5.4 – Undecidable Problems

- Halting Problems (revisit)
- Reductions
- Problems on r.e. sets w/ non-trivial properties: Rice’s Theorem

Universal Turing Machine
Encoding of TM’s: <M>

**Halting Problem:**

\[ K = \{ <M> | M \text{ is a TM and } M \text{ halts on the input } <M> \} \]  
(K is denoted by \( H_0 \) in the textbook)

**Theorem:** K is undecidable, but K is Turing acceptable.

Let \( HALT = \{ <M><x> | M \text{ is a TM and } M \text{ halts on the input } x \} \)

- We could also use \(<M,x>\) to denote \(<M><x>\).
- We also use \( H \) to denote \( HALT \).

We’ll show that \( H \) is undecidable. We’re going to use “reduction” to prove this result.

**Reduction:**

Suppose we have two languages A and B. If there is a Turing-computable function \( f \) (i.e., \( f \) can be computed by a TM \( M \) on all inputs & \( M \) always halts) such that \( \forall x : x \in A \iff f(x) \in B \), then we say that A is reducible to B, written as \( A \leq_m B \)

**Proposition 1:** If \( A \leq_m B \) and B is decidable, then A is decidable.

Proof: Assume that \( A \leq_m B \) via a reduction \( f \), then we know that \( f \) is Turing-computable, and \( \forall x : x \in A \iff f(x) \in B \). If B is decidable, then \( \exists \) a DTM \( M_B \) that decides B.

i.e., \( M_B \) on any input always halts and
- If $x \in B$, then $M_B$ accepts $x$
- If $x \not\in B$, then $M_B$ rejects $x$

Let $M_f$ be a TM that computes $f$. Construct a DTM $M_A$ to decide $A$ as follows:

$$f(x) = M_B(M_f(x))$$

This means on any input $x$, $M_A$ first simulates $M_f$ on $x$. Then, $M_A$ simulates $M_B$ on the output $f(x)$ of $M_f$ on $x$.

- Since $M_f$ on any input $x$ always halts, and $M_B$ on any input always halts, we know that $M_A$ on any input will always halt.

- Since $M_B$ has two halting states, $h_a$ and $h_r$, $M_A$ will also have two halting states.

Now $\forall x$, if $x \in A$, then $f(x) \in B$. Hence, $M_B$ accepts $f(x)$.

- Since $M_f(x) = f(x)$, we have: if $x \in A$, then $M_B$ accepts $M_f(x)$

This means that $M_A$ accepts $x$. Similarly, if $x \not\in A$, then $f(x) \not\in B$. Hence, $M_B$ rejects $f(x)$. This means that $M_A$ rejects $x$. Thus, $M_A$ decides $A$, so $A$ is decidable. \textbf{End of proof.}

\textit{Corollary.} If $A \leq_m B$ and $A$ is not decidable, then $B$ is not decidable.

Now we’re ready to show that:

$$H = \{ <M><x> | M \text{ is a TM & halts on } x \}$$

is not decidable by reducing $H_0$ to $H$.

We can construct this reduction as follows:

On any instance $<M>$ of $H_0$, define $f(<M>) = <M><M>$

Then it’s easy to see that $f$ is Turing-computable (a machine that duplicates its input).

And we have:

$$\forall <M>: <M> \in H_0 \iff M \text{ is a TM } \& M \text{ halts on } <M>$$

$$\iff <M><M> \in H$$

$$\iff f(<M>) \in H$$

Hence, we know that $H_0 \leq_m H$. Thus, $H$ is not decidable. \textbf{End of proof.}
**Proposition 2**: If $A \leq_m B$ and $B$ is Turing-acceptable, then $A$ is Turing-acceptable.

**Proof**: Since $A \leq_m B$, there is a Turing-computable reduction $f$, s.t. $\forall x : x \in A \iff f(x) \in B$. Let $M_f$ be a DTM that computes $f$. This means that on any input $x$, $M_f$ always halts and produces $f(x)$ as its output.

Assume that $B$ is Turing-acceptable. Then, there is a DTM $M_B$, s.t. $B = L(M_B)$. This means that for all input $x$, if $x \in B$, then $M_B$ on $x$ halts, and $x \notin B$, then $M_B$ on $x$ never halts.

Now we construct an acceptor for $A$ as follows:

$$M_A(x) = M_B(M_f(x))$$

On any input $x$, if $x \in A$, then $f(x) \in B$. Hence, $M_B$ on the input $f(x)$ halts. Since $f(x) = M_f(x)$, $M_B$ halts on $M_f(x)$. Thus, if $x \in A$, then $M_A$ on $x$ halts. On the other hand, if $x \notin A$, then $f(x) \notin B$. Hence, $M_B$ on the input $f(x)$ never halts. This implies that $M_A$ on $x$ never halts. Thus, $M_A$ is indeed an acceptor for $A$. So $A$ is Turing-acceptable. **End of proof.**

**Corollary**: If $A \leq_m B$ and $A$ is not recursively enumerable (same as Turing-acceptable), then $B$ is not recursively enumerable.

- This means that we have a way to show that a language is not Turing-acceptable.

For instance, we know that $\overline{H}$ is not Turing-acceptable (since $H$ is Turing-acceptable, but is not decidable).

If we can reduce $\overline{H}$ to a language $L$, then $L$ is not Turing-acceptable.

**Proposition 3**: $A \leq_m B$ iff $\overline{A} \leq_m \overline{B}$. (**Proof by Def.**)

**Example**:

Let $L = \{<M_1><M_2> | M_1$ and $M_2$ are TM’s and $L(M_1) \neq L(M_2)$

Then $L$ is not Turing-acceptable.

**Proof**: Reduce $H$ to $\overline{L}$ as follows:
Construct a reduction \( f \), s.t. on instance (input) of \( H \langle M \rangle \langle x \rangle \), \( f \) outputs two TM’s \( M_1 \) and \( M_2 \),

where \( M_1 \) accepts everything:

\[
\text{i.e., } L(M_1) = \sum^* ,
\]

and \( M_2 \) on input \( w \), will simulate \( M \) on \( x \). If \( M \) halts on \( x \), then \( M_2 \) halts on \( w \):

\[
\text{i.e., if } M \text{ on } x \text{ halts, then } L(M_2) = \sum^* .
\]

This means that \( f(\langle M \rangle \langle x \rangle) = \langle M_1 \rangle \langle M_2 \rangle \) and

\[
\langle M \rangle \langle x \rangle \in H \iff L(M_1) = \sum^* \text{ and } L(M_2) = \sum^* \\
\iff \langle M_1 \rangle \langle M_2 \rangle \in \overline{L}
\]

Hence, \( H \leq_m \overline{L} \). Hence, \( \overline{H} \leq_m L \).

This implies that \( L \) is not Turing-acceptable. \( \text{(End Example)} \)

**Problems on r.e. sets w/ non-trivial properties: Rice’s Theorem:**

We’ll call a set of r.e. sets a property. E.g.,

**Non-Trivial Properties:**

\( \{ \emptyset \} \) is the property of r.e. sets “being empty”

\( \{ \sum^* \} \) is the property of “being full”

\{regular languages\} is the property of “being regular”

For any given property \( P \), we consider languages

\[
L_P = \{ \langle M \rangle \mid M \text{ is a TM } \& L(M) \in P \}
\]

E.g.,

\[
E = \{ \langle M \rangle \mid M \text{ is a TM } \& L(M) = \emptyset \} \\
F = \{ \langle M \rangle \mid M \text{ is a TM } \& L(M) = \sum^* \} \\
R = \{ \langle M \rangle \mid M \text{ is a TM } \& L(M) \text{ is regular} \}
\]

- All of these languages are not decidable.

Say a property \( P \) is trivial if \( P = \emptyset \) or \( P \) contains all r.e. sets.

**Rice’s Theorem:** For any non-trivial property \( P \), \( L_P \) is not decidable.

- As a direct application of Rice’s Theorem, we know that \( E, F, R \) are not decidable.