Nikitin Anton

CS 321 HW - 7

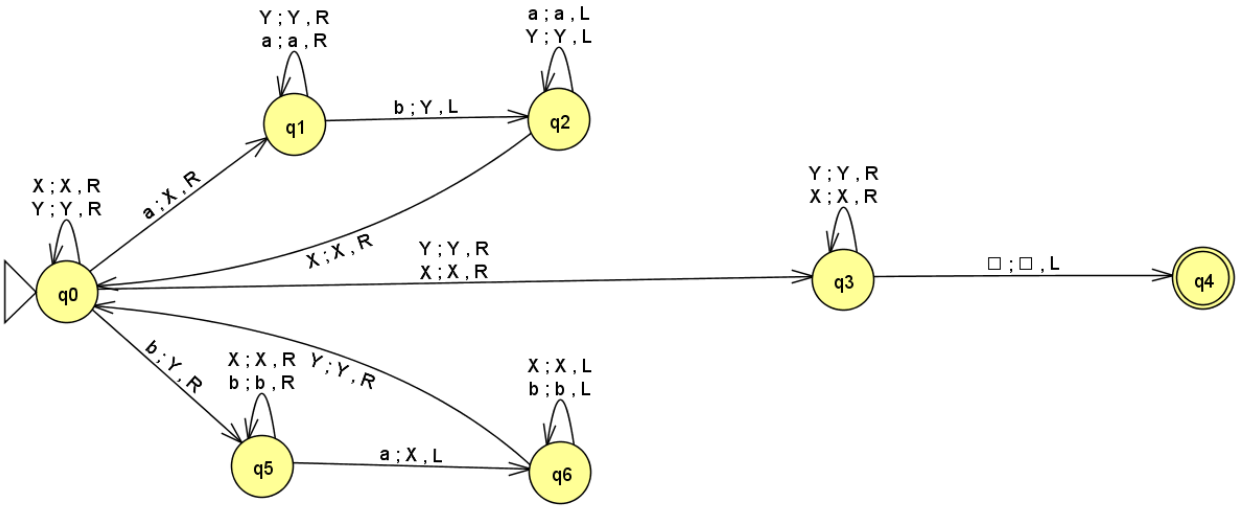
1. *(10 pts)* Design single-tape Turing machines that accept the following languages using JFLAP

   a) $L_2 = \{ w : n_a(w) = n_b(w) : w \in \{a, b\}^+ \}$.

Test case	Result
abbaba	accept
aaabbb	accept
aaaaaabbbbbb	accept
ba	accept
a	reject
abb	reject
bbaab	reject

Answer:

This Turing Machine will accept only the same numbers of a`s and b`s. We change a on X and going right. Further, when we see a, do nothing, if b it changes on Y and we going left. The same way for the first b. When we have only changed symbols X and Y we go ahead until blink. When found that we going to final state. Successful.



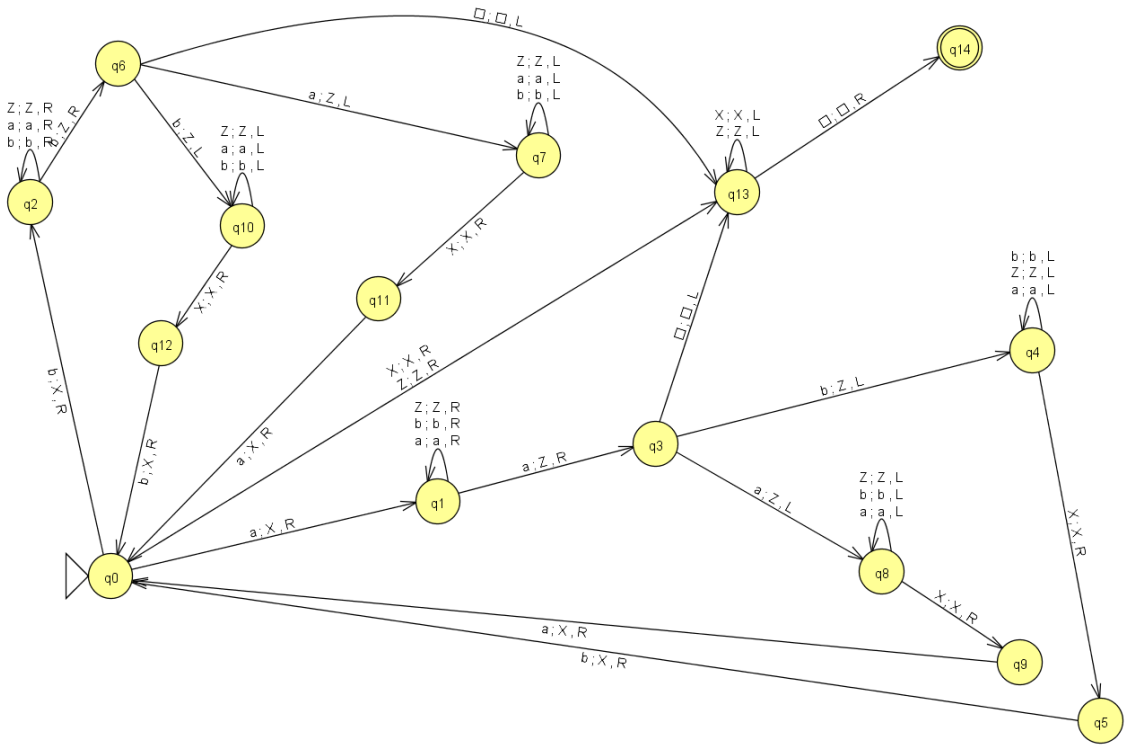
b) $L_3 = \{ww : w \in \{a, b\}^+\}$.

Test case	Result
abaaba	accept
bbbbbb	accept
aabbaabb	accept
a	reject
aabb	reject
bbb	reject

Answer:

In this case we have two strings which can contain a`s and b`s, or only one of that. The main idea of Turing Machine here:

We get first symbol from first string. After that it changes on X, further we go ahead until first symbol of second string will be the same (a or b) and change it to Z, turn back to first string (going left). If we find X go right and change this element on X, further we go ahead until we find Z, therefore we need to find the same element (but also if we found Z, just go right before a or b). As result, if we have the same set of symbols, machine will accept this string. In different situation reject.



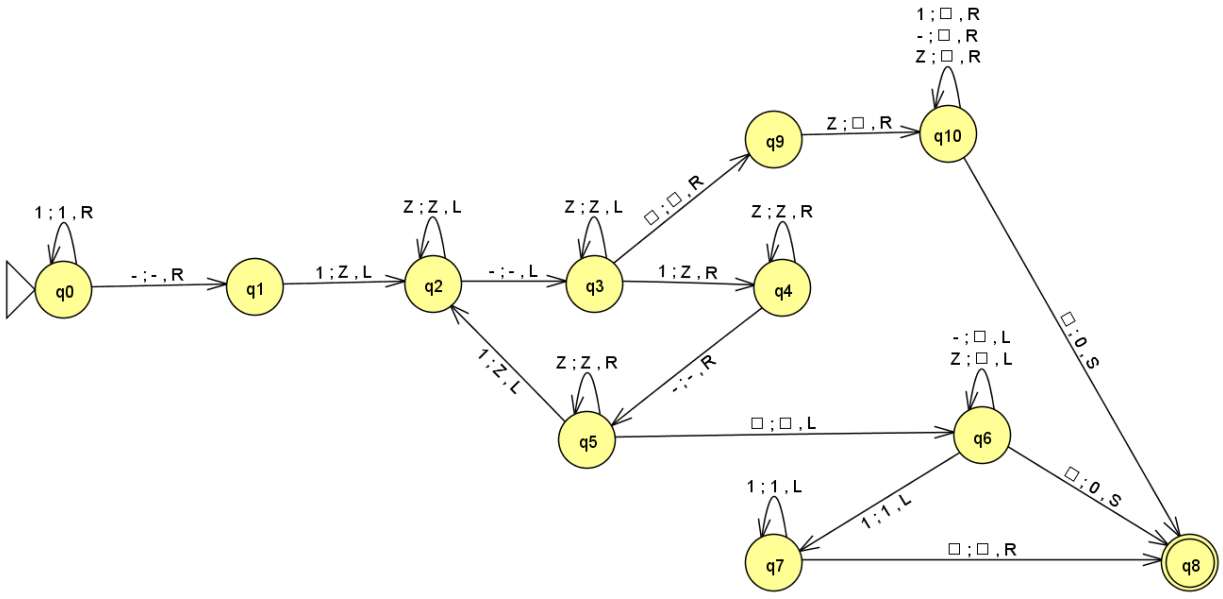
2. *(10 pts)* Design Turing Machines using JFLAP to compute the following functions for x and y positive integers represented in unary. The value $f(x)$ represented in unary should be on the tape surrounded by blanks after the calculation.

a) $f(x) = \begin{cases} x - y, & x > y \\ 0, & \text{otherwise} \end{cases}$

Input	Output	Result
11-1	1	Accept
1-1	0	Accept
111-1	11	Accept
1-1111	0	Accept
1111-11	11	Accept

Answer:

This Turing Machine work similar such as in 1.b. But here we start to change from second string. After we turning back to the first and change that and also we use the same symbol Z for both of strings. According with condition x – y, where x –first string, y –second string, if x > y turn number of 1`s, if it is equal or y > x turn 0. And more difference in compare with 1.b method. We start to change symbols in first string from end. If we have 1 or some number of 1`s after all changes, we deleting others symbols, turning back to left blink behind 1`s and turn them like output.

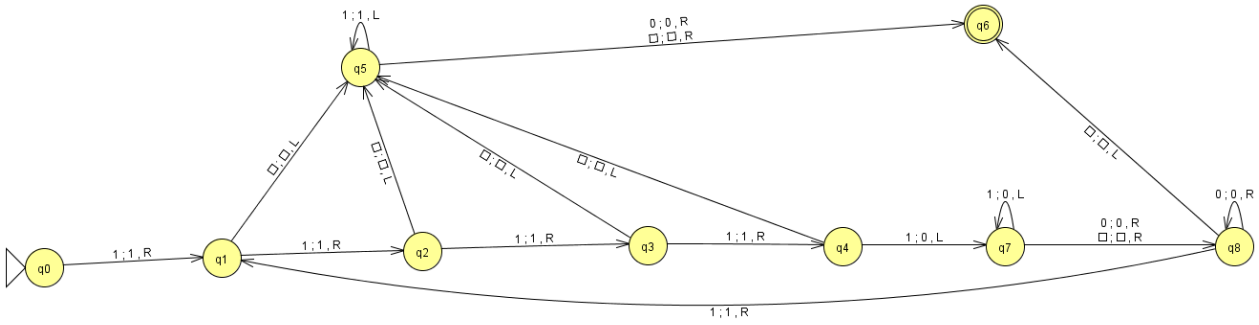


b) $f(x) = x \ mod \ 5$

Input	Output	Result
1	1	Accept
1111	1111	Accept
11111	0	Accept
1111111	11	Accept
1111111111	0	Accept
11111111111	1	Accept

Answer:

This Turing Machine going right, when it is seeing 1, until blink. After that we go back, until blink and turn number of 1`s. This case from one to four. If we see fifth unit we make it 0 (such as a different 1`s) and go back until blink, further go right until right`s blink and turn 0. For the case with six 1`s we make the same with one except, because we change one 0 to 1 and turn it. Different examples with 7, 8, 9,10 work similarly.



3. *(5 pts)* The nor of two languages is defined below:

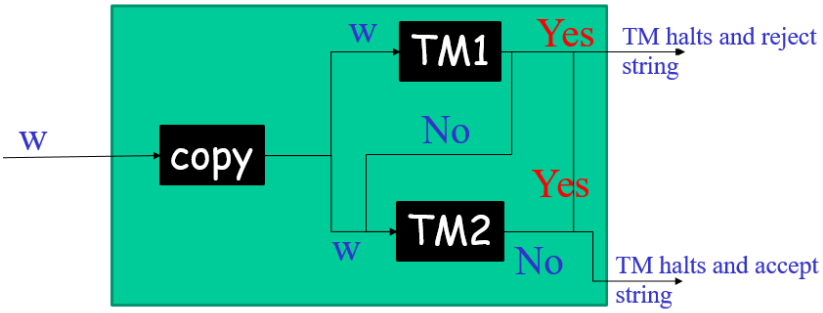
nor(L_1, L_2) = { w: w $\notin L_1$ and w $\notin L_2$}.

Prove that recursive languages are closed under the nor operation.

Answer:

The formal definition of Turing Machine is M = $\{Q, \Sigma, \Gamma, \delta, q_0, \Diamond, F\}$ where : Q – States, Σ – input alphabet , Γ – tape alphabet, $\delta - transition\ function$, $q_0 - Initiail\ state$, $\Diamond$ - blank, F – final state.

A language is recursive if some Turing machine accepts it and halts on any input string.


w
copy
w
TM1
Yes
TM halts and reject string
No
Yes
w
TM2
No
TM halts and accept string