CS81 Spring 2012: PCP[1]

We describe an interesting undecidable language that is not directly related to Turing machines.

> **Post Correspondence Problem** (PCP)
> *Input*: Two sets of strings $A = \{\alpha_1, \ldots, \alpha_n\}$ and $B = \{\beta_1, \ldots, \beta_n\}$ over Σ.
>
> *Output*: Decide if there are indices $i_1, \ldots, i_m \in \{1, 2, \ldots, n\}$ so that
>
> $$\alpha_{i_1}\alpha_{i_2}\ldots\alpha_{i_m} = \beta_{i_1}\beta_{i_2}\ldots\beta_{i_m}.$$

Example: Suppose A and B are described by the following table.

i	α_i	β_i
1	1	111
2	10111	10
3	10	0

A solution is given by $i_1 = 2$, $i_2 = i_3 = 1$, and $i_4 = 3$, since

$$(10111)(1)(1)(10) = (10)(111)(111)(0).$$

A variant of PCP requires that $i_1 = 1$; that is, the first pair must be used as the first element in the solution. It turns out we may assume this extra condition without loss of generality.

> **Modified Post Correspondence Problem** (MPCP)
> *Input*: Two sets of strings $A = \{\alpha_1, \ldots, \alpha_n\}$ and $B = \{\beta_1, \ldots, \beta_n\}$.
>
> *Output*: Decide if there are indices $i_2, \ldots, i_m$ so $\alpha_1\alpha_{i_2}\ldots\alpha_{i_m} = \beta_1\beta_{i_2}\ldots\beta_{i_m}$.

Theorem 1. MPCP $\preceq$ PCP.

Proof. (sketch) We show that if PCP is decidable then MPCP is decidable. Let $A = \{\alpha_1, \ldots, \alpha_n\}$ and $B = \{\beta_1, \ldots, \beta_n\}$ be two sets of strings over Σ that is an instance of MPCP. We convert this to an instance of PCP as follows. We use two additional symbols # and \$ that are not elements of Σ.

[1] Source: Hopcroft and Ullman.

We construct two sets of strings $C = \{\gamma_0, \ldots, \gamma_{n+1}\}$ and $D = \{\delta_0, \ldots, \delta_{n+1}\}$ which will be the instance of PCP. Let γ_i be obtained from α_i by inserting the symbol $\#$ *after* each symbol of α_i. Let δ_i be obtained from β_i by inserting the symbol $\#$ *before* each symbol of β_i. The following new words are also added to the collection C and D:

$$\begin{aligned} \gamma_0 &= \#\gamma_1 \\ \delta_0 &= \delta_1 \\ \gamma_{n+1} &= \$ \\ \delta_{n+1} &= \#\$ \end{aligned}$$

We claim that the instance of PCP with C and D has a solution iff the instance of MPCP with A and B has a solution. □

Theorem 2. HALT $\preceq$ MPCP.

Proof. (sketch) Given an instance $(\langle M \rangle, w)$ of HALT, we convert this to an instance of MPCP. Let $M = (Q, \Sigma, \delta, s)$. Consider the following sets A and B of strings. We assume that the symbol $\#$ is *not* an element of Σ (whereas the blank symbol $\square$ is).

Group	A	B	condition
Start	$\#$	$\#\,\square\, ws\#$	$w \in (\Sigma \setminus \{\square\})^\star$
Fill	X	X	$\forall X \in \Gamma$
	$\#$	$\#$	
Move			$\forall q \neq h, \forall X, Y, Z \in \Sigma$
	qX	Yp	if $\delta(q, X) = (p, Y, \mathrm{R})$
	ZqX	pZY	if $\delta(q, X) = (p, Y, \mathrm{L})$
	$q\#$	$Yp\#$	if $\delta(q, \square) = (p, Y, \mathrm{R})$
	$Zq\#$	$pZY\#$	if $\delta(q, \square) = (p, Y, \mathrm{L})$
Erase	$X\,h\,Y$	h	$\forall X, Y \in \Sigma$
	$X\,h$	h	
	$h\,Y$	h	
Stop	$h\,\#\#$	$\#$	

The idea of this construction is to use MPCP to *search* for a sequence of configurations from the *initial* configuration $\square\, ws$ to some *halting* configuration. □

Example: Consider the Turing machine M in Figure 1 whose start state is $s = q_1$ and halt state is $h = q_3$. The instance of PCP corresponding to M on input $w = 01$ is given by the following table.

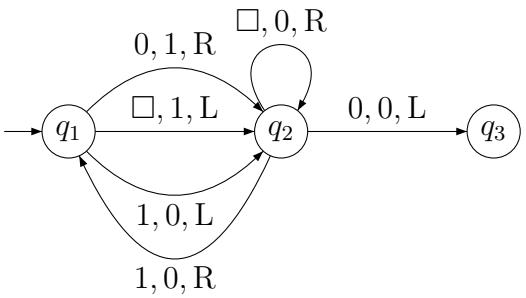

Figure 1: A Turing machine M.

Group	A	B
Start	$\#$	$\#\,\square\,01q_1\#$
Fill	0	0
	1	1
	$\square$	$\square$
	$\#$	$\#$
Erase	$0q_30$	q_3
	$0q_31$	q_3
	$0q_3\,\square$	q_3
	$1q_30$	q_3
	$1q_31$	q_3
	$1q_3\,\square$	q_3
	$\square\,q_30$	q_3
	$\square\,q_31$	q_3
	$\square\,q_3\,\square$	q_3
	$0q_3$	q_3
	$1q_3$	q_3
	$\square\,q_3$	q_3
	q_30	q_3
	q_31	q_3
	$q_3\,\square$	q_3
Stop	$q_3\#\#$	$\#$

Group	A	B
Move	q_10	$1q_2$
	$0q_11$	q_200
	$1q_11$	q_210
	$\square\,q_11$	$q_2\,\square\,0$
	$0q_1\,\square$	q_201
	$1q_1\,\square$	q_211
	$\square\,q_1\,\square$	$q_2\,\square\,1$
	$0q_1\#$	$q_201\#$
	$1q_1\#$	$q_211\#$
	$\square\,q_1\#$	$q_2\,\square\,1\#$
	$0q_20$	q_300
	$1q_20$	q_310
	$\square\,q_20$	$q_3\,\square\,0$
	q_21	$0q_1$
	$q_2\,\square$	$0q_2$
	$q_2\#$	$0q_2\#$

The computation of M on input 01 is given by the sequence of configurations:

$$01q_1 \to 0q_211 \to 00q_11 \to 0q_200 \to q_3000$$

Here is the encoding of that sequence using the PCP strings.

$$\begin{aligned} A &= \#\,\square\,01q_1\#\,\square\,0q_211\#\,\square\,00q_11\#\,\square\,0q_200\#\,\square\,q_3000\#q_300\#q_30\#q_3\#\# \\ B &= \#\,\square\,\underline{01q_1\#}\,\square\,\underline{0q_211\#}\,\square\,\underline{00q_11\#}\,\square\,\underline{0q_200\#}\,\square\,\underline{q_3000\#}\underline{q_300\#}\underline{q_30\#}\underline{q_3\#\#} \end{aligned}$$

As an application of the undecidability of PCP, we can show that it is undecidable to determine if a context-free grammar is ambiguous.

Theorem 3. *The language* $\mathsf{AMB} = \{\langle G\rangle : G \text{ is an ambiguous context-free grammar}\}$ *is undecidable.*

Proof. (sketch) We show $\mathsf{PCP} \preceq \mathsf{AMB}$. Let $A = \{\alpha_1, \ldots, \alpha_n\}$ and $B = \{\beta_1, \ldots, \beta_n\}$ be two sets of strings over Σ that is an instance of PCP. Let $c_1, \ldots, c_n$ be new symbols not in Σ. Define two languages

$$\begin{aligned} L_A &= \{\alpha_{i_1}\alpha_{i_2}\ldots\alpha_{i_m}c_{i_m}c_{i_{m-1}}\ldots c_{i_1} : m \geq 1\} \\ L_B &= \{\beta_{i_1}\beta_{i_2}\ldots\beta_{i_m}c_{i_m}c_{i_{m-1}}\ldots c_{i_1} : m \geq 1\} \end{aligned}$$

Consider the following context-free grammar $G = (V, \Gamma, R, S)$ where $V = \{S, S_A, S_B\}$, $\Gamma = \Sigma \cup \{c_1, \ldots, c_n\}$, and R consists of the rules:

- $S \to S_A \mid S_B$.
- $S_A \to \alpha_i S_A c_i \mid \alpha_i c_i$, for $i = 1, \ldots, n$.
- $S_B \to \beta_i S_B c_i \mid \beta_i c_i$, for $i = 1, \ldots, n$.

Thus, $L(G) = L_A \cup L_B$. If the instance (A, B) of PCP has a solution, say $i_1, \ldots, i_m$, then there is a word

$$\alpha_{i_1}\alpha_{i_2}\ldots\alpha_{i_m}c_{i_m}c_{i_{m-1}}\ldots c_{i_1} = \beta_{i_1}\beta_{i_2}\ldots\beta_{i_m}c_{i_m}c_{i_{m-1}}\ldots c_{i_1}$$

where the former is a string in L_A and the latter is a string in L_B. The former has a derivation that begins with $S \Rightarrow S_A$ whereas the latter has a derivation that begins with $S \Rightarrow S_B$. So, G is ambiguous.

It is also true that if G is ambiguous then any word derived from S_A has at most one leftmost derivation and similarly for any word derived from S_B. This is because these derivations are enforced by the choice of the symbols c_i's. This will imply that (A, B) has a solution as an instance of PCP. □