18 April 2008

Announcements. ① Lunch @ Matthh's Fridays @ noon w/ Prof. Kleinberg. Sign up at front of room.

② My office hours 11–12 this Thurs. (Gates 317)

TODAY Undecidability

$M(x)$ refers to halting status of M on input x: yes, no, halt, ↗

"M accepts x" means $M(x) = \text{yes}$.

$L(M) := \{x \mid M \text{ accepts } x\}$.

Def. $L \subseteq \Sigma^*$ is r.e. if $\exists M$ s.t. $(M \text{ accepts } x) \iff (x \in L)$.

$L \subseteq \Sigma^*$ is decidable if $\exists M$ s.t. $x \in L \Rightarrow M(x) = \text{yes}$, $x \notin L \Rightarrow M(x) = \text{no}$.

Lemma $\forall L \subseteq \Sigma^*$, L is decidable if and only if L is r.e. and $\overline{L}$ is r.e. ($\overline{L}$: complement of L in Σ^*)

Proof. ($\Rightarrow$) If L is decidable then

$\exists M$ s.t. $x \in L \Rightarrow M(x) = \text{yes}$, $x \notin L \Rightarrow M(x) = \text{no}$.

So $L = L(M)$, i.e. L is r.e.

Take M and reverse its outputs. (Change transition function so that "yes" becomes "no" & vice-versa.)

Call the modified machine $\overline{M}$. Then

$L(\overline{M}) = \overline{L} \Rightarrow \overline{L}$ is r.e.

($\Leftarrow$) If L and $\overline{L}$ are r.e., let M and $\overline{M}$ be Turing machines such that $L = L(M)$, $\overline{L} = L(\overline{M})$.

Let M^* be a two-tape Turing machine with state set $\{\text{states of } M\} \times \{\text{states of } \overline{M}\}$.

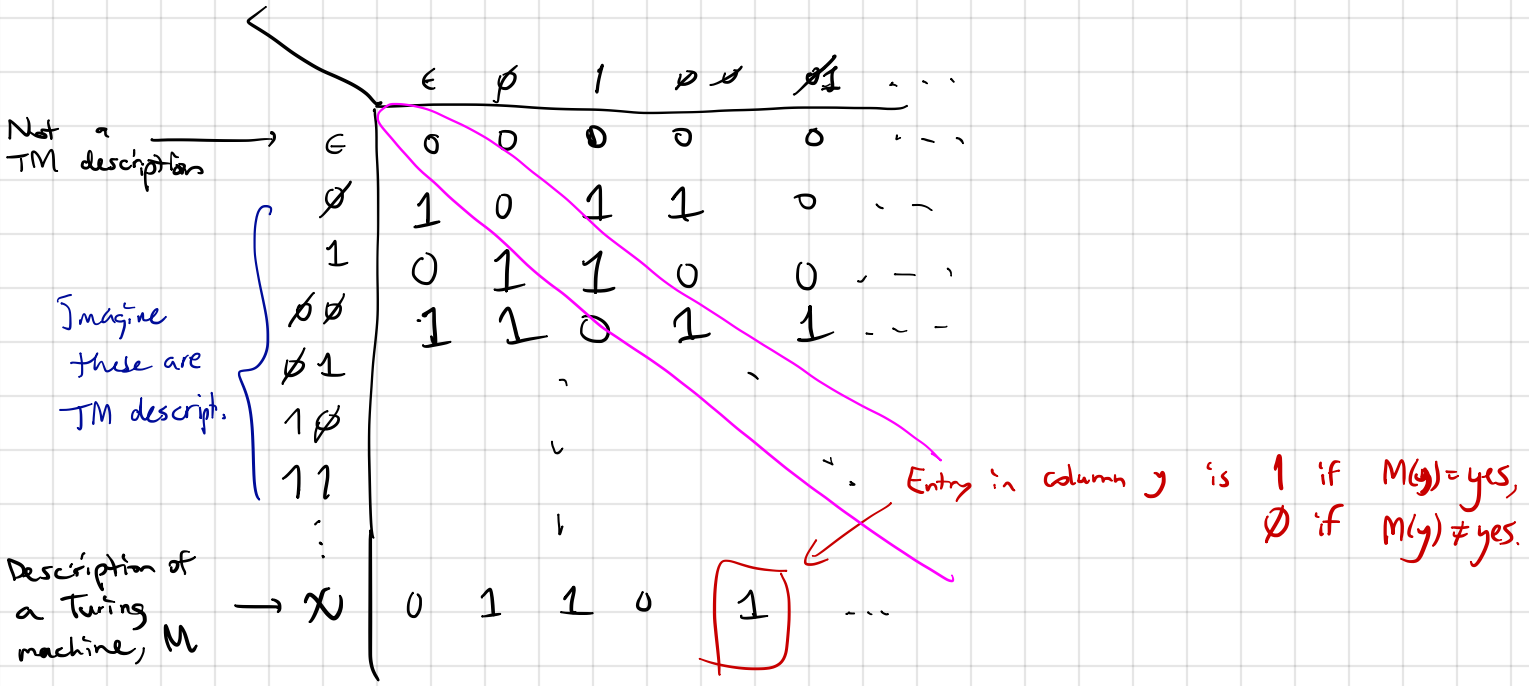
M^* first copies its input onto 2nd tape.

Then runs M and $\overline{M}$ in parallel each on one of the tapes.

When $M \to$ yes, $M^* \to$ yes. When $\overline{M} \to$ yes, $M^* \to$ no.

Constructing an undecidable problem: a set $D \subseteq \Sigma^*$ such that no Turing machine can decide D.

Construction is based on diagonalization. Consider a table with infinitely many rows, columns, indexed by Σ^*.



Let D be the set of strings x s.t. the table contains a 0 in location (x,x).

$$D = \{x \mid x \text{ is not a Turing machine description}\}$$
$$\cup \{x \mid x \text{ is a description of a Turing machine, } M, \text{ s.t. } M(x) \neq \text{yes}\}$$

D is not r.e. because if $D = L(M)$ let x be a description of M. Either $x \in D$ or $x \notin D$. But if $x \in D$ it means $x \in L(M)$, i.e. $M(x) = \text{yes}$, contradicting $x \in D$. And if $x \notin D$ it means $x \notin L(M)$, i.e. $M(x) \neq \text{yes}$, contradicting $x \notin D$. The contradiction must stem from the hypothesis $D = L(M)$. So D is not r.e.

The Halting Problem. $H = \{x;y \mid U(x;y) \neq \nearrow\}$.

$= \{x;y \mid x \text{ describes a machine } M \text{ that halts on input } y\}$

End of last lecture: H is r.e. Today: H is not decidable.

To show H not decidable we must show $\overline{H}$ not r.e.

Idea: Show $\overline{H}$ not r.e. by reduction, $D \leq \overline{H}$.

Notation means: assuming $M_{\overline{H}}$ is a Turing machine s.t. $\overline{H} = L(M_{\overline{H}})$ we use $M_{\overline{H}}$ to construct M_D such that $D = L(M_D)$.

Pseudocode for M_D: Given input string x.

if $M_{\overline{H}}(x;x) =$ yes ← could possibly spend forever on this line, running $M_{\overline{H}}$.

output yes

else

output { yes if $U(x;x) \neq$ yes
no otherwise

If $M_D(x) =$ yes it means either $M_{\overline{H}}(x;x) =$ yes $\Rightarrow x;x \notin H \Rightarrow$ Machine M rep'd by string x runs forever on input x

$\Rightarrow x \in D$

or $M_{\overline{H}}(x;x) \in \{n_o, \text{halt}\}$ and $U(x;x) \neq$ yes

$\Rightarrow x;x \in H \Rightarrow U(x;x)$ terminates

$U(x;x)$ terminates and $U(x;x) \neq$ yes $\Rightarrow x \in D$.

Proof abandoned for now, to be continued Friday.