $S_\Sigma(\varrho_{(1,1)}^{-1}(0)) = \{b\}$. The rules a and c cannot occur in any solution, hence, one can reduce the instance to rule b only.

Example 5. Let $\Sigma = \{a, b, c\}$, $\Delta = \{0, 1\}$ and h, g be:

	a	b	c
h	0	0	001
g	00	001	1
$\varrho_{(0,1)}$	0	-1	0

All images of g contain at least as many symbols 1 as the images of h. Therefore, one can delete all rules that have strictly more symbols 1 in their g-image. In terms of generalized balances, one has $\varrho_{(0,1)}^{-1}(0) = \{u \mid |u|_b = 0\}$ and $S_\Sigma(\varrho_{(0,1)}^{-1}(0)) = \{a, c\}$. The rule b cannot occur in any solution, hence, one can reduce the instance to rules a and c only. Indeed, we have $E(h, g) = (aac)^*$.

To open counting techniques for non-counting information, we state a simple corollary of Lemmas 1 and 3:

Corollary 6. *For $R \supseteq E(h, g)$ one has*

$$E(h, g) \subseteq \bigcap_{v \in \mathbb{Z}^n} \varrho_v^{-1}(0) \cap R \quad \text{and}$$

$$S_\Sigma\big(E(h, g)\big) \subseteq \bigcap_{v \in \mathbb{Z}^n} S_\Sigma\big(\varrho_v^{-1}(0) \cap R\big).$$

The ϱ-languages do not just rediscover balances, but really generalize them, as shown in the next example:

Example 7. Let $\Sigma = \{a, b, c\}$, $\Delta = \{0, 1\}$ and h, g be:

	a	b	c
h	0	00	011
g	000	101	00
$\varrho_{(2,1)}$	-4	0	0

The instance is well balanced, but we have $S_\Sigma(\varrho_{(2,1)}^{-1}(0)) = \{b, c\}$. Hence, rule a cannot occur in any solution.

To get the last result, one can use an algebraic approach. If $\Sigma = \{\sigma_0, \ldots, \sigma_{s-1}\}$, $|\Delta| = n$, and $h, g : \Sigma^* \to \Delta^*$, then we define $D(h, g)$ as the $(n \times s)$-matrix with $D(h, g)_{[i,j]} = \varrho_{|i\rangle}(\sigma_j)$. Here $|i\rangle$ denotes the unit vector $(0, \ldots, 0, 1, 0, \ldots, 0)$ of length n with the 1 at position i. We denote by $\mathsf{nsp}(D(h, g))$ the null-space of $D(h, g)$, that is the set $\{x \in \mathbb{Q}^s \mid D(h, g)x^\top = \mathcal{O}\}$. Then we have $\psi_\Sigma(E(h, g)) \subseteq \mathsf{nsp}(D(h, g))$. In Example 7 one has $D(h, g) = \left(\begin{smallmatrix} -2 & 1 & -1 \\ 0 & -2 & 2 \end{smallmatrix}\right)$ and $\mathsf{nsp}(D(h, g))$ has the basis $\{(0, 1, 1)\}$. Thus, for each $u \in E(h, g)$ there is some $\lambda \in \mathbb{N}$, such that $\psi_\Sigma(u) = \lambda(0, 1, 1)$. This clearly implies $|u|_a = 0$.

In a way, the approach via the equation system is better, since there is no need to guess the linear combination $\varrho_{(2,1)}(u) = 2\varrho_{(1,0)}(u) + \varrho_{(0,1)}(u)$. On the other hand, the ϱ-languages allow to include non-counting information by Corollary 6, something that is impossible with the equation system.

In practice, one would first ensure that $\mathsf{nsp}(D(h, g)) \cap (\mathbb{N} \setminus \{0\})^s$ is nonempty,[1] and afterwards one would use the ϱ-language technique as presented in the next section.

3. The ϱ-language technique

In the last section, we have seen how to build infinitely many context-free languages which are supersets of the equality set of two given morphisms, and how to use these languages to render some instances unsolvable. Corollary 6 allows to include non-counting information in form of a superset of the equality set. If one uses regular languages for that purpose, the resulting languages are still context-free and everything stays computable. All that is needed, is a regular superset of $E(h, g)$, and luckily, there is a simple one:

Lemma 8. *Let $M(h, g) = h(\Sigma^*) \cap g(\Sigma^*)$. Then $R(h, g) = h^{-1}(M(h, g)) \cap g^{-1}(M(h, g))$ is a regular language and $E(h, g) \subseteq R(h, g)$.*

Proof. $R(h, g)$ is clearly a regular language, since regular languages are closed under morphisms, intersection, and inverse morphisms. Assume $u \notin R(h, g)$. Then $u \notin h^{-1}(M(h, g))$ or $u \notin g^{-1}(M(h, g))$. If $u \notin h^{-1}(M(h, g))$, then $h(u) \notin h(\Sigma^*) \cap g(\Sigma^*)$ which implies $h(u) \notin g(\Sigma^*)$ and $u \notin E(h, g)$. The case $u \notin g^{-1}(M(h, g))$ is symmetric. □

Sometimes, the language $R(h, g)$ already tells us that an instance is unsolvable, or at least that some rules cannot occur in any solution:

[1] This is equivalent to the first step of the simplex algorithm.

In: $h, g : \Sigma^* \to \Delta^*$
$V \subseteq \mathbb{Z}^{|\Delta|}$
Out: One of {Unsolvable, Reduce(Σ'), Unknown}
{
$S \leftarrow \Sigma$;
$M \leftarrow h(\Sigma^*) \cap g(\Sigma^*)$;
$R \leftarrow h^{-1}(M) \cap g^{-1}(M)$;
for each $v \in V$ do
$L \leftarrow R \cap \varrho_v^{-1}(0)$;
if L is finite : return(Unsolvable);
$S \leftarrow S \cap S_\Sigma(L)$;
done
if $S = \emptyset$: return(Unsolvable);
if $S \neq \Sigma$: return(Reduce(S));
return(Unknown);
}

Algorithm 1. ϱ-language technique.

Example 9. Let $\Sigma = \{a, b, c, d\}$, $\Delta = \{0, 1\}$ and h, g be

	a	b	c	d
h	0	00	101	101
g	00	101	0	11

One has $h(\Sigma^*) = (0 + 101)^*$ and $g(\Sigma^*) = (0 + 11 + 101)^*$. Thus $M(h, g) = h(\Sigma^*) \cap g(\Sigma^*) = (0 + 101)^*$, $h^{-1}(M(h, g)) = \Sigma^*$, and $g^{-1}(M(h, g)) = (a+b+c)^*$. The language $R(h, g) = (a + b + c)^*$ does not contain the symbol d. Therefore, one can delete the rule for d.

Now we have the tools to present the new technique in Algorithm 9. It is called the ϱ-language technique. The correctness follows from the lemmas above.

Example 10. Let $\Sigma = \{a, b, c\}$, $\Delta = \{0, 1\}$ and h, g be

	a	b	c
h	0	0010	1
g	00	0	1011
$\varrho_{(1,1)}$	-1	3	-3

One has $h(\Sigma^*) = (0 + 1)^*$. Thus $M(h, g) = g(\Sigma^*) = (0+1011)^*$ and $R(h, g) = h^{-1}(M(h, g)) = (a+bcc+cacc)^*$. Consider now the language $\varrho_{(1,1)}^{-1}(0) = \{w \mid |w|_a + 3|w|_c = 3|w|_b\}$, and assume there is a word $u \in R(h, g) \cap \varrho_{(1,1)}^{-1}(0)$. The language $R(h, g)$ is a simple iteration of the 3 generating words a, bcc, and $cacc$. Let x, y, and z be the number of occurrences of these generating words in u. Then $|u|_a = x + z$, $|u|_b = y$, and $|u|_c = 2y + 3z$. But u is a member of $\varrho_{(1,1)}^{-1}(0)$ too, so it holds $|u|_a + 3|u|_c = x + z + 3(2y + 3z) = x + 6y + 10z = 3y = 3|u|_b$. This equation is fulfilled only for $x = y = z = 0$. It follows $R(h, g) \cap \varrho_{(1,1)}^{-1}(0) = \{\varepsilon\}$. Therefore, the instance is unsolvable.

Example 11. Let $\Sigma = \{a, b, c, d\}$, $\Delta = \{0, 1\}$ and h, g be

	a	b	c	d
h	0	0	10	111
g	000	001	111	1
$\varrho_{(1,0)}$	-2	-1	1	0

One has $h(\Sigma^*) = (0 + 10 + 111)^*$, $g(\Sigma^*) = (000 + 001 + 1)^*$ and $M(h, g) = h(\Sigma^*) \cap g(\Sigma^*) = ((\varepsilon + 1)00(100)^*(111 + 0) + 111)^*$. To come up with a regular expression for $R(h, g)$ is rather difficult. The corresponding finite automaton has 6 states and 19 transitions.[2] Finally, one gets $S_\Sigma(R(h, g) \cap \varrho_{(1,0)}^{-1}(0)) = \{b, c, d\}$. Thus, rule a cannot occur in any solution.

Algorithm 9 is more powerful than just looking at the regular superset or just calculating generalized balances, as shown in Examples 10 and 11: In both cases, the set $S_\Sigma(R(h, g))$ is equal to Σ, and the instances are well balanced. At this point, and if we are willing to see generalized balances as the essence of counting in PCP, we are truly beyond counting.

The example instances are neither periodic nor marked, nor unique block instances, so, no existing technique is able to decide them. Sure, it is easy for a human brain to prove that Example 10 is unsolvable,[3] but Example 11 needs a sophisticated reasoning that is difficult to implement. In fact, there exists no computer program that is able to decide the instance from Example 11. With the help of the ϱ-language technique, a computer program can reduce the instance by deleting rule a, and the resulting instance can be proved to be unsolvable by a standard argumentation.

4. Empirical data

Denote by $\mathsf{PCP}(s, w)$ the set of all instances of PCP with $|\Sigma| = s$ and $\max_{\sigma\in\Sigma} \max\{|h(\sigma)|, |g(\sigma)|\} \leqslant w$. Denote by $\mathsf{B}(s, w)$ the set of instances from $\mathsf{PCP}(s, w)$ with bad length- or element balance, by $\mathsf{R}(s, w)$ the set of all instances for which $S_\Sigma(R(h, g)) \neq \Sigma$, and by $\mathsf{C}(s, w)$ their union $\mathsf{R}(s, w) \cup \mathsf{B}(s, w)$. Use the instances

[2] Let the set of states be {0, 1, 2, 3, 4, 5}, with state 5 being start and final state. Then the transitions are: $\{0 \xrightarrow{d} 5,\ 1 \xrightarrow{a} 5,\ 1 \xrightarrow{b} 3,\ 1 \xrightarrow{c} 4,\ 1 \xrightarrow{d} 0,\ 2 \xrightarrow{a} 5,\ 2 \xrightarrow{b} 3,\ 2 \xrightarrow{c} 4,\ 2 \xrightarrow{d} 3,\ 3 \xrightarrow{a} 4,\ 3 \xrightarrow{b} 4,\ 3 \xrightarrow{c} 4,\ 3 \xrightarrow{d} 0,\ 4 \xrightarrow{a} 2,\ 4 \xrightarrow{b} 1,\ 5 \xrightarrow{a} 4,\ 5 \xrightarrow{b} 4,\ 5 \xrightarrow{c} 4,\ 5 \xrightarrow{d} 3\}$.

[3] Hint: The second symbol 1 in the g-image of c can only be covered by the h-image of c.

Table 1
Empirical data

(s, w)	$\lvert\mathsf{PCP}(s, w)\rvert$	$\lvert\mathsf{R}(s, w)\rvert$	$\lvert\mathsf{B}(s, w)\rvert$	$\lvert\mathsf{C}(s, w)\rvert$	$\lvert\mathsf{H}(s, w)\rvert$
(2, 2)	2.401	1.978	2.289	2.309	0
(2, 3)	50.625	46.594	48.377	49.689	120
(2, 4)	923.521	900.202	883.069	917.805	1.160
(3, 2)	117.649	78.878	104.881	109.285	144
(3, 3)	11.390.625	9.515.730	9.813.273	10.943.613	90.000
(3, 4)	887.503.681	852.535.982	745.795.189	878.583.817	3.148.584

from $\mathsf{PCP}(s, w) \setminus \mathsf{C}(s, w)$ as input to Algorithm 9 together with $V = \{(0, 1), (1, 0), (1, 1)\}$, and denote by $\mathsf{H}(s, w)$ the instances for which Algorithm 9 returns Unsolvable or a reduced instance.

Table 1 shows that Algorithm 9 indeed pushes the limits beyond conventional counting techniques, and that the number of instances that cannot be decided or reduced by Algorithm 9 is already quite small, even though we have not ensured that $\mathsf{nsp}(D(h, g)) \cap (\mathbb{N} \setminus \{0\})^s$ is nonempty, as suggested at the end of Section 2.

5. Conclusions

In this paper, we present a new technique that allows to decide more instances of Post's correspondence problem than known techniques. It is easy to understand and easy to implement, but nevertheless quite powerful. We give example instances which are decidable by the new technique but not by any other technique.

The new technique is extensible. For example, it is possible to include information about forbidden subwords in solutions, e.g., in Example 10 no solution can start with $aaaa$, or in Example 9 any solution must start and end with the symbol a. As long as one is able to encode such information as regular languages, it could easily be integrated into the algorithm.

Acknowledgements

Thanks to the anonymous referees for helpful comments and for call attention to [6] and [7].

References

[1] A. Ehrenfeucht, J. Karhumäki, G. Rozenberg, The (generalized) Post correspondence problem with lists consisting of two words is decidable, TCS 21 (2) (1982) 119–144.
[2] S.A. Greibach, A remark on code sets and context-free languages, IEEE Transactions on Computers C-24 (7) (1975) 741–742.
[3] V. Halava, T. Harju, M. Hirvensalo, Generalized Post correspondence problem for marked morphisms, IJAC 10 (6) (2000) 757–772.
[4] V. Halava, T. Harju, J. Karhumäki, M. Latteux, Extension of the decidability of the marked PCP to instances with unique blocks, TCS 380 (3) (2007) 355–362.
[5] V. Halava, M. Hirvensalo, R. de Wolf, Marked PCP is decidable, TCS 255 (2001) 193–204.
[6] O.H. Ibarra, C.E. Kim, A useful device for showing the solvability of some decision problems, JCSS 13 (2) (1976) 153–160.
[7] J. Karhumäki, Generalized Parikh mappings and homomorphisms, Information and Control 47 (3) (1980) 155–165.
[8] T. Harju, J. Karhumäki, Morphisms, in: G. Rozenberg, A. Salomaa (Eds.), Handbook of Formal Languages, I, Springer-Verlag, Berlin, 1997, pp. 439–510.
[9] E.L. Post, A variant of a recursively unsolvable problem, Bulletin of the American Mathematical Society 52 (1946) 264–268.