Top Hat 8680

$D_{TM} = \underline{\underline{D}} = \{\langle M\rangle : \underbrace{M \text{ does not accept}}_{\text{could mean } M(\langle M\rangle)\uparrow \text{ doesn't halt.}} \langle M\rangle\}$

Complement is

$\underline{K_{TM}} = \underline{\underline{K}} = \{\langle M\rangle : M \text{ \underline{does} accept } \langle M\rangle\}$

K is undecidable because D is undecidable. But K is c.e. because K is specially marked subset of

$A_{TM} = \{\langle M, w\rangle : M \text{ accepts } w\}$, Universal Turing Mach

which is the language of the "Turing Kit", or of any UTM.

The K_{TM} problem is the special case of the A_{TM} problem

A_{TM}: INST: A TM M and an input $w \in \Sigma^*$ to M
QUES: Does M accept w?

K_{TM} INST: Just M.
QUES: Does M accept $\langle M\rangle$? $\iff \langle M, M\rangle \in A_{TM}$

$\langle M\rangle \in K_{TM}$

Theorem 1: A_{TM} is c.e. but not decidable.

Goal: Get further such conclusions from observing that the function $f(\langle M\rangle) = \langle M, M\rangle$ satisfies the relation

$f(X) = \langle X, X\rangle$ $\quad X \in K_{TM} \iff f(x) \in A_{TM}$

(2)

Def^n: Given any languages $A, B \subseteq \Sigma^*$, we write

$A \leq_m B$ and say A mapping-/many-one reduces to B

if there is a computable function $f: \Sigma^* \to \Sigma^*$ s.t.

for all $x \in \Sigma^*$, $x \in A \iff f(x) \in B$.

Previous Example: $A = K_{TM}$, $B = A_{TM}$.

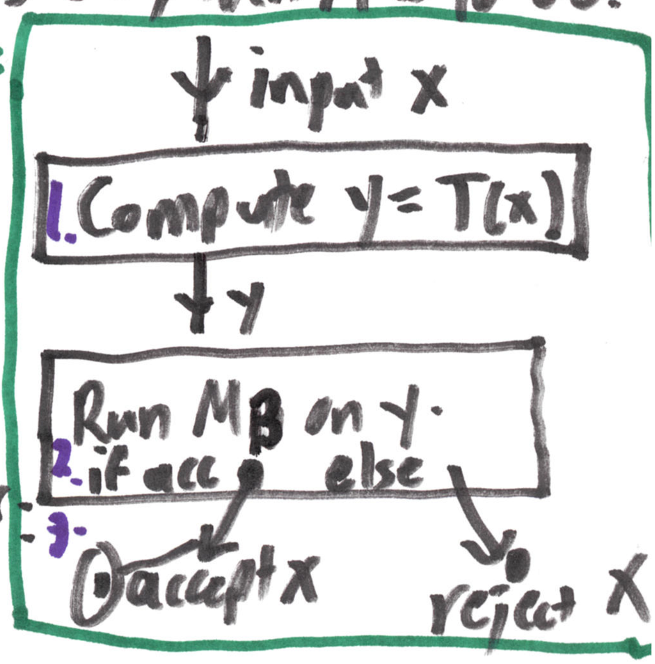
Theorem 2: Suppose $A \leq_m B$. Then

(a) If B is decidable then A is decidable

(b) If B is c.e. then A is c.e.

(c) If B is co-c.e., i.e. $\tilde{B}$ is c.e., then A is co-c.e.

M_A:



Proof: (a) Take a total TM M_B s.t. $L(M) = B$ and a total TM T computing f. View "Transducer" them as solid boxes:

Then M_A is total, and for all x:

M_A accepts $x \iff M_B$ accepts $y = f(x) \iff y \in B \iff x \in A$

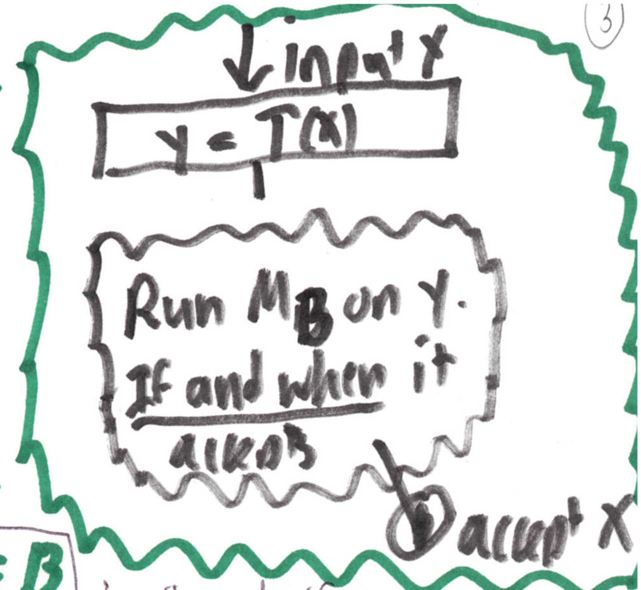
So M_A decides A. ■

by $L(M_B) = B$

by $x \in A \iff f(x) \in B$ since f is a reduction.

In (b), by B is c.e., we are only given that $L(M_B) = B$, not that M_B is total. We draw a "fuzzy box" for M_B:

M_A:



Then M_A represents executable code and for all x, $\boxed{x \in A \iff f(x) = y \in B}$ by the reduction property of f.

$\iff M_B$ accepts $y \iff M_A$ accepts x.

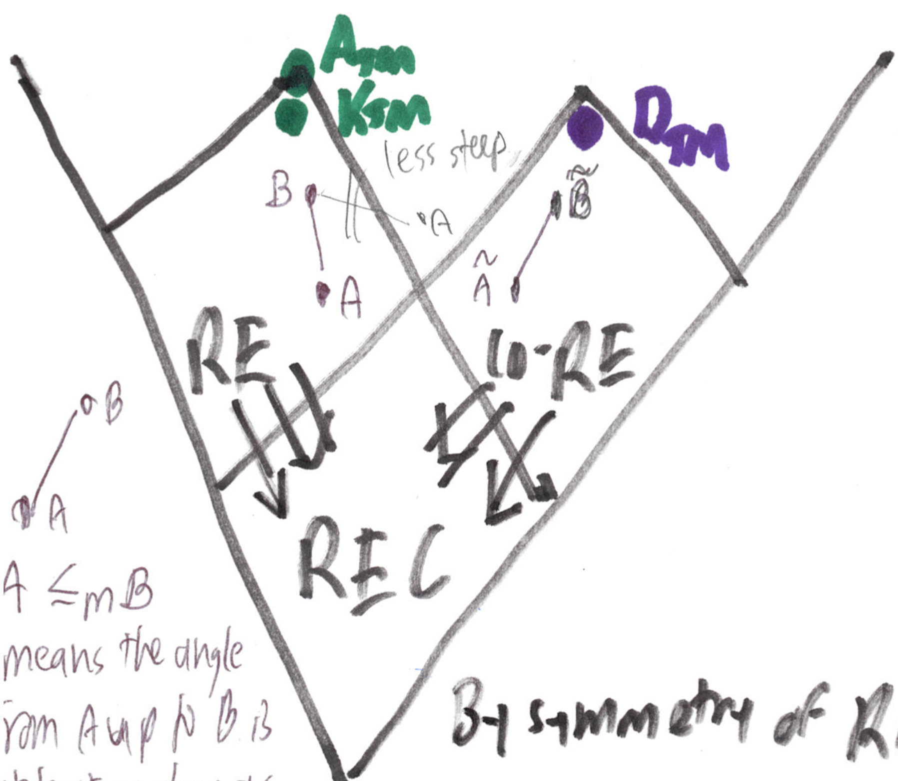
by flowchart code of M_A

$\therefore L(M_A) = A$, so A is c.e.

For (c), we have $A \leq_m B$ and B is co-c.e., ie $\tilde{B}$ is c.e.

We claim $\tilde{A} \leq_m \tilde{B}$ because

$$x \in A \iff f(x) \in B$$
$$\Updownarrow \qquad\qquad \Updownarrow$$
$$x \notin \tilde{A} \iff f(x) \notin \tilde{B}$$
$$\Updownarrow \qquad \Updownarrow \qquad \Updownarrow$$
$$x \in \tilde{A} \iff f(x) \in \tilde{B}.$$



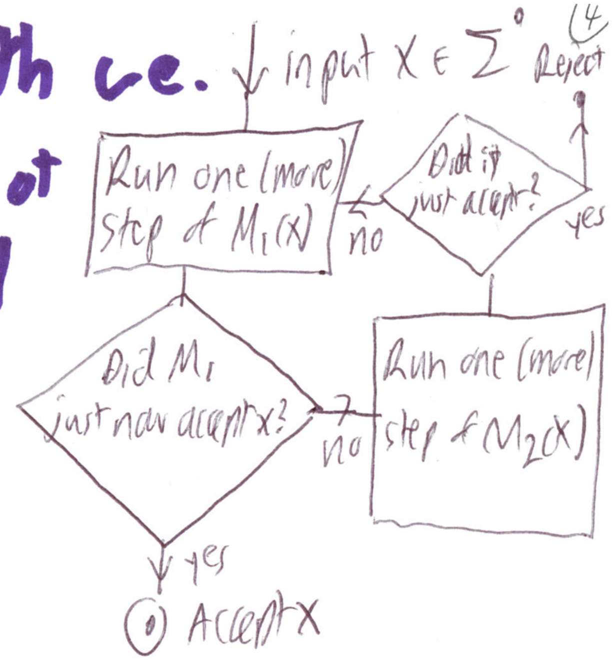
$A \leq_m B$ means the angle from A up to B is at least as steep as the diagonal walls.

Since $\tilde{B}$ is c.e., we get $\tilde{A}$ is c.e. by part (b), so A is co-c.e. ☒☒☒

By symmetry of REC, A decidable $\iff \tilde{A} \in$ REC

Diagram also conveys <u>Theorem</u> 3: A is decidable $\iff$ both A and $\tilde{A}$ are c.e. I.e. $\boxed{\text{RE} \cap \text{co-RE} = \text{REC}}$.

Proof: Suppose B and $\tilde{B}$ are both c.e. Then we can take TMs M_1, M_2 (not necessarily total) s.t. $L(M_1) = B$ and $L(M_2) = \tilde{B}$. Build M_3 to execute the following flowchart loop:



Since $L(M_1)$ and $L(M_2)$ are complementary, on any input x, exactly one of them will eventually accept. Thus $M_3(x)$ always exits the loop (by) then, so M_3 is total and $L(M_3) = B$, so $B \in REC$.

Contrapositive of ~~[illegible]~~ Theorem 2: Suppose $A \leq_m B$. Then

(a) if A is undecidable then B is undecidable

(b) if A is not c.e. then B is not c.e.

(c) if A is not co-c.e. then B ditto.

$A_{TM} \leq_m NE_{TM}$

Example: We can prove NE_{TM} is undecidable by showing

Goal: Build a computable function f s.t. $\langle M, w\rangle \in A_{TM} \iff f(M,w) \in NE_{TM}$.

$f(M,w)$ will be the code of a single machine M' such that

$$\boxed{L(M') \neq \emptyset \iff M \text{ accepts } w.}$$

Correctness of the reduction

Proof is mainly constructing M' from M and w.

$\langle M, w \rangle \xrightarrow{f} M' =$ ↓ input y [Simulate $M(w)$ if & when it accepts → Accept y]

We can build the code of M' given that of M and w.

And for correctness:

If M accepts w, then for all y, M' accepts y, so $L(M') = \Sigma^*$, so $L(M') \notin \emptyset$.

$\therefore \langle M, w \rangle \in A_{TM} \Rightarrow \langle M' \rangle \in NE_{TM}$.

But if M does not accept w, then for all y, M' never gets to accept y, so $L(M') = \emptyset$.

$\langle M, w \rangle \notin A_{TM} \Rightarrow \langle M' \rangle \notin NE_{TM}$ [$\in E_{TM}$ instead]

$\therefore \langle M, w \rangle \in A_{TM} \Leftrightarrow f(M, w) = \langle M' \rangle \in NE_{TM}$

and f is computable, so $A_{TM} \leq_m NE_{TM}$, and A_{TM} is undecidable, so <u>NE_{TM} is undecidable</u>.

Note, however, that NE_{TM} <u>is</u> c.e. → Thursday.