Homework 10:

Language: $\{ s \in \{0,1\}^* \mid s$ has twice as many 0's as 1's $\}$

write a TM recognizing L.

- 2-tape solutions
- 1-tape solutions (slower)

move through input, looking for two 0's and one 1 in the string

Do this repeatedly. $\Gamma = \{0, 1, x, \sqcup, x'\}$

0 0 0 0 1 1 0 1 1

x' x 0 0 x 1 0 1 1

States $q_{\text{saw } 1}$, $q_{\text{saw } 0}$ $\boxed{q_0, q_{acc}, q_{rej}}$

$q_{\text{saw } 01}$ $q_{\text{saw } 00}$ ⟶ 001

$q_{\text{move left until hit } x'}$

TM for twice as many 0's as 1's

2. an explanation of what each state in Q represents in terms of your algorithm,
3. a list of $\Sigma, \Gamma, q_0, q_{\text{accept}}, q_{\text{reject}}$, and
4. a description of δ, either by (1) a list of its values, or (2) a diagram.

Solution: High-level description: Scan through the input, left end to right end, marking off two 0's and one 1 with an x. Continue doing this until either (1) everything on the tape is blank and x's (we accept), or (2) what's left on the tape does not have at least two 0's and one 1 (we reject).

Initially we mark the first tape symbol with an x', to know when we have reached the leftmost tape cell (i.e., cell number 1). For simplicity, we'll have the initial state q_0 always write x' over the 0 or 1 it sees, and over any x's beforehand, since it this simply fills a block of cells at the beginning of the tape with x''s.

[There are many possible variants on the above.]

Here is a table of δ values, where b denotes the blank symbol:

Tape Symbol

State

$Q \setminus \Gamma$	0	1	x	x'	b
q_0	q_{s0}, x, R	q_{s1}, x, R	R		q_{acc}
q_{s0}	q_{s00}, x, R	q_{s01}, x, R	R		q_{rej}
q_{s1}	q_{s01}, x, R	R	R		q_{rej}
q_{s00}	R	q_{left}, x, L	R		q_{rej}
q_{s01}	q_{left}, x, L	R	R		q_{rej}
q_{left}	L	L	L	q_0, x', R	

Many Left?

$\Gamma \to \Gamma^2$

$0, 1, x, \ldots$

$\to (0,0), (0,1), \ldots$

Could work

$(0,*)$ explain: $*$ = any

$Y = \Gamma \setminus \{0\}$

with the convention is that R means "don't change the state or symbol on the tape and move R," and similarly for L; an empty value means that it is irrelevant.

The set of states, Q, consists of those in the first column of the table along with q_{acc} and q_{rej}. The tape alphabet, Γ are the symbols in the first row of the table. The staes q_0, q_{acc}, q_{rej} are eponymous.

The meaning of the states:

1. q_{s01} means that so far we have seen one 0 and one 1 in our search for finding two 0's and one 1 in what is left of the input;
2. q_{s0}, q_{s1}, q_{s00} similarly;
3. q_{left}: we are moving left to the first x' (which is the start of what's left on the tape.

In a separate PDF we have given a diagram of the above Turing machine, following the labeling conventions in [Sipser], as the last page of this document.

[Which do you think is easier to work with and verify for correctness?]

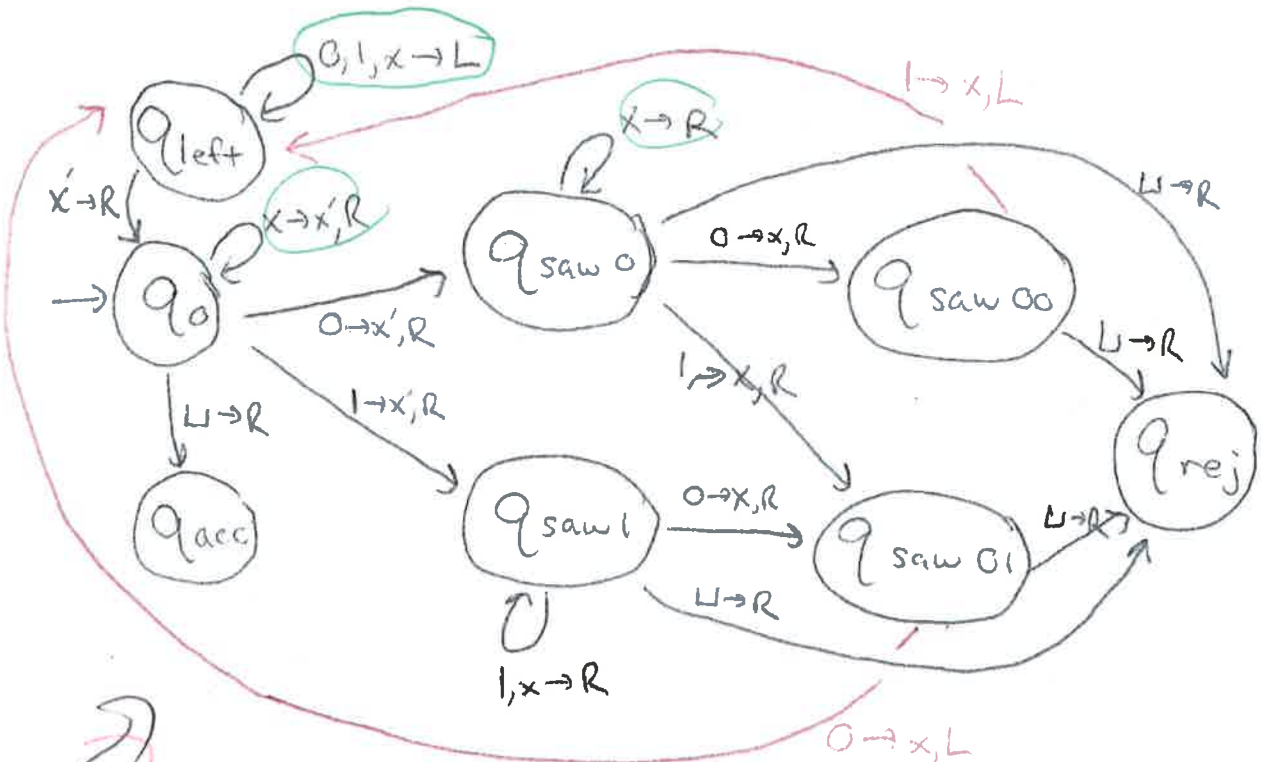
Question 4:

This problem is optional, worth 0 points. Let

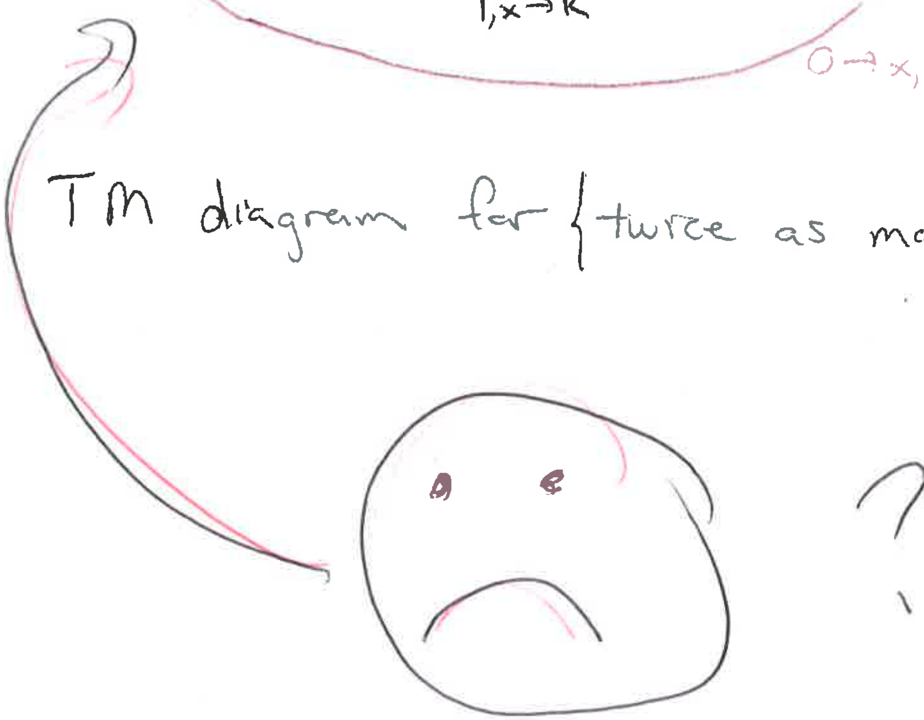
$$5\text{COLOR} = \{\langle G \rangle \mid G \text{ is colorable with 5 colors}\}.$$

Show that 5COLOR is NP-complete; you may use the fact that 3COLOR is NP-complete (see Problem 7.29 of [Sip]).

Solution: To see that 5COLOR $\in$ NP, a nondeterministic Turning machine can nondeterministically choose (or guess) one of five colours for each of the vertices, and then test whether this gives a valid 5-colouring of the graph.



TM diagram for {twice as many 0's as 1's}



[5] **3.** Let $L = \{a^{100}\}$. ~~Argue~~ Prove that a DFA that recognizes L must have at least 101 states. Explain your argument from scratch; i.e., if you use want to use Myhill-Nerode, then explain why it is true in this case.

Assume otherwise

Pumping Lemma: { let $s = a^{100} \in L$. Then $a^{100} = s = xyz$ s.t. $\begin{cases} y \neq \varepsilon \\ xy^iz \in L \end{cases}$
but xy^2z is of length ≥ 101, so $xy^2z \notin L$

Myhill-Neroda: { Compute AcceptingFutures(L, a^m), $m = 0, \ldots, 100$

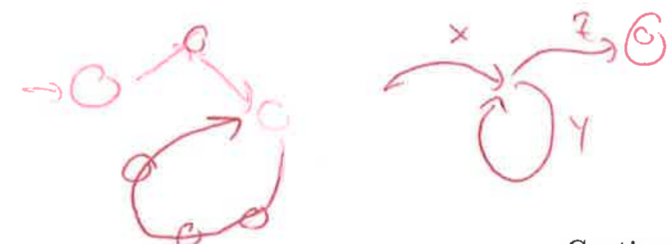
Walk-Counting { # words length n satisfies
$f(n) = c_1 f(n-1) + \ldots + c_{100} f(n-100)$

$AF(L, \varepsilon) = \{a^{100}\}$
$AF(L, a) = \{a^{99}\}$
$\vdots$
$AF(L, a^{100}) = \{\varepsilon\}$
$AF(L, a^{101}) = \emptyset$

for $0 \leq m \leq 100$
$AF(L, a^m) = \{a^{100-m}\}$
$AF(L, a^{101}) = \emptyset$
for each m, this is a one elt set

$L' = \{a^5, a^7, a^{100}\} \Rightarrow AF(L', \varepsilon) = \{a^5, a^7, a^{100}\}$ ← Look at
$AL(L', a) = \{a^4, a^6, a^{99}\}$ ← largest element

DFA $S = S_1 S_2, \ldots$



Continued on page 5

[32] **5.** 4 points per part. **Briefly justify** your answers; you will not get credit for just writing "yes" or "no" (or any short answer without justification).

(a) Show that if $C_1, C_2, \ldots$ are countable sets, then $C_1 \cup C_2 \cup \cdots$ is countable.

(b) What does Savitch's theorem assert, and why does this show that NPSPACE (non-deterministic polynomial space) is equal to PSPACE (polynomial space)?

(c) Let A be any problem that is complete for PSPACE under polynomial time reductions. Is PSPACE contained in P^A?

SNEAKY-NTM $= \{ \langle M, w, 1^t \rangle \mid M$ accepts w in time $\leq t \}$

(d) Let $\boxed{L_{\text{NP easy}}}$ consist of all descriptions of a triple, $\langle M, i, t\rangle$ where M is a non-deterministic Turing machine that accepts input i, running in time t where t is expressed in unary. Show that any language in NP can be reduced to $L_{\text{NP easy}}$ by a polynomial time reduction.

L non-det M in time Cn^k

Input w to M $\longrightarrow$ $\langle M, w, 1^{Cn^k} \rangle$ where $n = |w|$

SNEAKY-NTM

Continued on page 7

5(d) Say $L \in NP$ can be recognized by a non-det T.m. M in time Cn^k.

Let $f: \Sigma_L^* \to \Sigma_{\text{SNEAKY-NTM}}^*$ be given by

$$f(w) = \langle M, w, 1^{Cn^k} \rangle \text{ where } n = |w|.$$

Then f reduces L to SNEAKY-NTM

For any L, is there an oracle TM, M^L, that recognizes $HALT^L$?

Simple: Is HALT recognizable by a TM? If so

" " $HALT^L$ " " " TM oracle L

Answer: Yes, recognize with univ TM

- - - - - - - with oracle L

[Simplify problem 5(e)]

Let L be anything at all

(e) ~~Let HALT be the Halting Problem for Turing machines~~, and let $\mathrm{HALT}^{\mathrm{HALT}}$ be the Halting Problem for oracle Turing machines with oracle HALT. Is there an oracle Turing machine with oracle HALT that can recognize $\mathrm{HALT}^{\mathrm{HALT}}$? L

A universal T.m. with oracle L can be used to recognize HALT^L for any L, by simulating M^L on input $\langle M^L, w\rangle$. Taking L = HALT shows that the answer is yes.

(f) Same problem as the last part, except with "recognize" replaced with "decide."

No: for any L, A^L_{TM} and HALT^L_{TM} can be reduced to each other. Hence (Problem 2, Homework 8) since A^L_{TM} is undecidable by T.M.'s with oracle L, so is HALT^L. Taking L = HALT shows answer is no.

(g) Say that you are given a description, $< A, w >$, of an NFA (non-deterministic finite automaton), $A = (Q, \Sigma, \delta, q_0, F)$ and a word, $w \in \Sigma^*$, and that the transition function $\delta\colon Q \times \Sigma \to Power(Q)$ is written out by each of its values. Can you decide if A accepts w in polynomial time of the length of $< A, w >$?

Yes: an NFA runs in $|w|$ steps, and you can simulate each step by keeping track of a vector of length $|Q|$ that tells you which states you could be in.

(h) For each non-negative integer, n, let SAT_n be those strings of length n that lie in SAT. If SAT is in P, how do we prove that SAT_n has polynomial size circuits? How does this related to the question of P versus NP?

See p.305

Fig 7.38

The Cook-Levin theorem allows you to write a Boolean circuit whose gates are $X_{i,j,s}$ in this theorem, times a constant more gates to compute each $\{X_{i,j,s}\}_{j,s}$ from the $\{X_{i-1,j,s}\}_{j,s}$ from each s. This gives a circuit of polynomial size.

Continued on page 8

People are trying to prove $P \neq NP$ by showing that SAT does not have poly-size circuits.