Reduction

- **A Reduction is an algorithm for transforming one problem into another problem.**
- Why we need reduction?
  - Solve a problem that is similar to a problem we've already solved.
  - Prove that a problem is hard to solve.
- Examples of reduction:
  1. **Mapping Reduction** (i.e. Many-One Reduction)

     $$L_1 \leq_M L_2$$

  2. **Turing Reduction**

     $$L_1 \leq_T L_2$$

  3. **Karp Reduction** (i.e. Polynomial-time Mapping Reduction)
  4. **Cook Reduction** (i.e. Polynomial-time Turing Reduction)
- Examples:
  1. $a \cdot b \leq_T a^2$ since $a \cdot b = \frac{(a+b)^2 - a^2 - b^2}{2}$. (*Turing Reduction*)
  2. $Max\ Flow \leq_M Maximum\ Matching$. (*Mapping Reduction*)

Def 4.4(Mapping Reduction) **Let $L_1, L_2 \subseteq \{0,1\}^*$. Say L_1 is mapping reducible to L_2, denoted by $L_1 \leq_M L_2$, if there is a computable function $\phi : \{0,1\}^* \to \{0,1\}^*$ s.t. $x \in L_1 \Leftrightarrow \phi(x) \in L_2$ for any x.**

Prop 4.5

1. For any L, $L \leq_M L$.
2. $L_1 \leq_M L_2 \Leftrightarrow \bar{L_1} \leq_M \bar{L_2}$.
3. If $L_1 \leq_M L_2, L_2 \leq_M L_3$, then $L_1 \leq_M L_3$.

Def 4.6(Oracle Turing Machine) **A k-tape TM with an oracle for language L is a k-tape TM with 2 additional states q_{ask} and $q_{response}$. The first $k-1$ tapes are input and work tapes. The last tape is the oracle tape. When it enters q_{ask}, the following actions are performed in one step:**

1. The string z that is written on the oracle tape is erased.
2. If $z \in L$, symbol 1 is written to the leftmost cell of the oracle tape. Otherwise, 0 is written.
3. The oracle tape head is moved to the leftmost.
4. The machine enters $q_{response}$ state.

- Similarly, we can define a TM with an oracle function f.

Def 4.7 **Let $L_1, L_2 \subseteq \{0,1\}^*$. L_1 is Turing reducible to L_2, denoted by $L_1 \leq_T L_2$, if there is an oracle TM, with an oracle for L_2, that decides L_1.**

Prop 4.8

1. For any decidable languages L_1 and L_2, we have $L_1 \leq_T L_2$.
2. If $L_1 \leq_T L_2, L_2 \leq_T L_3$, then $L_1 \leq_T L_3$.
3. For any L, $L \leq_T L$, and $L \leq_T \bar{L}$.
4. If $L_1 \leq_M L_2$, then $L_1 \leq_T L_2$.

Def 4.14(Nontrivial Property of Languages) **Property P is about the language recognized by TMs if, whenever $L(M) = L(N)$, P contains $\langle M \rangle \Leftrightarrow P$ contains $\langle N \rangle$. The property is nontrivial if there is a TM_α s.t. $\alpha \in P$, and $\exists TM_\beta$ s.t $\beta \notin P$.**

Thm 4.15(Rice's Theorem) **Any nontrivial property about the language recognized by TMs is undecidable.**

Proof: WLOG(Without Loss of Generality), assume $\phi \notin P$. Assume for contradiction that P is decidable by a TM M_P.

Since P is nontrivial, pick an arbitrary $\beta \in P$. (Since $\phi \notin P$, the empty TM (that always rejects) is not in P)

We construct a TM M_{accept} as follows:

1. On input (α, x), construct TM M_P, $\gamma = \gamma(\alpha, x)$ as follows:
   a. Simulate M_α on input x until M_α accepts x. Otherwise, make M_α loop forever.
   b. Simulate M_β on input y and accept if and only if M_β accepts.
2. Run M_P on input γ, accept if and only if M_P accepts.

Claim M_{accept} decides L_{accept}

- Case 1: $(\alpha, x) \in L_{accept}$, $L(M_\gamma) = L(M_\beta)$. So, M_P accepts γ.
- Case 2: $(\alpha, x) \notin L_{accept}$, $L(M_\gamma) = \phi$. So, M_P rejects γ.

Q.E.D.

Post Correspondence Problem(PCP)

- *Domino* $[\frac{a}{ab}]$
- *A Collection of Dominos* $\{[\frac{b}{ca}], [\frac{a}{ab}], [\frac{ca}{a}], [\frac{abc}{c}]\}$
- *A Match* $[\frac{a}{ab}] \cdot [\frac{b}{ca}] \cdot [\frac{ca}{a}] \cdot [\frac{a}{ab}] \cdot [\frac{abc}{c}]$
- **The PCP is to determine whether a collection of dominos has a match.**

Formally, an instance of PCP is $P = \{[\frac{t_1}{b_1}] \cdot \ldots \cdot [\frac{t_k}{b_k}]\}$, where $t_i, b_i \in \Sigma^*$. A match is a sequence $i_1, i_2, \ldots, i_l$, where

$$t_{i_1} t_{i_2} \ldots t_{i_l} = b_{i_1} b_{i_2} \ldots b_{i_l}$$

$PCP = \{\langle P \rangle | P \ has \ a \ match\}$.

Thm 4.16(Emil Post 1946) **PCP is undecidable.**

Proof idea: Reduce L_{accept} to PCP. Given any α, x, construct a PCP instance $P_{\alpha,x}$ s.t. M_α accepts $x \Leftrightarrow P_{\alpha,x}$ has a match.

Proof: Since L_{accept} is undecidable, we can conclude PCP is undecidable.

Given a TM_α and input w, construct a PCP instance $P = P(\alpha, w)$ s.t.

1. If M_α accepts w, then $P \in PCP$.
2. If M_α does not accept w, then $P \notin PCP$.

Goal:

1. $L_{accept} \leq_M MPCP$
2. $MPCP \leq_M PCP$

- **Handle 3 technical points:**
    1. M never attempts to move its head off the leftmost end of the tape.
    2. If $w = \epsilon$, use the string in place $\sqcup$ in place of w.
    3. Modify PCP to require that a match starts with $[\frac{t_1}{b_1}]$.

       Let $MPCP = \{\langle P\rangle | P \in PCP\ and\ \exists\ a\ match\ that\ starts\ with\ [\frac{t_1}{b_1}]\}$.

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

       Algorithmically construct $P' = P'(\alpha, w)$ s.t. $P' \in MPCP \Leftrightarrow M_\alpha$ accepts w.

Part 1. Put $[\frac{\#}{\#q_0w_1...w_n\#}]$ into P' as the first domino.

Part 2. For every $a, b \in \Gamma$, and every $q, r \in Q$, if $\delta(q, a) = (r, b, R)$, put $[\frac{qa}{br}]$ into P'.

Part 3. If $\delta(q, a) = (r, b, L)$, put $[\frac{cqa}{rcb}]$ into P' for every $c \in \Gamma$.

Part 4. For every $a \in \Gamma$, put $[\frac{a}{a}]$ into P'.

Part 5. Put $[\frac{\#}{\#}], [\frac{\#}{\sqcup\#}]$ into P'.

Part 6. For every $a \in \Gamma$, put $[\frac{aq_{accept}}{q_{accept}}], [\frac{q_{accept}a}{q_{accept}}]$ into P'.

Part 7. Put $[\frac{q_{accept}\#\#}{\#}]$ into P'.

We claim M_α accepts $w \Leftrightarrow P' \in MPCP$.