Thomas DeMasse - HW4

1) (a) Read through the tape until the end. If the tape ends in either 111 or 10 it accepts but anything else it rejects

(b) M_1 = On input String W:

1.) Sweep left to right accross the tape until the end. If the tape is blank, reject.

2.) Move left in q_1 get 0 move left, go q_2

3.) In q_2 get 1, accept else, reject

4.) In q_1 get 1 Change to x, move left, go q_3.

5.) In q_3 get 1, Change to X, move left, go q_4, else reject

6.) In q_4 get 1 accept, else reject

(c) 7-Tuple!

$Q = \{q_0, q_1, q_2, q_3, q_4, q_{accept}, q_{reject}\}$

$\Sigma = \{0,1\}$

$\Gamma = \{0, 1, x, y, \sqcup\}$

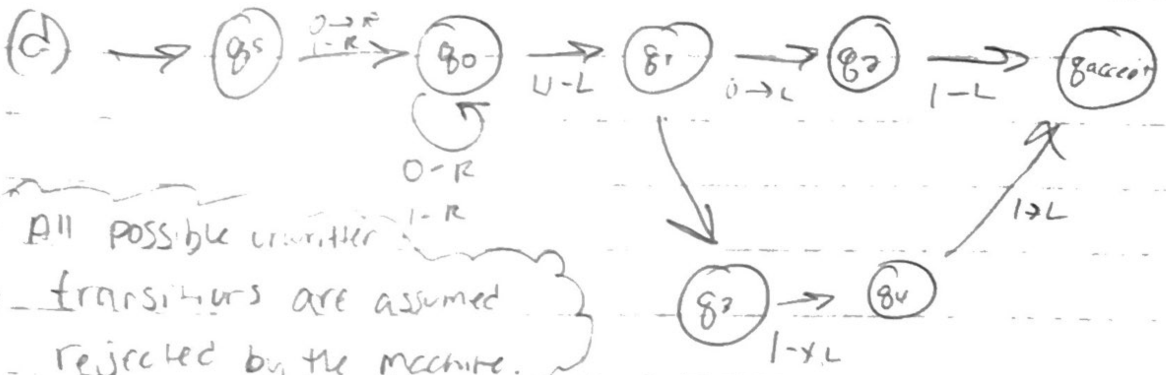
δ: Describe δ with a state diagram

q_s is start state

q_{accept} is accept state

q_{reject} is reject state

(d)



All possible character transitions are assumed rejected by the machine.

(e) I 101

$\sigma(q_s, 1) = (q_0, Y, R)$

$\sigma(q_0, 0) = (q_0, X, R)$

$\sigma(q_0, 1) = (q_0, Y, R)$

$\sigma(q_0, U) = (q_1, U, L)$

$\sigma(q_2, Y) = (q_3, Y, L)$

$\sigma(q_3, X) = (q_{reject}, X, L)$

halts at reject state

II $\sigma(q_s, 1) = (q_0, Y, R)$

$\sigma(q_0, 1) = (q_0, Y, R)$

$\sigma(q_0, 1) = (q_0, Y, R)$

$\sigma(q_0, U) = (q_0, U, L)$

$\sigma(q_1, Y) = (q_3, Y, L)$

$\sigma(q_3, Y) = (q_4, Y, L)$

$\sigma(q_4, Y) = (q_{accept}, Y, L)$

Accepted

2.) (a) Begin reading characters. set first a to w
then begin zig zagging over the tape matching all
A's and B's with x to know they were matched,
Subsequently as we are matching change all a's
to y and b's to z as long as only w, x, y, z are
found on the tape it is accepted.

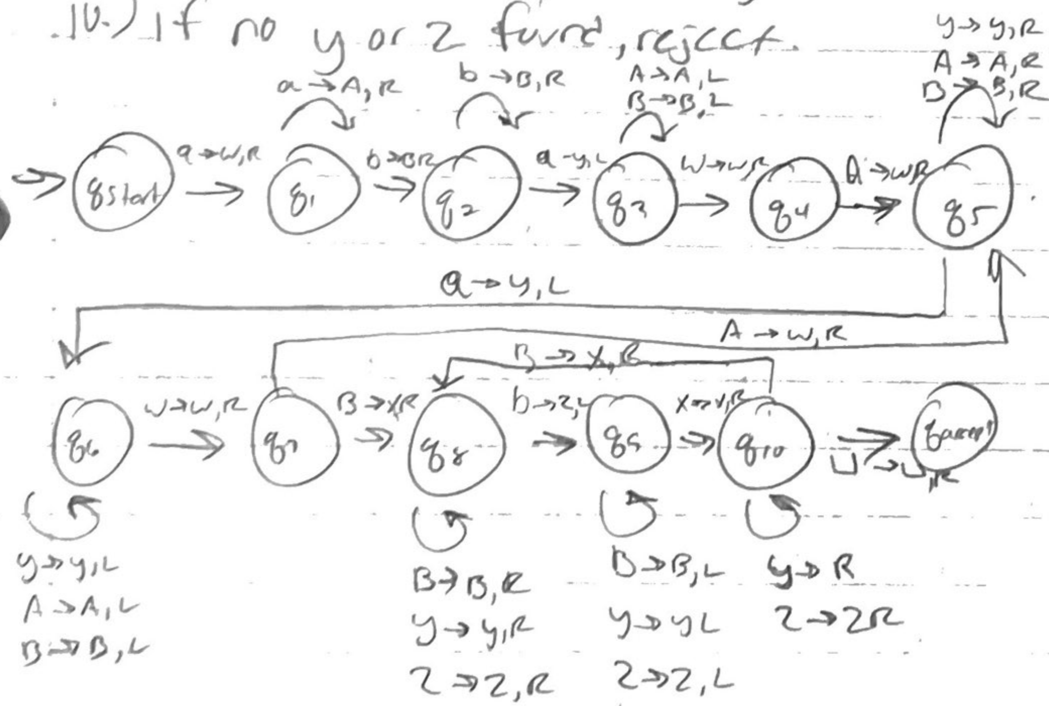
(b) M_2 = "On input string w:
1.) If w is empty, reject
2.) If w does not start with a, reject.
3.) Sweep left to right over the tape get
first a change to w then begin replacing
all remaining a's with A's, if read no b's, reject.
4) Hit first set of b's, begin replacing b's with B's
if blank found, reject.
5.) If found an a, change to y, halt and go
left to the first w and move right.
6.) Replace A with w and go right to find
a matching "a", if none found, reject
when a is found, replace a with y and
move to the left a w is found
move right of the w, if "A" is found
repeat 6

7.) If there is no "B" found, reject. There must be a set of B's after A's

8.) Replace B with x and move right on tape to find matching "b", if no "b" is found, reject. If matching b is found replace with z and move left until x is found. Move right, if found "B" repeat 8.

9.) If no B is found, and only found all y or z, accept

10.) If no y or z found, reject.

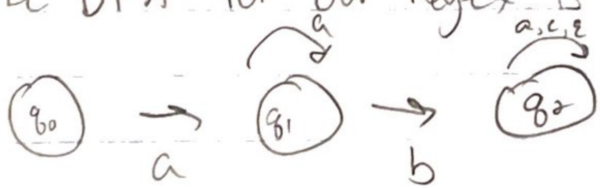


All unwritten transitions are assumed to be rejected by the machine.

3.) (a) The string "bacc" is not a part of the regular expression given that the string starts with b and the regex is looking for an a to start. It is not accepted

(b) The string "abb" is not apart of this regex because it contains 2 b's. The regex only contains 1 b. Given this it is not apart of the regex.

(c) the DFA for our regex is as follows!



where the string must start with an 'a', then have as many a's followed by a single b then as many a's or c's or none at all. This proves that the string B is accepted by the DFA. This is also because our string B is the same as that of the regex.

4.) We can express the language that we will use as such. $L = (x, y)$ where x is our DFA and y is a regex with $L(x) - L(y)$.
"$EQ_{DFA} = \{(A, B) \mid A$ and B are DFA's and $L(A) = L(B)\}$"

In our case where A is equal to X and B is equal to Y. We will assume that T is our Turing Machine which decides our language L. We can define our Turing machine such that: T = On input ⟨X, Y⟩ where X is a DFA and Y is a regex: Convert Y into a DFA using Kleene's theorem such that we have D_Y. Uses the Turing Machine defined TM and a decider A all based off of theorem 4.5 on our input of ⟨X, D_Y⟩. With this if our decider A accepts, the language is accepted. If A rejects then the language is rejected.

5.) Given that B is the set of all infinite sequences $(b_1, b_2, b_3, b_4, \ldots)$ such that all elements of b_i are in {0, 1}. To start we can suppose that B is countable. With this we can define a correspondence j between $X = \{1, 2, 3, 4, \ldots\}$ and B. For $x \in X$ we will let $j(x) = (b_{x1}, b_{x2}, \ldots)$ such that b_{xi} is ith bit in the xth sequence. The table below helps to visualize

x	j(x)
1	$(b_{11}, b_{12}, b_{13}, b_{14} \ldots)$
2	$(b_{21}, b_{22}, b_{23}, b_{24} \ldots)$
3	$(b_{31}, b_{32}, b_{33}, b_{34})$
⋮	

$y_1 = 0101100$

$y_2 = 1110000$

$y_3 = 0011001$

$y_4 = 1101000$

$y_5 = 0000001$

$y_6 = 0111111$

$y_7 = 111111$

Now we can define the infinite sequence $y = (y_1, y_2, y_3, y_4 \ldots) \in B$ where if the i^{th} bit in a row i is 0 set i^{th} bit of y to 1 and vice versa. we can model this in a table. So $y = 1000100$. y is definitely an infinite binary string. However, eventhough y is infinite it cannot be in the list of all binary strings. because it is in disagreement with all values of Y in the list. Therefore this is a contradiction proving that B must be uncountable.

6.) To show that $INFINITE_{PDA}$ is decidable, we can construct a turing machine for the following: Given that M_{15} is a PDA, we can convert it into an equivalent CFG labeled N. We can then take this CFG N and convert it to the Chemistry Normal Form and call it N'. Then we can perform a breadth-first search on N's grammer rules looking for recursion. If there exists a derivation such that $B \Rightarrow uBv$, then N accepts if not N rejects.