Last time: TMs for specific languages

Two views of a TM:

(1) hardware:
   a TM for a specific task
   is built in hardware
   task 1 → TM 1
   task 2 → TM 2

(2) software:
   a TM description is a program
   (in the TM programming language),
   and any such program can be
   executed on some fixed "universal"
   hardware

Turing favoured the "software" point of view!
In his 1936 paper, he showed the existence
of a Universal Turing Machine (UTM)

\[ U(\langle M, w \rangle) \text{ simulates TM } M \text{ on input } w \text{, accepting } w \text{ iff } M \text{ accepts } w \]
<M, w> is an encoding of a TM M and a string w (input to M). Any fixed (natural) encoding will do.

Remarks:
(1) Turing's original description of a UTM in his 1936 paper had bugs (the first program bugs in history!), but these can be easily corrected. So a UTM does exist.

(2) A simple UTM U will simulate any given TM M in time O(t_M^2), where t_M is the runtime of M. A more sophisticated (2-tape) UTM is known that achieves the simulation time O(t_M \cdot \log t_M).

(3) Turing's UTM was the first compiler, & was the birth of a general purpose computer we have
A general purpose computer we have today!

**Turing's Motivation:** Show some problems cannot have algorithms, i.e., are not computable solvable.

**\( \Sigma^* \) (decidable language):**
A language \( L \subseteq \Sigma^* \) is called **decidable** if there exists a TM \( M \) s.t.
1. \( M \) halts on every input \( x \in \Sigma^* \),
2. \( M \) accepts every \( x \in L \), and \( M \) rejects every \( x \notin L \).

In general, a TM on a given input may not terminate (may enter an infinite loop).

**\( \Sigma^* \) (a decider):**
A TM \( M \) is called a **decider** if \( M \) halts on every input.

**Halting Problem:**
**Halting Problem:**

**Given:** TM M

**Decide:** Does M halt on ε?

**Def (semi-decidable language):**
A language $L \subseteq \Sigma^*$ is **semi-decidable** if $\exists$ TM M s.t.

1. $\forall x \in \Sigma^*$, $x \in L \Rightarrow M$ on $x$ **accepts**

2. $\forall x \notin \Sigma^*$, $x \notin L \Rightarrow M$ on $x$ **does not accept**

**Claim:** Halting Problem is **semi-decidable**.

**Proof:** Given $\langle M, \varepsilon \rangle$

want to (1) accept if $M$ accepts $\varepsilon$

(2) not accept if $M$ does not accept $\varepsilon$.

Follows from UTM! 

**Claim:** Halting Problem is **not**
Claim: Halting Problem is not decidable.

Proof: Enumerate all TMs:

\[ M_1 \quad M_2 \quad M_3 \ldots \]

\[ W_1 \quad W_2 \quad W_3 \ldots \]

Define \( D = \{ <M_i> \mid \text{TM } M_i \text{ does not accept input } <M_i> \} \)

Sub-Claim: \( D \) is not semi-decidable.

Proof (by contradiction):
Suppose \( D \) is semi-decidable.

Then \( \exists \) TM \( M_i \) s.t. \( M_i \) semi-decides \( D \).

Question: What does \( M_i \) do on input \( <M_i> \)?

Answer: Either
(1) $M_i$ accepts $<M_i>$, or
(2) $M_i$ does not accept $<M_i>$.

(1) $\Rightarrow$ $<M_i> \notin \Delta$ (by defn of $\Delta$)
   $\Rightarrow M_i$ semi-deciding $\Delta$
   does not accept $<M_i>$.
   Contradiction.

(2) $\Rightarrow$ $<M_i> \in \Delta$ (by defn of $\Delta$)
   $\Rightarrow M_i$ does accept $<M_i>$
   Again, contradiction.

We conclude $\Delta$ is not semi-decidable. $\Box$

Sub-claim 2:
If Halting Problem is decidable,
then $\Delta$ is decidable.

Proof:

$<M_i>$: Want to decide
if $M_i$ accepts $<M_i>$.
Assume can decide
Need to argue:

\[ \text{Halt} : \text{Given } M, \text{ decide if } M \text{ halts on } e. \]

\[ \text{Halt'} : \text{Given } (M, w), \text{ decide if } M \text{ halts on } w. \]

Are equivalent!

That is, Halt is decidable if Halt' is.
Exercise:  Show that $\text{Halt} \& \text{Halt}'$ are equivalent w.r.t. decidability!