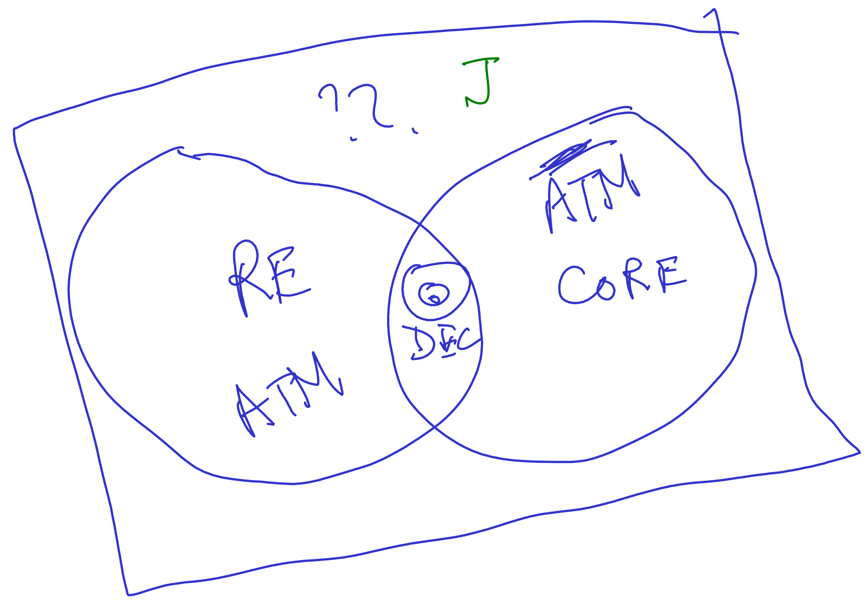
CSE322 Theory of Comput. (L22)



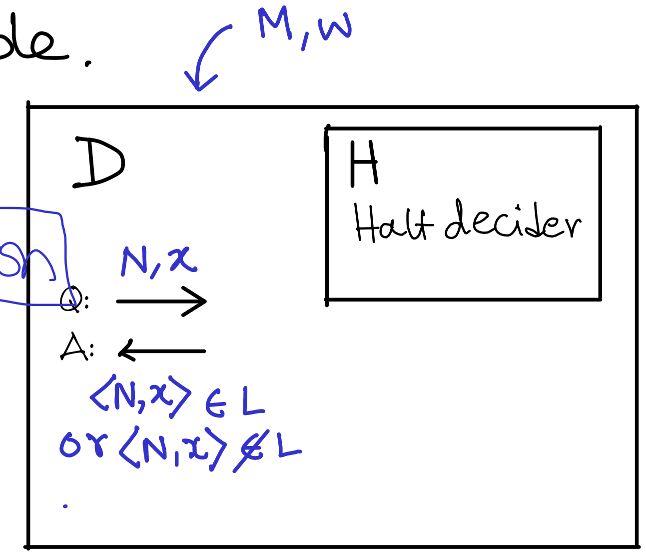
$P1 \leq P2$

(A) Construct a decider (recognizer) for P1 assuming access to a decider (recognizer) for P2

(B) Since P1 is undec (unrecog), so should be P2.

Another proof of HALT is undecidable.

A-TM $\boxed{\le}$ HALT using a Turing reduction



Level 1: Proof by contradiction.

Suppose H can decide HALT.

Construct decider D for A-TM.

D can call H on <u>any</u> ⟨N,x⟩.

(A)

D(⟨M,w⟩): // Does M accept w?

1. Run H on ⟨M,w⟩
2. If H accepts ⟨M,w⟩ // M(w) halts

   Run M on w and do what M does

3. If H rejects ⟨M,w⟩ // M(w) doesn't halt

   D rejects

Claim: D decides A-TM

(A)

D(⟨M,w⟩): // Does M accept w?

1. construct M' : copy of M but changed to loop on q-rej
2. Run H on ⟨M',w⟩
3. If H accepts ⟨M',w⟩ // M' halts on w

   D accepts

3. If H rejects ⟨M',w⟩ // M' doesn't halt on w ≡ M(w) loops or rejects

   D rejects.

(B)

Thm. If HALT is decidable, then A-TM is decidable.

Corollary. Using the undecidability of A-TM, HALT is undecidable.

Prove by Turing Reduction

$P1 \le_T P2 \quad P2 \le_T P3 \implies P1 \le_T P3$ } transitive

Given decider for X, construct decider for Y $\quad Y \le X$

Given recognizer for W, construct recognizer for Z $\quad Z \le W$

...

Useful for proving that ...

* X is undecidable (when Y is known to be undecidable)
* W is not recog. (when Z is known to be not recognizable)
* If X is decidable then Y is decidable.
*

$E\text{-}TM = \{ M : L(M) = \{\} \}$ Does this browser EVER correctly display any webpage?

From M and w (string), construct Mw s.t. <M,w> is in A-TM iff Mw is not in E-TM.

M accepts w iff $L(M_w) \neq \{\}$

Mw (z):

1. If z != w, reject
2. If z = w, simulate M on w (accept, reject accordingly)

L(Mw)={} iff M does not accept w

L(Mw)={w} iff M accepts w M_w rejects every other string

$A\text{-}TM \leq_T ETM$

A Given decider E for E-TM, create decider D for A-TM.

Q: What does this say about E-TM?

D(<M,w>):

1. Create Mw using M and w
2. Run DETM(<Mw>).
3. If DETM accepts: D rejects.
4. If DETM rejects: D accepts. $L(M_w) \neq \{\}$

Q: Given decider E for E-TM, create decider D for complement of A-TM.

Ex:- $\overline{ATM} \leq ETM$

E-TM = { M : L(M) = {} } is undeciable

Level-1: We will prove this by contradiction and reduction from A-TM. Specifically, we will construct a decider D for A-TM assuming the existence of a decider E for E-TM.

Level-2: Consider the following TM.

D(<M, w>):

1. Create encoding of a TM Mw from M and w as follows:

   Mw (z):

   1. If z != w, reject
   2. If z = w, simulate M on w (accept, reject accordingly)
2. Run E(<Mw>)
3. If E accepts <Mw>, D rejects <M, w>
4. Else, D accepts <M, w>

Claim: If M accepts w, $L(Mw) = \{w\}$. Else, $L(Mw) = \{\}$.

// Add brief justification for claim...

Claim: D is a decider for A-TM.

// Add justifications: (i) D is a decider (ii) L(D) = A-TM

Since A-TM is undecidable, E cannot exist. Hence, E-TM is undecidable.

EQ-TM = { $\langle M,N \rangle$: L(M)=L(N) }

$E_{TM} \leq_T EQ_{TM}$

Does your quicksort implementation match my implementation?

Level-1: Proof by reduction from E-TM. (equivalently, reducing E-TM to EQ-TM)

Level-2:

Suppose you have a decider for EQ-TM. That is, ... some TM (err... algorithm) which takes M1 and M2 and decides (yes/no) if L(M1) = L(M2). (How) Can you use it to decide if L(T) = {} for ANY given TM T ?

$D_{ETM}(\langle M \rangle)$:

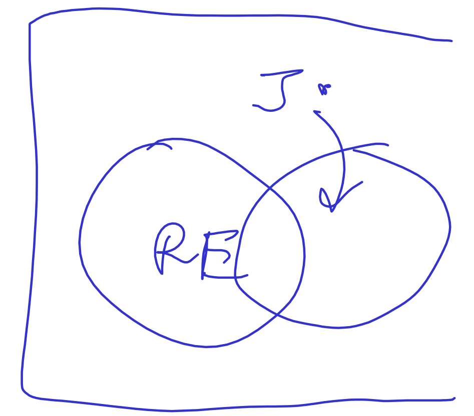
1. Construct DTM N s.t. N rejects every string L(N) = {}
2. Run $D_{EQTM}(\langle M, N \rangle)$,
3. If D_{EQTM} accepts : D_{ETM} accepts → L(M) = L(N) = {}
4. " " rejects : " rejects ↳ L(M) ≠ L(N) = {}

5.12 $J = \{ w : w=0x$ for some x in ATM or

$w=1x$ for some x in ATM-complement $\}$

Show that J is not in RE or coRE.

Assume $J \in RE$
$\Rightarrow ATM \in RE$ #



$\overline{ATM} \le_{R_J} J \Rightarrow J \notin RE$

1. Reduce ATM-complement to J. What does this prove?

$R_{\overline{ATM}}(y = \langle M, w \rangle)$: // if M accepts w, reject
" M doesn't accept w, accept.

X
Run $R_J(0y)$
If R_J accepts $0y$, → $y \in ATM$ reject
If " rejects " , accepts.

// If R_J loops on $0y \Rightarrow y$ should be accepted.

Run $R_J(1y)$
If R_J accepts, → $y \in \overline{ATM}$ $R_{\overline{ATM}}$ accepts
" " rejects " rejects.

5.12 $J = \{ w : w = 0x$ for some x in ATM or

$w = 1x$ for some x in ATM-complement $\}$

$\overline{ATM} \le \overline{J}$

Show that J is not in RE or coRE.

$\overline{J} = \{ w : 1x$ where $\cancel{x} \in ATM$
or $0x$ " $x \in \overline{ATM} \}$

Let $J \in coRE$
$\Rightarrow \overline{J} \in RE$

$ATM \le J \Rightarrow$ $\Rightarrow \overline{ATM} \in RE$
#

2. What is J-compl? How to prove that J is not in coRE?

$\therefore J \notin coRE$

Given $R_{\overline{J}}$ for $\overline{J}$, construct $R_{\overline{ATM}}$ for $\overline{ATM}$.

def $R_{\overline{ATM}}(\langle \overset{y=}{M, w} \rangle)$:

Call $R_{\overline{J}}(0y)$

If $R_{\overline{J}}(0y)$ accepts: $y \in \overline{ATM}$, $R_{\overline{ATM}}$ accepts

" rejects: rejects.