(1) PROVE THAT THE LANGUAGE
$L_1 = \{\langle M, x\rangle \mid M$ HALTS ON INPUT $X\}$ IS UNDECIDABLE

SUPPOSE THAT $L_d$ IS A DECIDER FOR $L_1$.
FIX $M_i$ AND CONSIDER W'S OF THE FORM
$\langle M_j\rangle$ FOR SOME OTHER TM, $M_i$.

LIST OUT ENCODINGS OF TM'S IN LEX ORDER
$\langle M_0\rangle, \langle M_1\rangle, \ldots$ THEN WE CAN CREATE AN
INFINITE BINARY SEQUENCE WHERE WE HAVE A 1
IN THE $j$TH SLOT IF $\langle M_j\rangle$ CAUSES $M_i$ TO
HALT AND A 0 OTHERWISE. IF $L_d$ IS A
DECIDER $L_1$ THEN WE CAN CONSIDER A VARIANT
ON THE COMPLEMENT OF THE DIAGONAL OF THE
MAP $F: \langle M_i\rangle \mid \rightarrow (A(\langle M_i, \langle M_0\rangle\rangle),$
$A(\langle M, \langle M_1\rangle\rangle), \ldots)$ IN PARTICULAR, WE CAN LET
$D$ BE THE MACHINE:

$D$ = "ON INPUT $\langle M\rangle$, WHERE $M$ IS A TM:
- RUN $H$ ON INPUT $\langle M, \langle M\rangle\rangle$
- IF $H$ SAYS YES, RUN FOREVER,
  IF $H$ SAYS NO, HALT AND ACCEPT

CONSIDER $D(\langle D\rangle)$. MACHINE $D$ HALTS IF AND
ONLY IF $L_d$ ON INPUT $\langle D, \langle D\rangle\rangle$ REJECTS. BUT
$L_d$ ON INPUT $\langle D, \langle D\rangle\rangle$ REJECTS MEANS THAT $D$
DID NOT HALT ON INPUT $\langle D\rangle$. THIS IS
CONTRADICTORY.

3) Steps to be followed

1) Stimulate k-tape to one tape by copying the contents of each k-tape to one tape followed by # sign.

2) Read from left to right on stimulated one tape to read the '_' to determine the state.

3) Pick a transition based on the input.

4) For each section of the stimulated tape we move from right to left, while a content on the tape and make it as a underscore '_'.

(2) Show $ALL_{CFG} = \{\langle G \rangle \mid G$ is a CFG w/c accepts all strings$\}$ is undecidable.

Suppose D decides $ALL_{CFG}$.

Consider the machine w/c on input $\langle M, w \rangle$

From the encoding $\langle M, w \rangle$ we can compute the encoding of a CFG $\langle G \rangle$ w/c generates strings s.t. either

(1) the string does not start w/ a starting configuration $C$ of M on w.

(2) the string does not end w/ a ending configuration of M on w.

(3) there is a substring $\#C_i\#C_{i+1}\#$ of length 2 configs s.t. $C_{i+1}$ does not follow $C_i$ according to M.

→ the pt is G will acapt all strings iff M does not accept w. So if had a decider D for $ALL_{CFG}$ would run it on $\langle G \rangle$ and decide $A_{TM}$ ⇒⇐

(4) In class we argued that a for any nondeterministic TM which computes a language in time $f(n)$ there's a constant $c$ such that it can be simulated by a deterministic machine running in time $c^{f(n)}$. How is $c$ determined?

Let N be a NTM and we choose D as a TM that simulates N. The idea is to have D try all possible brances of N's compution.
We define c be the finite maximum number of choices in any given state reading a given symbol for a next state. So each computation we are working on we are using 1 cycle. But we might have more than 1 cycle and we might have computations of length 2. So we can say for computations length 2 we have the time of simulation $c^2$.
If we have more length for example computations of length n then we have the time $f(n)$.
And if we notice the runtime of this algorithm to simulate $f(n)$ steps is $O(1+c+c^2+\dots+c^{f(n)}) = O(c^{f(n)+1}) = O(c^{f(n)})$ so will be the TIME $(c^{f(n)})$ by linear speed-up.

(5) Suppose we could decide the language $\{\langle M' \rangle \mid M'$ has a useless state$\}$ via machine D.

Consider the map

$$\langle M, w \rangle \longmapsto \langle M' \rangle$$

where $M'$ is a machine which on input $x$ checks is $x \neq w$ and if so halts and rejects and otherwise if $x = w$, $M'$ simulates $M$ on $w$ and halts and accepts iff $M$ on $w$ halts & accepts. So the accept state of $M'$ is useless iff $M$ does not accept $w$. The map $\langle M, w \rangle$ to $\langle M' \rangle$ can be computed by a TM so this is a Turing reduction

(6)

Because it is difficult to measure space-complexity with traditional TM models, when looking a sublinear space. because an input of size $n$, already takes up space $n$. Thus, we use an offline TM. w/ 2 tapes.

We require space in addition to the space required for input.

8

1st part

7)

Some turing machine with mapping function f

$$x \in \underset{\substack{\uparrow \\ \text{decidable}}}{L} \xrightarrow{f} \underbrace{x' \in L'}_{\text{regular}}$$

PRACTICE FINAL

CS154 section

for function f:

when reading input from x

if input is accepted → change to zero

if input is rejected → erase it

when f is done:

x' is the output with either a string of zeros or nothing (empty string).

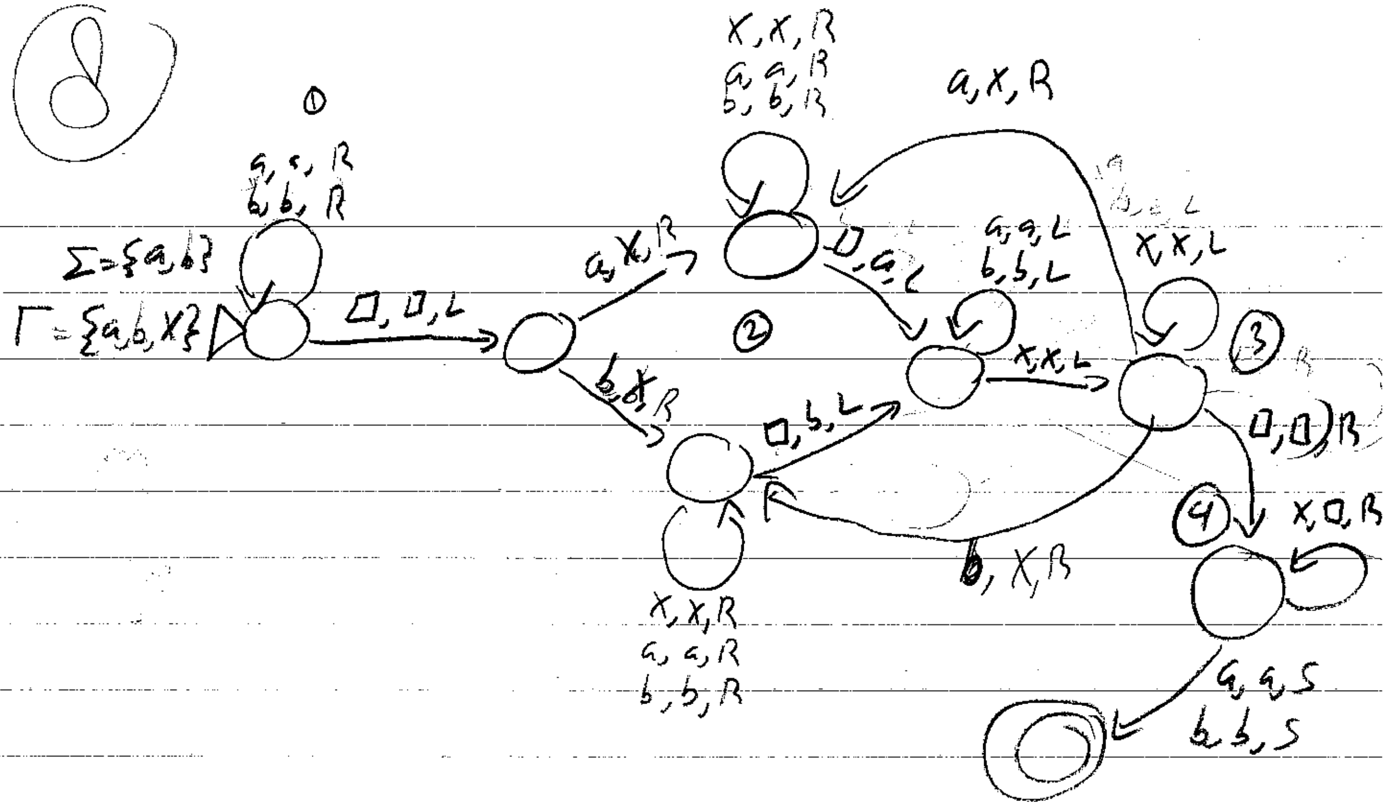
So, if x' is a string of zeros then it is in L' which is a regular language.

2nd part

$A_{TM}$ is R.E. but not decidable, because every regular language is decidable.

therfore $A_{TM}$ is not regular.

8




① move to end

② if it reads an 'a', it marks that character as done (X) an moves to the end of the string and writes an 'a'. it does the same for 'b', except it writes a 'b'.

③ moves back to the last "not-done" character in the original string and goto step ②.

④ if the character after the first X is a space (we finished reading the original string) we move right, spacing out each X until we reach the start of the reversed string, then we accept. BLAMO.

9) It is decidable:

Using a universal turing machine, to decide ...

1) run input TM on x for n steps
2) If TM halts at any step then accept, else reject

10 We Showed in class that (via universal machine)

$A_{TM}$ is r.e but not decidable *

If $A_{TM}$ were Co. re.

Then it would be both r.e & Co.r.e.
So from class we'd know it was decidable
This would contradict *.

$\therefore$ $A_{TM}$ is r.e but not Co.r.e.