Lecture 9
Intro to Theory of Computation

So far:

$A_{TM}$ is Turing-recognizable but not decidable

$A_{TM}$ is not Turing-recognizable

$\text{HALT}_{TM} = \{<M,w> : TM M halts on input w\}$

Thus $\text{HALT}_{TM}$ is undecidable

Proof sketch: Show that if $\text{FM R}$ deciding $\text{HALT}_{TM}$ can build a TM $S$ for $A_{TM}$

S: on input $<M>$

- Run $R$ on input $<M>$
- If $R$ rejects, then reject
- If $R$ accepts then run $U$ on $<M>$ and output its answer

$E_{TM} = \{<M> : M \text{ is a TM with } L(M) = \emptyset\}$

Thus $E_{TM}$ is undecidable

Proof sketch: Suppose $E_{TM}$ is a TM $E$ deciding $E_{TM}$

We build a new TM $F$ that decides $A_{TM}$ as follows:
On input \(<M,w>\):
- Produce new \(<M_w>\) where TM \(M_w\) erases its input and replaces it with \(w\).
- Run \(E\) on input \(<M_w>\).
  - If \(E\) accepts then reject.
  - If \(E\) rejects then accept.

Note: \(L(M_w) = \begin{cases} \emptyset & \text{if } M \text{ does not accept } w \\ \Sigma^* & \text{if } M \text{ accepts } w \end{cases}\) (\(M_w\) behaves the same on all inputs)

Correctness:

\[
\langle M, w \rangle \in \text{ATM} \iff L(M_w) \neq \emptyset \iff L(M_w) \neq E(TM)
\]

The answer that \(E\) gives on input \(<M_w>\) is correct.

Each of these proofs took a (suggested) algorithm for one problem and showed how to produce an algorithm for another (in this case \(\text{ATM}\)).

Previously, we saw something similar where we took an algorithm for \(\text{ADFA}\) and used it to solve \(\text{AFA}\) and then put one in turn was used to get an algo for \(\text{AF}\) (in that case there algorithms existed).
To define this, we need to talk about TMs computing string functions.

**Definition:** A TM $M$ computes a function $f: \Sigma^* \rightarrow \Sigma^*$ iff for all $w \in \Sigma^*$ given to $M$ as input, $M$ halts on the 1st cell of the tape with $f(w)$ on the tape followed by all blanks.

**Definition:** We say that $f$ is computable iff there is some TM that computes it.

**Mapping Reduction:**

**Definition:** Given $A, B \subseteq \Sigma^*$, we say that $A$ is mapping reducible to $B$, $A \leq_m B$, if there is a computable function $f: \Sigma^* \rightarrow \Sigma^*$ such that $w \in A \iff f(w) \in B$.

**Notation:** $A \leq_m B$

**Idea:** Roughly: $A$ is (roughly) as easy as $B$, $B$ is (roughly) as hard as $A$.

**Then:** If $A \leq_m B$ and $B$ is decidable, $A$ is decidable.

**Proof:** Since $B$ is decidable, there is a...
decider TM \( M_B \) for \( B \)

Since \( A \equiv \text{m} B \) there is a function \( f \) computable by some TM \( M_f \) s.t. 
\[ \forall w \in \Sigma^* \quad w \in A \iff f(w) \in B. \]

We build a decider \( M_A \) for \( A \) as follows:

\[ w \xrightarrow{w} M_A \xrightarrow{f(w)} M_B \]

correctness follows directly:
\[ w \in A \iff f(w) \in B \iff M_B \text{ accepts } f(w) \iff M_A \text{ accepts } w \]
\[ w \notin A \iff f(w) \notin B \iff M_B \text{ rejects } f(w) \iff M_A \text{ rejects } w \]

Thus if \( A \equiv \text{m} B \) and \( B \) is \( T \)-rec

then \( A \) is \( T \)-rec

Proof The same construction as the case for decidability.

Proof of correctness is the same except we replace "reject" by "doesn't accept".

For if \( A \equiv \text{m} B \) and \( A \) is not decidable

then \( B \) is not decidable

(Similarly for not \( T \)-rec.)
Then HALT_M is undecidable

2nd proof: Claim: \( A_M \leq_m \text{HALT}_M \)

Want: \( <M,w> \rightarrow <M',w> \)

\[<M,w> \text{HALT} \iff <M',w> \text{HALT} \]

i.e. \( M \) accepts \( w \) \( \Rightarrow \) \( M' \) halts on input \( w \)

Design \( M' \): Just like \( M \) except

map \( <M,w> \rightarrow <M',w> \)

could be computable

Correctness:

\[<M',w> \text{HALT} \iff M' \text{halts on input } w \]

\[\iff M' \text{accepts } w \]

\[\iff M \text{accepts } w \]

Example: Consider the proof that \( E_{TM} \) was undecidable.

we actually showed that

\[ A_M \leq_m \overline{E_{TM}} \]

\[<M,w> \rightarrow <M,w> \]

\[\iff A_M \text{ undecidable } \Rightarrow E_{TM} \text{ undecidable} \]

\[\Rightarrow \]
This also means $\overline{ATM} \leq E_{TM}$

$\therefore E_{TM}$ is not Turing-recognizable

Another problem:

$EQ_{TM} = \{ <M_1, M_2> : M_1, M_2 \text{ are TM} \}
\land L(M_1) = L(M_2) \}$

Thus $EQ_{TM}$ is undecidable

Proof: Claim $E_{TM} \leq_m EQ_{TM}$

Want $<M> \xrightarrow{f} <M_1, M_2>
\text{ st. } L(M) \neq \emptyset \iff L(M_1) = L(M_2)$

Idea: Let $M_1 = M$

and $M_2 = M_{\emptyset}$ a simple TM that rejects all strings

ie. $L(M_2) = \emptyset$
\[ \langle M \rangle \mapsto \langle M, M^\phi \rangle \]

computable

\[ \langle M \rangle \in \text{E}_{\text{tm}} \iff L(M) = \phi \iff L(M) = L(M^\phi) \iff \langle M, M^\phi \rangle \in \text{E}_{\text{O}_{\text{tm}}} \]

\[ \therefore \text{E}_{\text{tm}} \subseteq \text{EO}_{\text{tm}} \]