

## Post's Correspondence Problem

Robert M. Keller  
Harvey Mudd College  
3 November 2003

### What is Post's Correspondence Problem (PCP)?

- Named after logician Emil Post\* who, like Turing, studied fundamental models of computation, of a caliber comparable to Gödel, Turing, and Church (but "scooped" by them).
- *NOT* a problem that Post had with Zipcodes (as suggested by a Princeton student on his final exam).
- I don't know if relation to Emily Post (1873–1960), also a philosopher.

### Emil Post (center), 1897-1954



### Definition of the PCP

- Devise an algorithm that will do the following:
  - Given two equal-length lists of non-null strings over a common alphabet  $x = [x_1, x_2, \dots, x_n]$  and  $y = [y_1, y_2, \dots, y_n]$ , determine whether there is a list of indices  $[i_1, i_2, \dots, i_m]$ , possibly with repetition, such that the **concatenations** of the **corresponding** elements in each list are equal:
- A pair of lists is an instance of the problem, and when there is such a list of indices, the **instance** is said to have a **solution**.

$$x_{i_1} x_{i_2} \dots x_{i_m} = y_{i_1} y_{i_2} \dots y_{i_m}$$

### Other Formulations

- Obviously we could present the input as a single **set** of pairs (or "dominoes")

$$\{(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)\}$$

rather than as two separate lists.

### Example 1

- $x = [100, 0, 1]$   
 $y = [1, 100, 00]$
- This instance has a solution:  $[1, 3, 1, 1, 3, 2, 2]$ , since
$$\begin{array}{r} 100\ 1\ 100\ 100\ 1\ 0\ 0 \\ =\ 1\ 00\ 1\ 1\ 00\ 100\ 100 \end{array}$$
- As pairs, the problem would be stated:  $\{(100, 1), (0, 100), (1, 00)\}$ .

## Example 2

- $x = [10, 011, 101]$   
 $y = [101, 11, 011]$
- As pairs, the problem would be stated:  
 $\{(10, 101), (011, 11), (101, 011)\}$ .
- This instance has no solution. A solution would obviously have to start with 1:  
10  
101  
then be followed by 3:  
10 101  
101 011  
then by another 3:  
10 101 101  
101 011 011  
and so on, but never closing.

## Why is the PCP Important?

- The PCP is a problem that doesn't require the notion of Turing machines in its statement.
- A child can understand instances of the problem ("dominoes").
- The problem is seductive: looks solvable, superficially.
- The PCP will be shown unsolvable.
- Therefore the PCP is a clean, isolated unsolvable problem, that can be reduced to other problems to show them unsolvable.

## Notes

- Individual instances can be shown solvable or unsolvable by ad hoc means.
- Solvability of individual instances is not the problem.

## How to show unsolvable?

- Attempt to reduce a known unsolvable problem to PCP.
- Possible if we could get an instance to simulate a Turing machine:
  - The instance would have a solution iff the TM halts.
  - Would need an instance of PCP for every possible TM.
  - For example,  $\text{TM halts} \leq \text{PCP}$  has a solution.

## Modified PCP (MPCP)

- The MPCP is a stepping stone that makes the reduction simpler.
- The MPCP is the same as the PCP, except that to be valid, a solution **must** begin with the first index.
- We will show:

$$\text{Halting} \leq \text{MPCP} \leq \text{PCP}$$

## $\text{MPCP} \leq \text{PCP}$

- This means that for every instance M of MPCP there is an instance P of PCP such that M has a solution iff P does.
- Let M be an instance of MPCP. Select two new symbols not otherwise used in M, say # and \$.
- For each pair  $(x, y)$  of M define a pair  $(x', y')$  of P:
  - Each symbol of x is followed by # to get  $x'$ .
  - Each symbol of y is preceded by # to get  $y'$ .
- There are two additional pairs in P:
  - A pair just like the first pair above, with x preceded by #.
  - A pair  $($, #)$ .

### Example 3

- M:
 
$$x = [10111, 1, 10]$$

$$y = [10, 111, 0]$$
- P:
 
$$x' = [1\#0\#1\#1\#1\#, 1\#, 1\#0\#, \#1\#0\#1\#1\#, \$]$$

$$y' = [\#1\#0, \#1\#1\#, \#0, \#0, \#\$]$$
- A solution of M: [1, 2, 2, 3]
 
$$\underline{10111} \ \underline{1} \ \underline{1} \ \underline{10} = \underline{10} \ \underline{111} \ \underline{111} \ \underline{0}$$
- A solution of P: [4, 2, 2, 3, 5]
 
$$\underline{\#1\#0\#1\#1\#1\#} \ \underline{1\#} \ \underline{1\#} \ \underline{1\#0\#} \ \underline{\$}$$

$$= \underline{\#1\#0} \ \underline{\#1\#1\#1} \ \underline{\#1\#1\#1} \ \underline{\#0} \ \underline{\#\$}$$

As constructed, P has a solution iff M does

- Proof:
  - **Assume that P has a solution.** By the form of the pairs in P, the solution must **start** and **end** with the last two pairs that were added in going from M to P. Start because there is only one pair in which both strings begin with # and end because there is only one in which both end with \$.
  - But the start pair corresponds to the first pair of M.
  - Also, if there is a solution to P, then there is a corresponding solution to M be removing the # and \$ symbols.
  - **Conversely, assume that M has a solution.** Then there is a corresponding solution of P having as its first pair the first of the new pairs added and as its last pair the second of the new pairs added, and all other pairs the same.

### Halting $\leq$ MPCP

- Let T be an arbitrary Turing machine.
- We will show how to construct an instance M of MPCP such that T halts iff M has a solution.
- Certain technical assumptions will be made to make the construction work.
- These will be indicated as needed.
- We will enumerate the pairs in M in groups.

### Halting $\leq$ MPCP Construction

- Suppose the initial state of the TM is  $q_0$  and the initial tape is  $w$ . Then there will be a pair in M:
 
$$(\#, \#q_0w\#)$$
- For each tape symbol  $X$ , there will be a pair in M:
 
$$(X, X)$$
- There will be a pair in M:
 
$$(\#, \#)$$

### Halting $\leq$ MPCP Construction

- For each transition  $q, X \rightarrow p, Y, R$  there is a pair:
 
$$(qX, Yp)$$
 and a pair:
 
$$(q\#, Yp\#)$$
- For each transition  $q, X \rightarrow p, Y, L$  and each tape symbol  $Z$  there is a pair:
 
$$(ZqX, pZY)$$
 and a pair:
 
$$(Zq\#, pZY\#)$$

### Halting $\leq$ MPCP Construction

- For the halting state  $h$ , and each tape symbol  $X$  and  $Y$ , there are pairs:
 
$$(XhY, h)$$

$$(Xh, h)$$

$$(hY, h)$$
- Finally, there is the pair:
 
$$(h\#\#, \#)$$
- We call the pairs on this page "Completion Pairs".

## Technical Assumptions (WLOG)

- Assume that the TM never writes a blank.
- Assume that it never moves left of its initial position (i.e. one-way infinite tape).

## Operation

- As the TM goes through its transitions, the corresponding pairs of the MPCP build up a "partial solution", meaning that one pair is always a prefix of the other. Each time a pair is added to the partial solution, it is of the form (*Before*, *After*) where *Before* is the tape and control configuration before, and *After* is the configuration after.
- In order to keep to a partial solution, the *Before* of the next pair must **match** the *After* of the previous pair. #'s are used to keep the configurations separate.
- If and when the TM **halts**, the partial solution is extended to a complete solution by using the **Completion Rules**.

## Further Details

- For further details of the proof and a worked example, please consult Martin pp 425-430.

## Some Applications

- Undecidability of context-free language properties:
  - Do the languages of two context-free grammars intersect in a non-empty set?
  - Is the language of a context-free grammar ambiguous?
- Undecidability of certain theories in predicate logic (next part of course)