**PCP is undecidable**

\[ PCP = \{ \langle P \rangle \mid P \text{ is an instance of the Post correspondence problem with a match} \} \]

We deal with a modified version of the problem

\[ MP_{PCP} = \{ \langle P \rangle \mid P \text{ is an instance of the Post correspondence problem with a match starting with the first domino} \} \]

Then we transform \( A_{TM} \) to \( MP_{PCP} \) in such a way that, for each \( x = \langle M, w \rangle \):

(*) the matched string generated by the dominos for \( x \) will encode accepting computation of \( M \) on \( w \).

Let \( M = (Q, \Sigma, \Gamma, \delta, q_0, q_{accept}, q_{reject}) \).

---

**Three Kinds of Dominos**

1. The Initial Domino: \([#q_0x_1\ldots x_n#] \).
   The lower part is one computational step ahead of the upper part.

2. The Computation Dominos:
   Correspond to configuration rewriting rules. Filling the upper part that is lagging behind with the computational dominos will advance the lower part by a single step of \( M \).

3. The Cleaning Dominos:
   Dominos that gradually "eat up" the lower part while keeping the state in \( q_{accept} \).

---

**Post Correspondence Problem (PCP)**

Given a collection of dominos, each containing a string on each half, decide whether the dominos can be placed with repetition in line so that the upper halves and the lower halves read the same from left to right. Such a placement of dominos is called a match.

**Example:** Given a collection

\[ \left\{ \frac{b}{ca}, \frac{a}{bc}, \frac{ca}{a}, \frac{abc}{c} \right\} \]

the list

\[ \left\{ \frac{a}{ab}, \frac{b}{ca}, \frac{ca}{a}, \frac{a}{ab}, \frac{abc}{c} \right\} \]

yields the string \( abcaabc \) on both halves.
From MPCP to PCP

For a string \( u = u_1 u_2 \cdots u_m \), let \( *u = *u_1 *u_2 * \cdots *u_m \), \( u* = u_1 *u_2 * \cdots *u_m * \), where * is a new symbol.

Modify the start domino \( \frac{1}{L} \) to \( \frac{x_i}{L} \) and each other domino \( uv \) to \( \frac{x_j}{L} \); add a new domino \( \frac{x_i}{o} \), where \( o \) is a new symbol.

This will force the start domino to be the first one and the newly added one to be the last one.

The Computation Dominos

- \( \frac{\#}{\#} \), \( \frac{\#}{\#} \), and \( \frac{a}{a} \) for each \( a \in \Gamma \).
- For each \( p, q \in Q \) and \( a, b, c \in \Gamma \) such that \( \delta(p, a) = (q, b, L) \), \( \frac{\#px}{\#y} \) and \( \frac{epx}{q} \).
- For each \( p, q \in Q \) and \( a, b, c \in \Gamma \) such that \( \delta(p, a) = (q, b, R) \), \( \frac{pc}{q} \).

Computable Functions

A function \( f : \Sigma^* \to \Sigma^* \) is **computable** if there exists a Turing machine \( M \) such that for every \( x \in \Sigma^* \), \( M \) on \( x \) halts with just \( f(x) \) on its tape.

**Example:** Let \( \Sigma \) be a fixed alphabet. Define \( f : \Sigma^* \to \Sigma^* \) as follows:

- If \( w = \langle M \rangle \) for some Turing machine, then \( f(w) = \langle M' \rangle \) where \( M' \) is \( M \) with \( q_{\text{accept}} \) and \( q_{\text{reject}} \) swapped.
- Otherwise, \( f(w) = w \).

Then \( f \) is computable.
A language $U \subseteq \Sigma^*$ is mapping reducible to $V \subseteq \Sigma^*$ (write $U \leq_m V$) if there exists a computable function $f: \Sigma^* \rightarrow \Sigma^*$ such that for every $x \in \Sigma^*$,

$$x \in A \text{ if and only if } f(x) \in B.$$ 

In other words, the function $f$ maps members of $A$ to members of $B$ and nonmembers of $A$ to nonmembers of $B$.

Example: $A_{TM} \leq_m HALT_{TM}$. We can use $f(x) = x$ if $x$ is not of form $\langle M, w \rangle$. Otherwise, $f(\langle M, w \rangle) = \langle M_1, w \rangle$, where $M_1$, for input $y$, simulates $M$ on $y$, and accepts $y$ if $M$ accepts $y$, and otherwise runs forever.

Then $x \in A_{TM}$ (which implies $x$ is of form $\langle M, w \rangle$ and $M$ accepts $w$) iff $f(x) \in HALT_{TM}$.

---

**Equivalence of the Turing-Recognizable Languages**

**Theorem.** $EQ_{TM}$ is neither Turing-recognizable nor co-Turing-recognizable.

**Proof.** Show that $A_{TM}$ is mapping reducible to $EQ_{TM}$ as well as to $\overline{EQ_{TM}}$. Let $s \in EQ_{TM}$ and $t \in EQ_{TM}$ be fixed.

**Reduction to $EQ_{TM}$**

- If $x$ is of the form $\langle M, w \rangle$, then $f(x) = \langle M_1, M_2 \rangle$, where
  - $M_1$ accepts every input $y$; and
  - $M_2$ first simulates $M$ on $w$ and accepts its own input $y$ if $M$ has accepted $w$.

- Otherwise, $f(x) = t$.

$f$ is computable, and for every $x$, $x \in A_{TM}$ if and only if $f(x) \in EQ_{TM}$.
Proof (cont’d)

Reduction to $EQ_{TM}$

- If $x$ is of the form $\langle M, w \rangle$, then $g(x) = \langle M_1, M_2 \rangle$, where
  - $M_1$ rejects every input $y$; and
  - $M_2$ first simulates $M$ on $w$ and accepts its own input $y$ if $M$ has accepted $w$.
- Otherwise, $g(x) = s$.

$g$ is computable and for every $x, x \in A_{TM}$ if and only if $g(x) \notin EQ_{TM}$.

Thus, $A_{TM} \leq_m EQ_{TM}$ and $A_{TM} \leq_m \overline{EQ_{TM}}$. 

\[\]