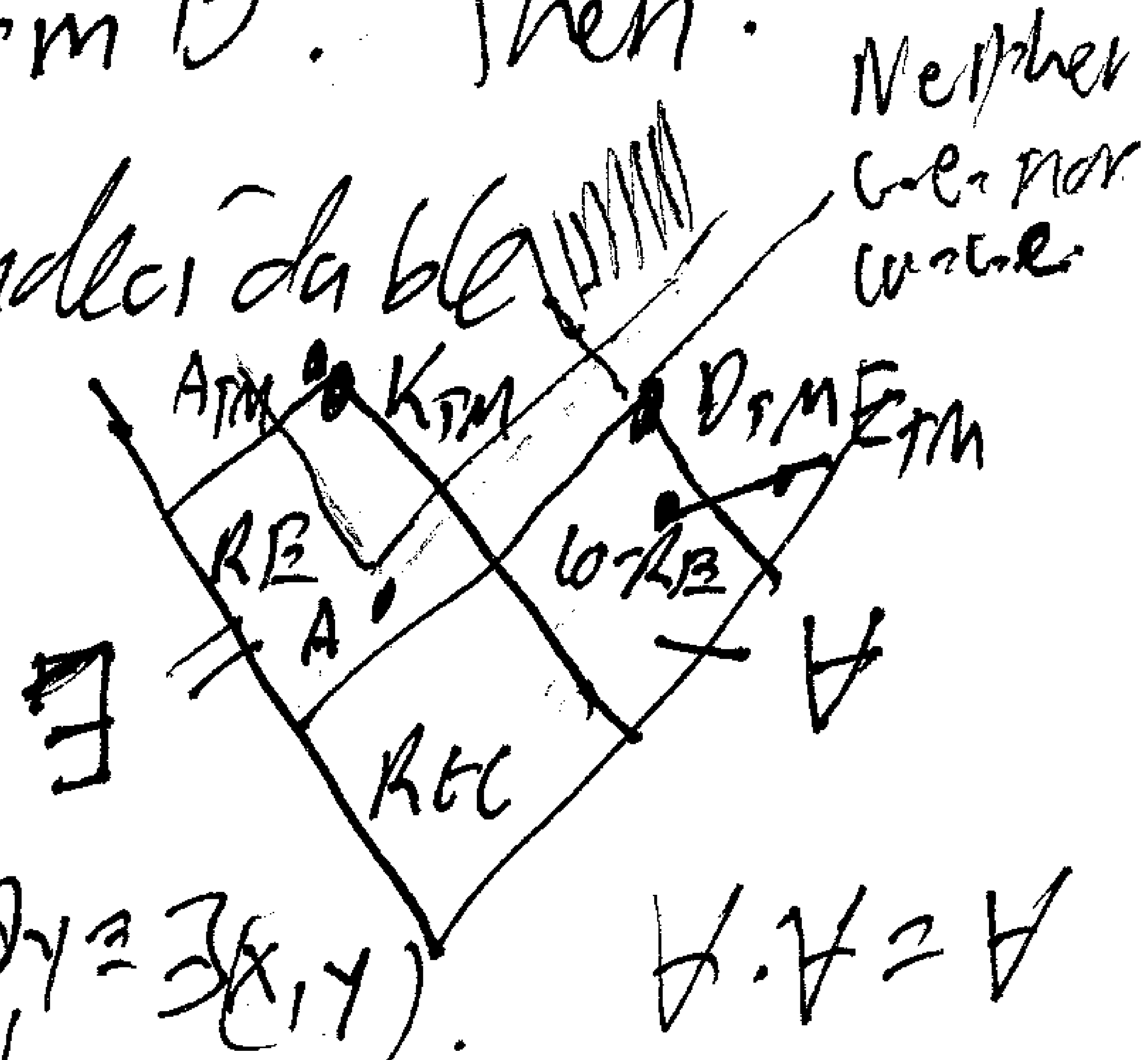
Lecture Tue May 5 — Undecidability Via (or Not Via) Reductions.

Contrapositive of last Thm: Suppose $A \leq_m B$. Then:

- If A is undecidable then B is undecidable
- If A is not c.e., then B is not c.e.
- If A is not co-c.e., then B is not co-c.e.



Suppose $A \leq_m C$ where A is not co-c.e. and also $D_{TM} \leq_m C$ (we know D_{TM} is not c.e.)

$E_{TM} = \{\langle M \rangle : L(M) = \emptyset$ i.e. for all x, $\{M$ does not accept $x\}\}$ — not a decidable predicate.

$= \{\langle M \rangle :$ for all x and all traces $\vec{c}$ (which are sequences of configurations) $\boxed{\vec{c} \text{ is not a valid accepting trace of a computation of } M \text{ on input } x}\}$.

$NE_{TM} = \{\langle M \rangle : (\exists x)(\exists c)$ [c is an accepting computation trace of M on input x]$\}$

We might expect to be able to do $A_{TM} \leq_m NE_{TM}$, but not $A_{TM} \leq_m E_{TM}$.

NE_{TM} is c.e.: A nondet TM N can guess any x and a c, and feed $\langle M, x, c \rangle$ to a universal TM computation checker.

$NE_{TM} = L(N)$, and N can be simulated by a DTM.

Show NE_{TM} is undecidable by (a) arguing as in §5.1 or (b) by reduction.

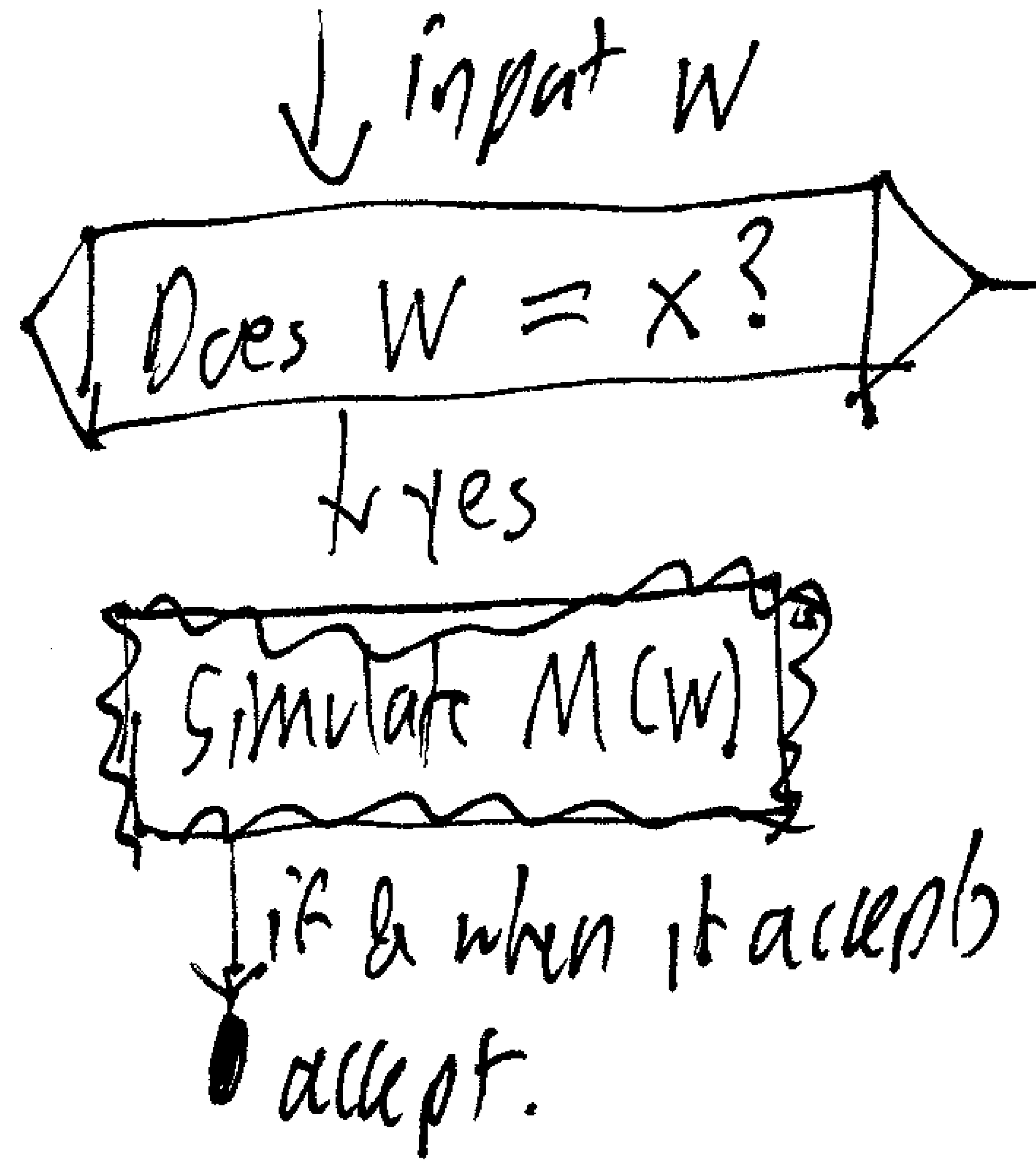
(a) §5-1. Suppose we had a decider R for NE_{TM}. (or for E_{TM})
then we could build a decider S for A_{TM} as follows.

(instance of A_{TM})

S = "On input $\langle M, x\rangle$ where M is a det TM and x is a string:

$\boxed{1.}$ Modify M's code to a TM M' that first does a test for whether its own input w equals x:

built from M and x. → M' =



Either $L(M') = \emptyset$ or $L(M') = \{x\}$

$\underbrace{L(M') = \{x\}}$ accepts iff w = x and M(w) accepts, ie $\langle M, x\rangle \in A_{TM}$.

(Step 1 can be done computably / effectively)

$\boxed{2.}$ Feed $\langle M'\rangle$ to our decider $\boxed{R}$ for NE_{TM}, and copy back the same answer

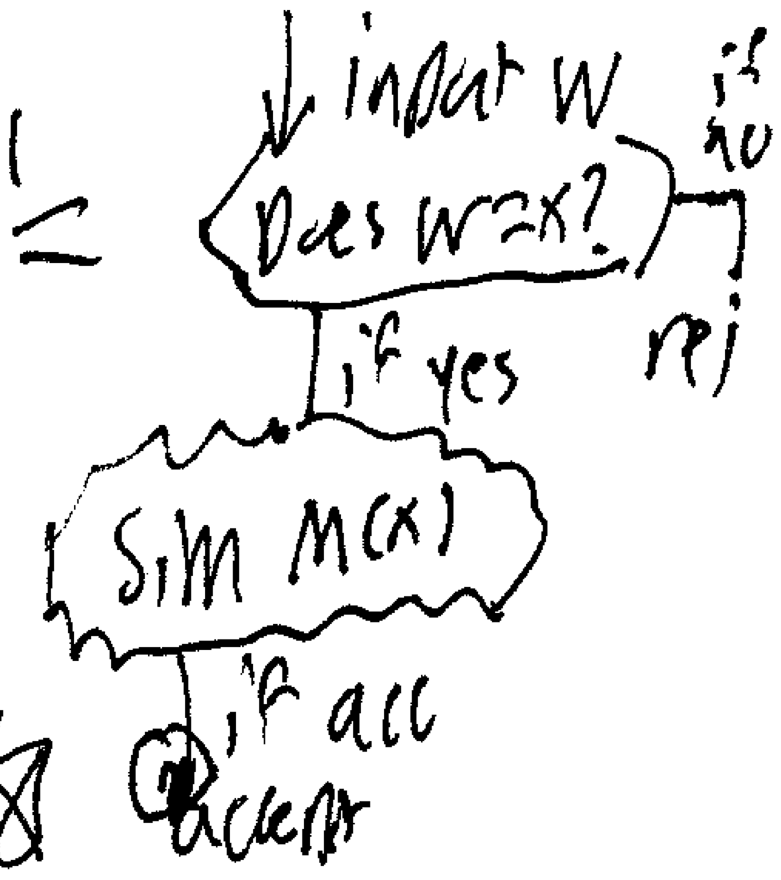
[3. If R accepts, accept, else reject.]

Given that R is total, S is total because steps $\boxed{1}$ and $\boxed{2}$ (and $\boxed{3}$) always halt.

Then S would decide A_{TM} since $\langle M, x\rangle \in A_{TM} \equiv$ M accepts x $\Leftrightarrow$ M' accepts x (and no $w \neq x$) $\Leftrightarrow L(M') \neq \emptyset \Leftrightarrow \langle M'\rangle \in NE_{TM}$.

But since A_{TM} is undecidable, S cannot exist. ∴ R cannot exist ∴ NE_{TM} is undecidable ☒.

(b) §5.3: To show $A_{TM} \leq_m NE_{TM}$, $\underbrace{\langle M, x\rangle}_{\text{instance of } A_{TM}} \xrightarrow{f} M' =$



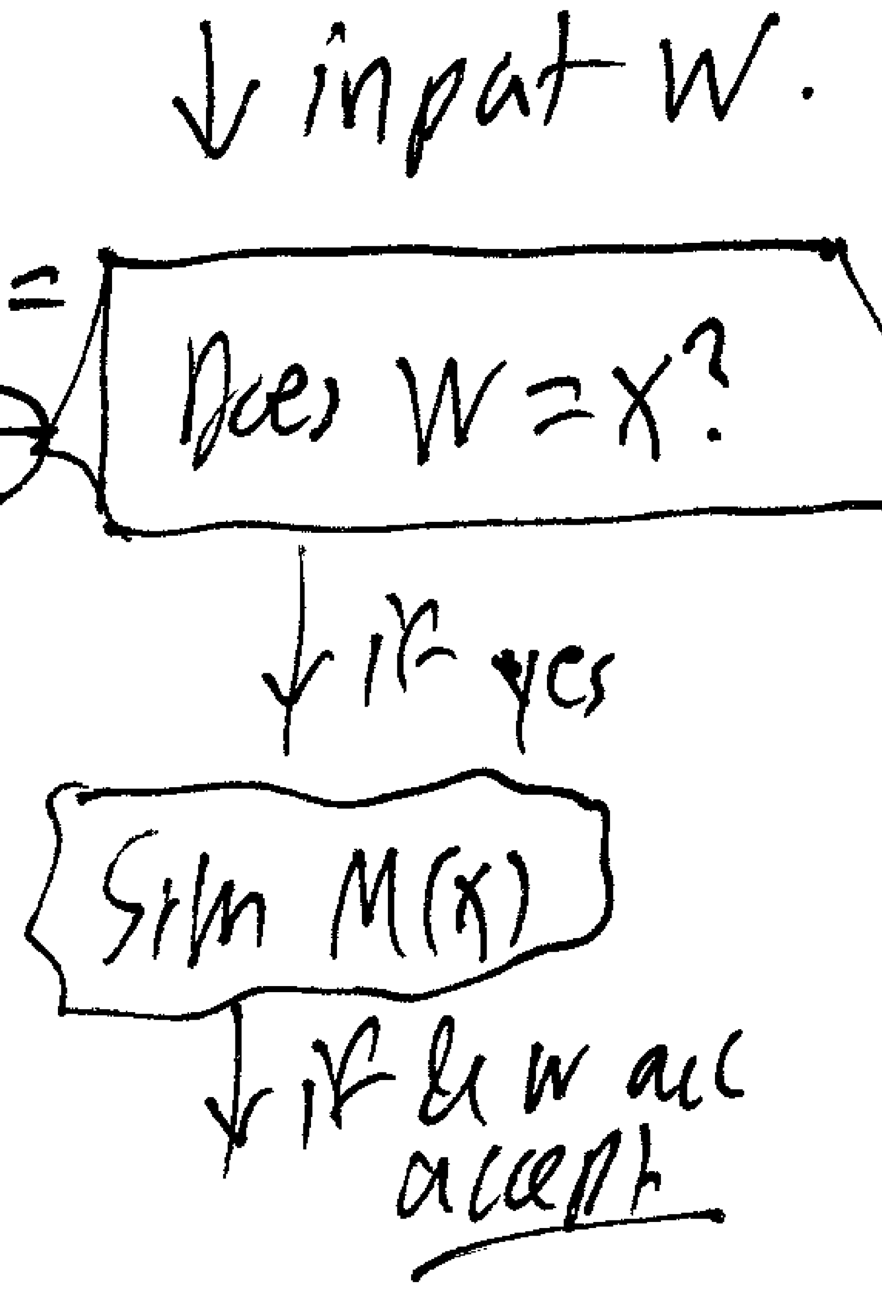
define a computable function f by:

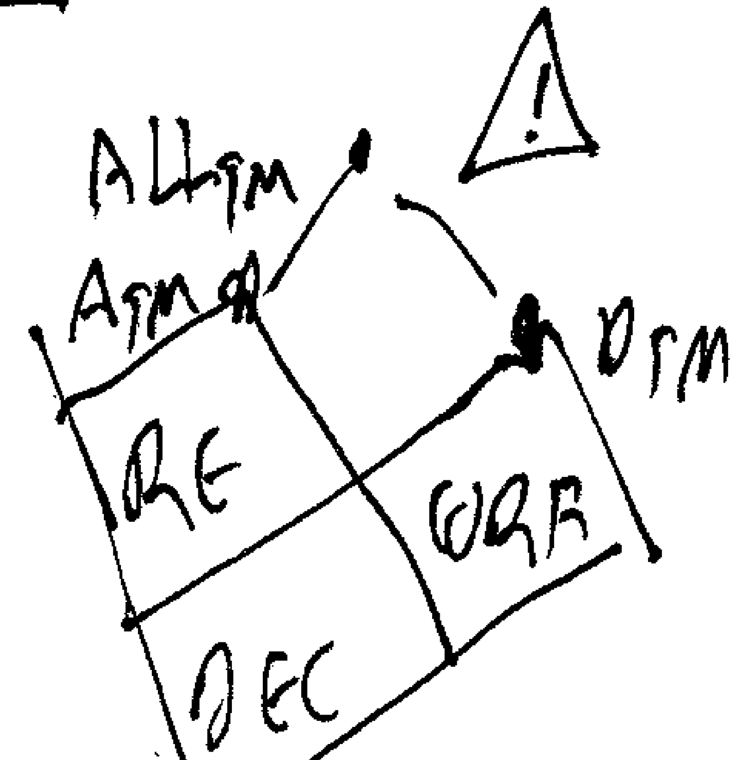
f is computable since it only does a code translation. The reduction f is correct because $\langle M, x\rangle \in A_{TM} \Leftrightarrow f(M,x) = M' \in NE_{TM}$ as above. ☒

$A_{TM} \leq_m \{M : L(M) = \Sigma^*\} \equiv ALL_{TM}$ via reduction.

~~Wrong but interesting~~: $(\forall x \in \Sigma^*)(\exists \vec{c})[\vec{c}$ is a valid acc trace of $M(x)]$

Let's try a variant of the last reduction: $\langle M, x \rangle \xrightarrow{f} M' =$





Clearly this f is computable — it just makes one change to the previous f

Analysis:

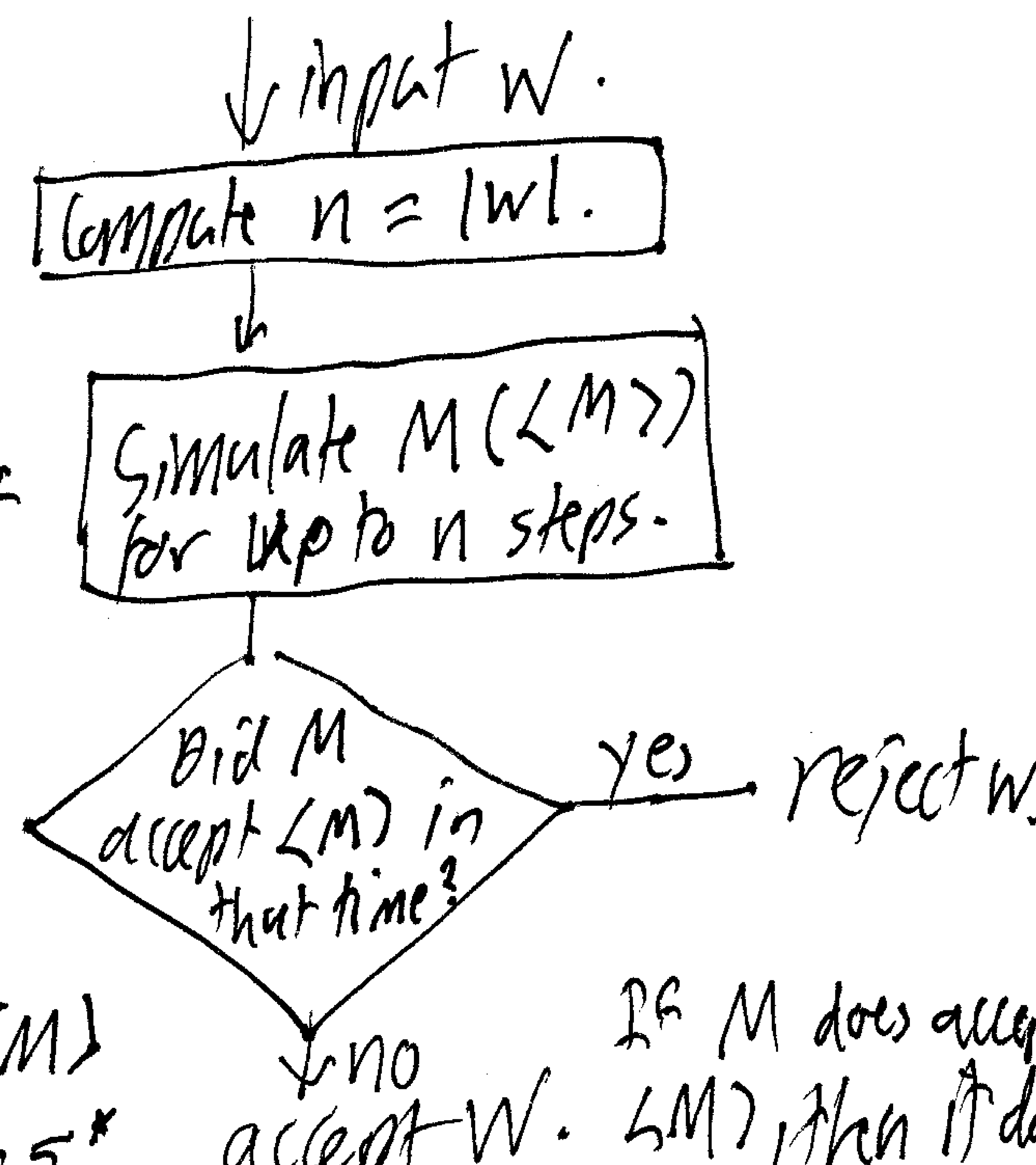
$\langle M, x \rangle \in A_{TM} \Leftrightarrow M'$ accepts all strings w including $w = x \Leftrightarrow f(M, x) = M' \in ALL_{TM}$.

$\therefore A_{TM} \leq_m ALL_{TM}$ so ALL_{TM} is undecidable, indeed not co-c.e.

Does $D_{TM} \leq_m ALL_{TM}$ as well?

Yes, by a "waiting" code construction.

$\langle M \rangle \xrightarrow{g} M'' =$



instance of $D_{TM} \equiv \{M : M$ does not accept $M\}$.

Then g is also a computable code translation.

Correctness: $\langle M \rangle \in D_{TM} \equiv M$ does not accept $\langle M \rangle$

$\Leftrightarrow$ the diamond never rejects any $w \Leftrightarrow L(M'') = \Sigma^*$

$\equiv f(M) \in ALL_{TM}$.

If M does accept $\langle M \rangle$, then it does so in some number n of steps $\Rightarrow$ all w with $|w| \geq n$ get rejected $\Rightarrow$ $L(M'') \neq \Sigma^*$, indeed finite

$\therefore$ since D_{TM} is not c.e., ALL_{TM} is not c.e. either.

$\therefore ALL_{TM}$ is neither c.e. nor co-c.e.