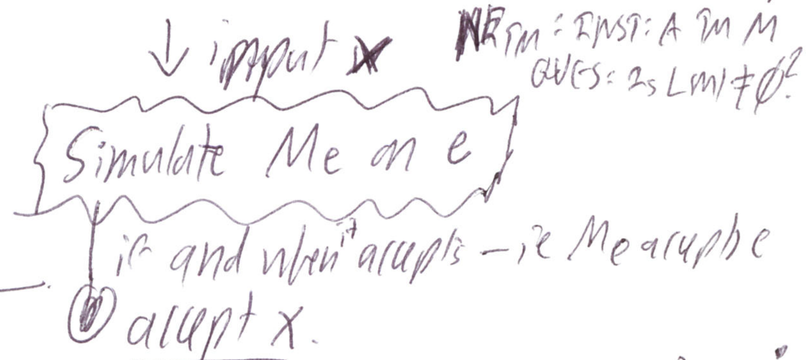
Last lecture: reduction *to* NE_{TM} can also be *from* K_{TM}

$K_{TM} = \{e : M_e \text{ accepts } e\}$ (in Gödel Number notation)
$\equiv K$

NE_{TM}: INST: A TM M
QUES: Is $L(M) \neq \emptyset$?

$f(e) = e'$ where e' is the code of the TM $M_{e'}$:

ALL-OR-NOTHING SWITCH



$e \in K \Rightarrow M_e$ accepts $e \Rightarrow$ for all x, $M_{e'}$ accepts $x \Rightarrow L(M_{e'}) = \Sigma^*$
$\Rightarrow L(M_{e'}) \neq \emptyset \Rightarrow f(e) = e' \in NE_{TM}$

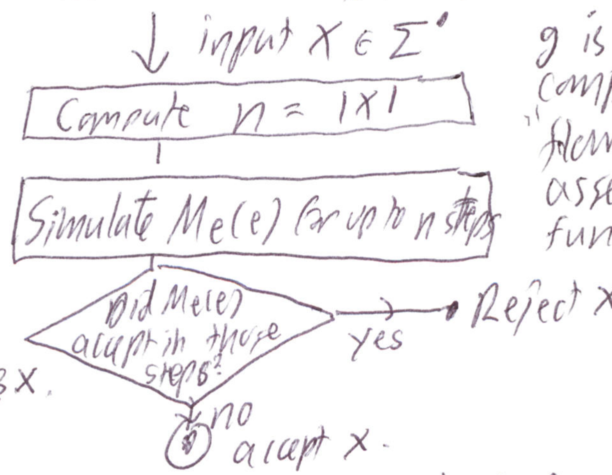
$e \notin K \Rightarrow M_e$ does not accept $e \Rightarrow$ for all x, $M_{e'}$ does not accept x
$\Rightarrow L(M_{e'}) = \emptyset \Rightarrow f(e) = e'$ st $e' \notin NE_{TM}$.

$\therefore$ $e \in K \iff f(e) \in NE_{TM}$, and since f is a computable "flowchart assemblage"
$K \leq_m NE_{TM}$, so NE_{TM} is undecidable (not co-r.e. too).

We also get $e \in K \iff L(M_{e'}) = \Sigma^*$, so $K_{TM} \leq_m ALL_{TM}$ by the same f.

"DELAY SWITCH"

$g(e) = e''$ code of $M_{e''}$ =

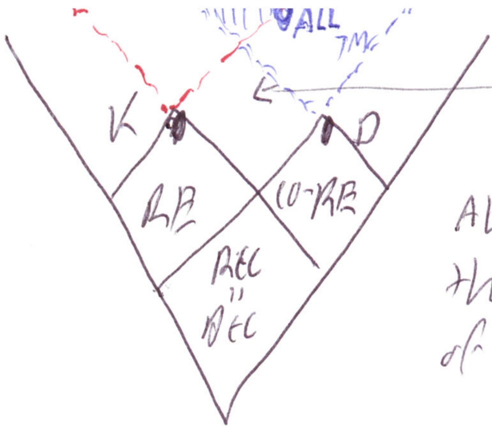


g is also a computable "flowchart assemblage" function.

$e \in K \Rightarrow M_e$ accepts $e \Rightarrow$ there is an $n_0 \geq 1$ st. M_e accepts e in n_0 steps $\Rightarrow$ for all x with $|x| \geq n_0$, $M_{e''}$ sees this and rejects x.
$\Rightarrow L(M_{e''})$ is finite $\Rightarrow L(M_{e''}) \neq \Sigma^*$

$e \notin K \Rightarrow \forall x$ $M_{e''}$ never sees $M_e(e)$ accept $\Rightarrow \forall x$ $M_{e''}$ accepts $x \Rightarrow L(M_{e''}) = \Sigma^*$.

Thus $e \in D_{TM} \iff L(M_{e''}) = \Sigma^*$, so $D_{TM} \leq_m ALL_{TM}$ via this g.
so ALL_{TM} is not r.e. either.



Neither r.e. nor co-r.e.

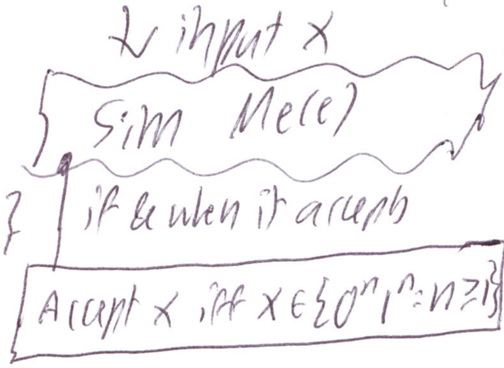
ALL_{TM} has "More than one Degree of Undecidability"

Thus the language $ALL_{TM} = \{e : L(M_e) = \Sigma^*\} = \{\langle M\rangle : L(M) = \Sigma^*\}$ (text style) is neither r.e. nor co-r.e., i.e. neither it nor its complement is Turing recognizable. ②

Both switches can be applied with further "Language Filtering"

$f' \quad e \longmapsto M_{e'} =$

$\Sigma = \{0,1\}$



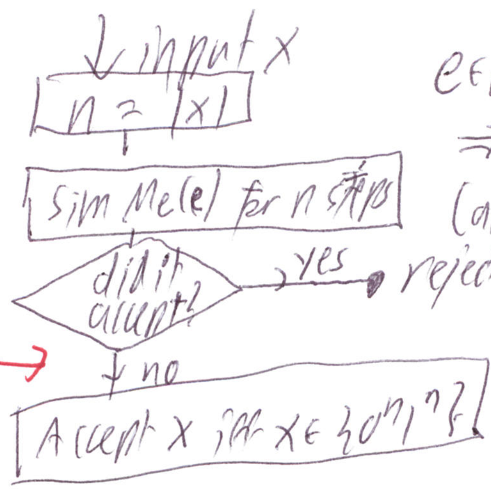
$e \in K \Rightarrow L(M_{e'}) = \{0^n1^n : n \geq 1\}$ ($e, \notin D$)

$\Rightarrow L(M_{e'})$ is nonregular.

$e \notin K \Rightarrow L(M_{e'}) = \emptyset$ ($e \in D$) which is regular, so $f(e) \in REGULAR_{TM} \stackrel{\text{def}}{=} \{\langle M\rangle : L(M) \text{ is regular}\}$.

We get $e \in D \iff f'(e) \in REGULAR_{TM}$ so $REGULAR_{TM}$ is not r.e.

$g' \quad e \longmapsto M_{e''}$



$e \in K \Rightarrow L(M_{e''})$ is finite $\Rightarrow L(M_{e''})$ is regular (all finite sets are regular) $\Rightarrow g'(e) \in REGULAR_{TM}$

$e \notin K \Rightarrow L(M_{e''}) = \{0^n1^n\} \Rightarrow g'(e) \notin REGULAR_{TM}$.

o $K \leq_m REGULAR_{TM}$
ia g' and

if $e \notin K$, ie $e \in D$ this is still letting all x through.

$D \leq_m REGULAR_{TM}$ via f'

Thus $REGULAR_{TM}$ is neither r.e. nor co-r.e. [FYI: it has 3 degrees of undecidability] This adds to what the text says in §5.1 about it.

Study Guide: What happens if we put the $x \in \{0^n1^n\}$ test first?

Note: $REGULAR_{TM}$ is not the same as your HW(3) language.

$REGULAR_{CFG} = \{\langle G\rangle : G \text{ is a CFG and } L(G) \text{ is regular}\}$.

Fact: [Text gives details in §5.1's second half. §5.2 skipped]

There are computable functions h and h' s.t. for any TM M_e:

- $h(e)$ is a CFG G such that $L(G) = \Sigma^* \iff M_e \in E_{TM}$ i.e. $L(M_e) = \emptyset$ and such that if $M_e \notin E_{TM}$, $L(G)$ is not regular. (*)

- $h'(e)$ is a pair $\langle M_1, M_2 \rangle$ of DPDAs such that $M_e \in E_{TM} \iff L(M_1) \cap L(M_2) = \emptyset$. over all inputs x.

G generates the set of all "broken computations" of $M_e(x)$
A computation to be valid is like $x \# x' \# x'' \# x''' \ldots$ where the strings between the hashes are mostly equal $\approx$ DOUBLE WORD.
The complement $\sim\{ww : w \in \{0,1\}^*\}$ is a CFL. $L(G) \approx \Sigma^* \# \{ww\} \# \Sigma^*$

$h'(e)$ comes from writing comp's as $\underbrace{x \# x'^R}\ \# \underbrace{x'' \# x'''^R}\ \# x'''' \ldots$
M_1 checks
M_2 checks

h reduces E_{TM} to ALL_{CFG}, so ALL_{CFG} is undecidable.
h' reduces E_{TM} to $\{\langle G_1, G_2 \rangle : L(G_1) \cap L(G_2) = \emptyset\}$ (**)
Contrast with E_{CFG} being decidable.

(**) And when $L(G_1) \cap L(G_2) \neq \emptyset$, it is rigged to be infinite and like the language of palindromes

Complexity Theory, NP Completeness $\Rightarrow$ Thurs.

text: "computation histories".

(*) To wit, $L(G) = \{$all strings that do not have substrings that code valid accepting computations of M_e on some $x\}$
If $L(M_e) = \emptyset$, then these are all strings, period, so $L(G) = \Sigma^*$. But if $L(M_e) \neq \emptyset$, then $L(G)$ not only skips the string code of some accepting computation $\vec{z}$, it skips it in a way that creates a non-regular language.