Lecture 7
Reduction

“A is reducible to B” means I could solve A if I had a subroutine for B

Ex:
Finding the max element in a list is reducible to sorting
pf: sort the list in increasing order, take the last element
(A big hammer for a small problem, but never mind...)
The Halting Problem

\[ \text{HALT}_{TM} = \{ <M,w> \mid \text{TM } M \text{ halts on input } w \} \]

Theorem: The halting problem is undecidable

Proof:

A = A_{TM}, B = \text{HALT}_{TM} Suppose I can reduce A to B. We already know A is undecidable, so must be that B is, too.

Suppose TM R decides \text{HALT}_{TM}. Consider S:

On input \(<M,w>\), run R on it. If it rejects, halt & reject; if it accepts, run M on w; accept/reject as it does.

Then S decides A_{TM}, which is impossible. R can’t exist.
Another Way

Rather than running $R$ on $<M,w>$, and manipulating that answer, manipulate the input to build a new $M'$ so that $R$'s answer about $<M',w>$ *directly* answers the question of interest.

Specifically, build $M'$ as a clone of $M$, but modified so that if $M$ halts-and-rejects, $M'$ instead rejects by looping.

Then halt/not-halt for $M'$ == accept/reject for $M$

Again, this reduces $A_{TM}$ to $HALT_{TM}$
S':

Build $M'$
Pass $<M',w>$ to $R$

R:

Halt?

Yes

M': same as $M$, but $q_{reject}$ replaced by a loop

rej

acc
Reduction

Notation (not in book, but common):

\[ A \leq_T B \] means “A is Turing Reducible to B”

I.e., if I had a TM deciding B, I could use it as a subroutine to solve A

Facts:

- \[ A \leq_T B \& \& B \text{ decidable implies } A \text{ decidable} \] (definition)
- \[ A \leq_T B \& \& \text{A undecidable implies B undecidable} \] (contrapositive)
- \[ A \leq_T B \& \& B \leq_T C \text{ implies } A \leq_T C \]
EMPTY$_{TM}$ is undecidable

EMPTY$_{TM} = \{ <M> \mid M \text{ is a TM s.t. } L(M) = \emptyset \}$

\[
\begin{align*}
&M' : \text{erase input} \\
&M' : \text{write } w \\
&M' : \text{run } M \\
&M' : \text{if } M \text{ rejects } w \\
&M' : \text{write } w \\
&M' : \text{if } M \text{ accepts } w \\
\end{align*}
\]

\[\text{ATM} \leq_T \text{EMPTY}_{TM}\]
REGULAR\textsubscript{TM} is undecidable

REGULAR\textsubscript{TM} = \{ <M> | M is a TM s.t. L(M) is regular \}

A \text{ TM} \text{ s.t. } \text{RegTM}

\text{Given: } <M, w> \text{ build } M'

That

\exists n' \text{ s.t. } n' \geq 0 \text{ and } w = 0^n1^n

M' on input w:

if w \in \{0^n1^n \mid n \geq 0\} accept

if not accept type

Write w', Simulate M on w'