Lecture 8

Monday, May 27, 2019 12:28 PM

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)

$U(\langle M, w \rangle)$ simulates TM M on input w, accepting iff M accepts w

input w, accepting iff M accepts w.

$\langle M, w \rangle$ is an <u>encoding</u> of a TM M and a string w (input to M). Any fixed (natural) encoding will do.

<u>Remarks</u>:

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

(2) A <u>simple</u> 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 <u>general purpose computer</u> we have

a general purpose computer we have today!

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

Def (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 $x \in L$, &
M rejects every $x \notin L$.

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

Def (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) $\forall x \in \Sigma^*$, $x \notin L \Rightarrow M$ on x does not accept

Claim: Halting Problem is semi-decidable.

Proof: Given $\langle M, \varepsilon \rangle$, want to (1) accept if M accepts ε

(2) not accept if M does not accept ε.

Follows from UTM! $\square$

Claim: Halting Problem is not

Claim: Halting Problem is <u>NOT</u> decidable.

Proof: Enumerate all TMs:

Define $D = \{ \langle M_i \rangle \mid \text{TM } M_i \text{ does } \underline{\text{not}} \text{ accept input } \langle M_i \rangle \}$

<u>Sub-Claim:</u> D is not semi-decidable.

<u>Proof (by contradiction):</u>

Suppose D is semi-decidable.

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

<u>Question:</u> What does M_i do on input $\langle M_i \rangle$?

<u>Answer:</u> Either

(1) M_i accepts $\langle M_i \rangle$, or

(2) M_i does not accept $\langle M_i \rangle$.

(1) $\Rightarrow$ $\langle M_i \rangle \notin \mathcal{D}$ (by defn of $\mathcal{D}$)

$\Rightarrow$ M_i semi-deciding $\mathcal{D}$ does <u>not</u> accept $\langle M_i \rangle$.

Contradiction.

(2) $\Rightarrow$ $\langle M_i \rangle \in \mathcal{D}$ (by defn of $\mathcal{D}$)

$\Rightarrow$ M_i does accept $\langle M_i \rangle$

Again, contradiction.

We conclude $\mathcal{D}$ is <u>not</u> semi-decidable. $\square$

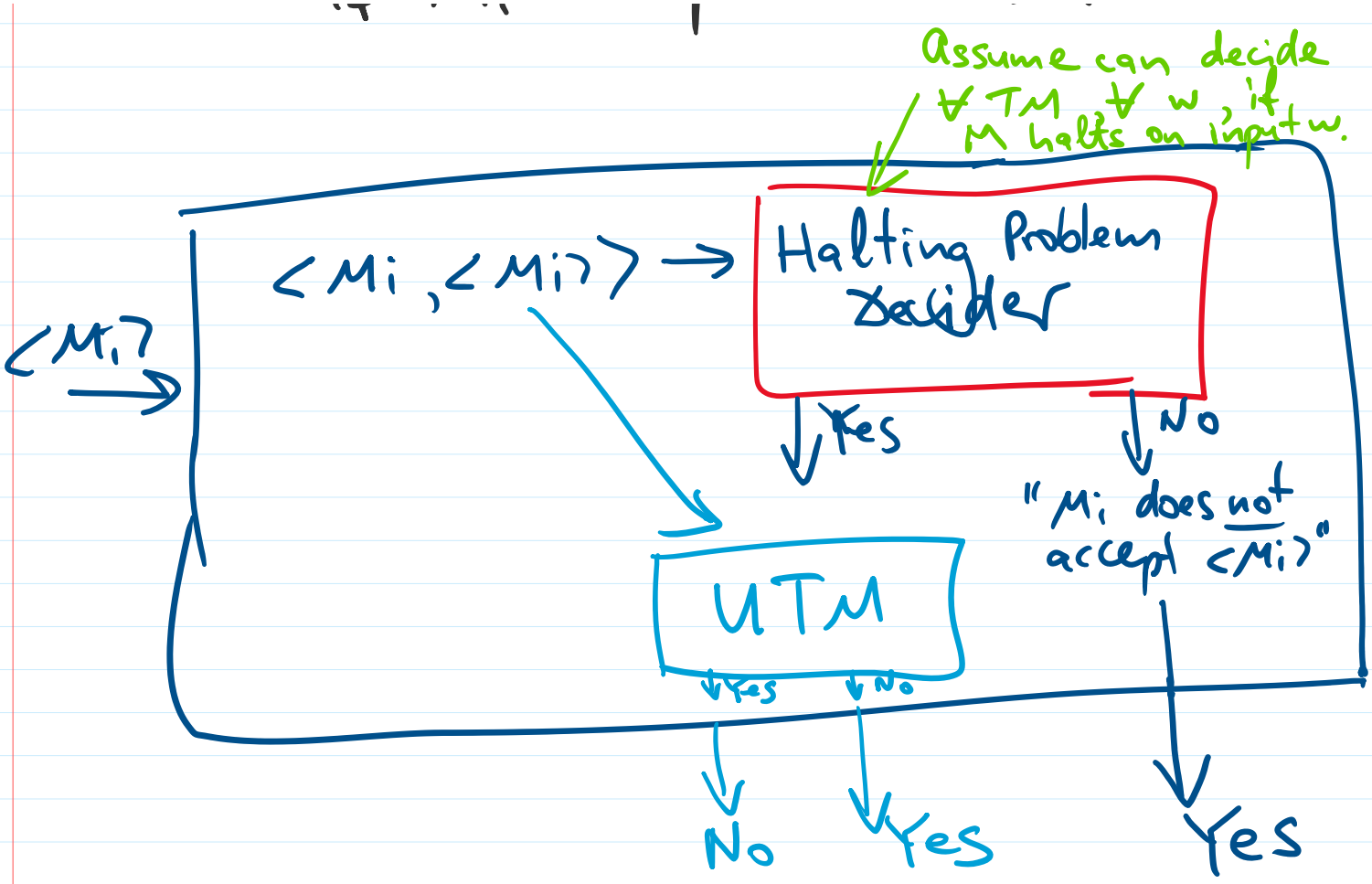
<u>Sub-claim 2:</u>

If Halting Problem is decidable, then $\mathcal{D}$ is decidable.

<u>Proof:</u>

$\langle M_i \rangle$: Want to decide if M_i accepts $\langle M_i \rangle$.

Assume can decide



Need to argue :

<u>Halt</u> : Given M, decide if M halts on ε.

&

<u>Halt'</u> : Given (M, w), decide if M halts on w.

are equivalent!

That is, Halt is decidable iff Halt' is.

Exercise: Show Halt & Halt' are

<u>Exercise</u>: Show Halt & Halt' are equivalent w.r.t. decidability!