Lesson 9: Reducibility

Theme: Reductions as a tool to prove undecidability.

1 Reductions

Consider a function $F : \Sigma^* \rightarrow \Sigma^*$. A TM $M$ that computes $F$ is a 2-tape TM that accepts every word $w \in \Sigma^*$ and when it halts, the content of its second tape is $F(w)$. There is no restriction on the content of the first tape. That is, on every word $w$, $M$ accepts $w$ with the accepting run:

$$(q_0, \bullet w, \bullet) \vdash \cdots \vdash (q_{\text{acc}}, u, \bullet F(w))$$

for some string $u$ (which denotes the content of the first tape). A function is computable, if there is a TM that computes it.

Definition 9.1 A language $L_1$ is mapping reducible to another language $L_2$, denoted by $L_1 \leqslant_m L_2$, if there is a computable function $F$ such that for every $w \in \Sigma^*$:

$$w \in L_1 \text{ if and only if } F(w) \in L_2$$

The function $f$ is called mapping reduction.

Sometimes we omit the word “mapping” and call it simply “reducible” or “reduction,” instead of “mapping reducible” or “mapping reduction.” Intuitively $L_1 \leqslant_m L_2$ means that $L_2$ is “computationally more general,” or “more general” than $L_1$ and that a TM for deciding $L_2$ can be used to decide $L_1$.

Definition 9.2 A language $L_1$ is Turing reducible to another language $L_2$, denoted by $L_1 \leqslant_T L_2$, if by assuming that $L_2$ is decidable by a TM $M_2$, there is a TM $M_1$ that decides $L_1$ using $M_2$ as a “subroutine.”

Moreover, we also assume that $M_2$ decides $L_2$ in one step. We call $M_1$ a TM with oracle access to $L_2$.

Obviously, if $L_1 \leqslant_m L_2$, then $L_1 \leqslant_T L_2$. Also, if $L_1 \leqslant_T L_2$ and $L_1$ is undecidable, so is $L_2$.

2 Some variants of Halting problem

The following languages are all undecidable.

- $L_0 := \{[M] \mid L(M) = \emptyset\}$.
  That is, $[M] \in L_0$ if and only if $M$ does not accept any word.

- $L_1 := \{[M] \mid L(M) = \{0, 1\}^*\}$.
  That is, $[M] \in L_1$ if and only if $M$ accepts every word.

- $L_2 := \{[M] \mid M$ accepts the empty word $\epsilon\}$
  That is, $[M] \in L_2$ if and only if $M$ accepts the empty word $\epsilon$.

- $L_3 := \{[M] \mid M$ accepts the word $1101\}$.

- $L_4 := \{[M] \mid L(M) = \{a^n b^n \mid n \geq 0\}\}$.

- $L_5 := \{[M] \mid L(M)$ is a regular language\}.
3 Some undecidable problems concerning CFL

We consider the following three problems defined below.

<table>
<thead>
<tr>
<th>Problem</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CFL-Intersection</td>
<td>Input: Two CFG’s $G_1 = (\Sigma, V_1, R, S)$ and $G_2 = (\Sigma, V_2, R_2, S_2)$. Task: Output True, if $L(G_1) \cap L(G_2) \neq \emptyset$. Otherwise, output False.</td>
</tr>
<tr>
<td>CFL-Universality</td>
<td>Input: A CFG $G = (\Sigma, V, R, S)$. Task: Output True, if $L(G) = \Sigma^*$. Otherwise, output False.</td>
</tr>
<tr>
<td>CFL-Subset</td>
<td>Input: Two CFG’s $G_1$ and $G_2$. Task: Output True, if $L(G_1) \subseteq L(G_2)$. Otherwise, output False.</td>
</tr>
</tbody>
</table>

**Theorem 9.3** All the problems above, CFL-Intersection, CFL-Universality and CFL-Subset, are undecidable.