# Busy Beaver PCPs

Mario Schmidt, Heiko Stamer, and Johannes Waldmann *

Institut für Informatik, Universität Leipzig
Augustusplatz 10-11, D-04109 Leipzig, Germany
pcp@informatik.uni-leipzig.de

## 1 Introduction

One way to investigate a problem is to look for small instances that are hard. The goal is to bound the hardness of the instance by a function of its size. (The existence of) such a function gives information on the problem's (decidability and) complexity. We apply this idea to the Post Correspondence Problem (PCP) [Pos46]. This resembles the hunt for Busy Beaver Turing machines [Mar00].

A *PCP instance* is a finite list $[(u_1, v_1), \dots (u_n, v_n)]$ of pairs of words $u_i, v_i \in \Sigma^*$ that determines two morphisms $\phi, \psi : \Delta^* \to \Sigma^*$ with $\Delta = \{1, \dots, n\}$ by $\phi(i) = u_i$ resp. $\psi(i) = v_i$. A *solution* of an instance is a member of the equality set $E(\phi, \psi) = \{w \mid w \in \Delta^+, \phi(w) = \psi(w)\}$. It is well known that it is undecidable whether a PCP instance has a solution.

The *size* of an instance is the number of its pairs (i. e. $|\Delta|$). We call PCP($n$) the set of all instances of size $n$. It is known that solvability of PCP(7) is undecidable [MS96], while PCP(2) is decidable [EKR82]. We focus on PCP(3).

The *width* of an instance is the maximal length of a word $\max_{i\in\Delta}\{|u_i|, |v_i|\}$. We call PCP($n, w$) the set of all instances of size $n$ and width $w$. We want to know, for fixed $n = 3$, how the maximal length of a minimal solution grows with $w$; with the aim of collecting information on the decidability of PCP(3).

In this note, we show some of the current record holding instances, then we describe a family of instances that seems to contain a lot of hard ones, and finally we discuss growth rates.

While we could not determine the status of PCP(3) so far, we think that our work nicely demonstrates techniques from combinatorics of words, (string) rewriting, and termination, applied to an interesting problem that is accessible to students, and thus can be used in teaching. In fact Stamer's implementation of the search algorithms started from a programming assignment.

Details will be elaborated in Schmidt's forthcoming diploma thesis, and an accompanying technical report. This abstract just collects results and conjectures.

* corresponding author

## 2 PCP(3) record holders

When looking for solutions of (randomly generated) instances, it is important to prune unsuccessful branches of the search tree, and to reject unsolvable instances early, without wasting too much time.

Moreover, note that it is not enough to find (or guess) and then verify a solution, since we also have to check that the solution has minimal length.

Some applicable techniques are explained (independently) in [Lor00], and we implemented some extensions. We found these PCP(3) instances of seemingly maximal hardness (the first one already is in [Lor00]):

(NOTE: table not printed because it became obsolete. Refer to the web site for up–to–date information.)

The current list of record holders is maintained at our web site `http://www.informatik.uni-leipzig.de/~pcp/`. We welcome any addition, correction, or independent verification of entries of this list. Moreover, if you visit our web site, you can play an online PCP puzzle game. Each day, a fresh (semi-hard) PCP instance is generated automatically.

## 3 Morphic PCPs

We noticed that a lot of the hard instances seem to follow the simple pattern $M(u,v) = \begin{pmatrix} 0 & 1 & u \\ v & 0 & 1 \end{pmatrix}$, where $v$ starts with 0, and $u$ ends with 1. (For the record PCP(3,3) (see above), we even have $u = v$.)

We call these *morphic* instances, because if we only use the first two pairs, we obtain the infinite word that is the limit of the morphism $0 \mapsto v, 1 \mapsto 0$. We get a (different) infinite word by using the last two pairs only, from the morphism $0 \mapsto 1, 1 \mapsto \tilde{u}$. The question is how these words "meet".

A special case are *Fibonacci* instances $F(u) = M(u, 01)$, so named because the first two pairs give the infinite Fibonacci word $f = 01001010010\ldots$.

It is clear that for $F(u)$ to be solvable, $u$ must be a factor of $f$ that ends with 1. Is this condition also sufficient for solvability? We could verify this for $|u| \leq 6$, and some larger ones, and found rather hard instances among them. The table lists the lengths of the minimal solutions:

| $u$ | 1 | 01 | 001 | 101 | 0101 | 1001 | 00101 | 01001 | 001001 | 100101 | 101001 | 0100101 | 0101001 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $l$ | 1 | 3 | 30 | 44 | 6 | 78 | 8 | 80 | 120 | 36 | 108 | 55 | 22 |

Solvability of $F(1001001)$ is open.

NOTE: $F(1001001)$ is unsolvable. See the slides for a proof sketch.

## 4 Growth rates

As noted in [Lor00], the instance family $\begin{pmatrix} 0 & 10^n \\ 0^n1 & 0 \end{pmatrix}$ with minimal solutions $A^nB^n$ achieves the fastest known growth (of the length of a minimal solution, seen as a function of the instance width) among PCP(2). Still there does not seem to be a simple proof for this — such a proof would imply a simple proof of decidability of PCP(2).

For PCP(3) families, we can achieve quadratic growth, by $\begin{pmatrix} 0 & 0^n & 1^n0 \\ 00 & 1 & 0 \end{pmatrix}$, with minimal solution $A^{n^2}B^nC$. A more non–obvious, but still only quadratic behaviour is exhibited by $\begin{pmatrix} 01 & 1^{2n}0 & 1 \\ 1 & 10 & 1010 \end{pmatrix}$, for suitable values of $n$:

| n | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| l | 4 | 10 | 32 | 22 | 84 | 170 | 240 | 46 | 208 | 522 | 128 | 280 | 760 |

Interestingly, the proof requires some number theory: the length of the solution depends on the number of incongruent values of $2^k$ modulo $2n-1$.

So far, we did not find larger growth for PCP(3) families (neither polynomials of higher degree, nor exponentials). This gives hope for decidability ...

## References


[EKR82] A. Ehrenfeucht, J. Karhumaki, and G. Rozenberg. The (generalized) post correspondence problem with lists consisting of two words is decidable. *Theoretical Computer Science*, 21(2):119–144, November 1982.

[Lor00] Richard J Lorentz. Creating difficult instances of the post correspondence problem. 2nd International Conference on Computers and Games, Hamamatsu, 2000.

[Mar00] Heiner Marxen. Busy beaver. `http://www.drb.insel.de/~heiner/BB/index.html`, 2000.

[MS96] Yuri Matiyasevich and Géraud Sénizergues. Decision problems for semi-Thue systems with a few rules. In *Proceedings, 11th Annual IEEE Symposium on Logic in Computer Science*, pages 523–531, New Brunswick, New Jersey, 27–30 July 1996. IEEE Computer Society Press.

[Pos46] Emil Post. A variant of a recursively unsolvable problem. *Bulletin of the American Mathematical Society*, 52:264–268, 1946.