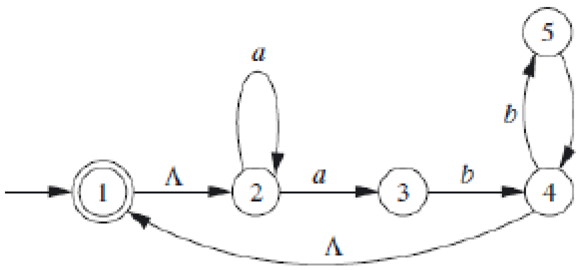
BLG311E – FORMAL LANGUAGES AND AUTOMATA

2017 SPRING

RECITEMENT 7

1) Consider the following NFA



a) Give the regular expression for this NFA heuristically

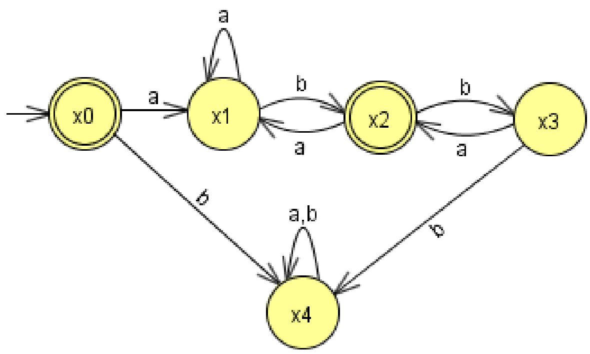
b) Systematically transform the NFA to an equivalent DFA. Draw the DFA.

Solution:

a) $(a^*ab(ba)^*)^* = (a^+b(ba)^*)^*$

b) $x_0 = R(S_1) = \{\mathbf{S_1}, S_2\}$

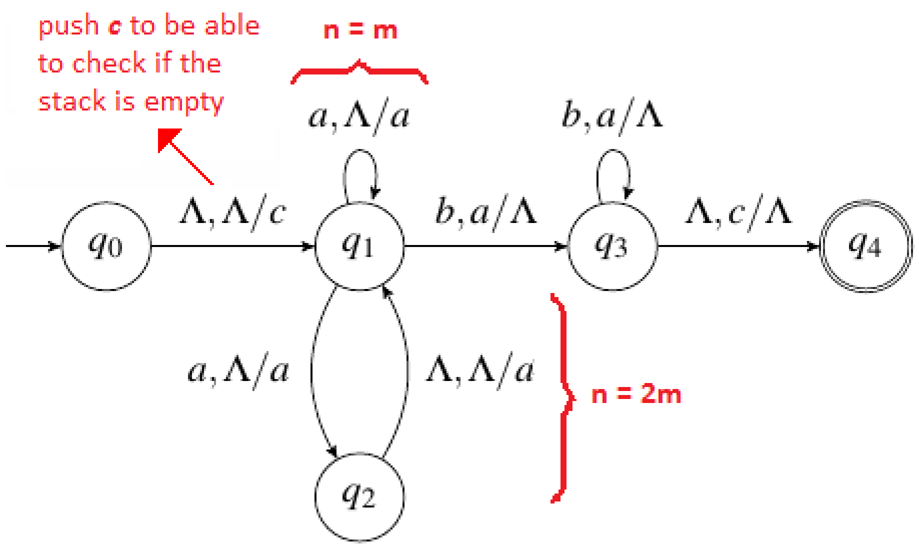
$\delta'(x_0,a) = \delta(\{S_1, S_2\},a) = R(S_2) \vee R(S_3) = \{S_2, S_3\} = x_1$
$\delta'(x_0,b) = \delta(\{S_1, S_2\},b) = \emptyset$
$\delta'(x_1,a) = \delta(\{S_2, S_3\},a) = R(S_2) \vee R(S_3) = \{S_2, S_3\} = x_1$
$\delta'(x_1,b) = \delta(\{S_2, S_3\},b) = R(S_4) = \{\mathbf{S_1}, S_2, S_4\} = x_2$
$\delta'(x_2,a) = \delta(\{S_1, S_2, S_4\},a) = R(S_2) \vee R(S_3) = \{S_2, S_3\} = x_1$
$\delta'(x_2,b) = \delta(\{S_1, S_2, S_4\},b) = R(S_5) = \{S_5\} = x_3$
$\delta'(x_3,a) = \delta(\{S_5\},a) = R(S_4) = R(S_4) = \{\mathbf{S_1}, S_2, S_4\} = x_2$
$\delta'(x_3,b) = \delta(\{S_5\},b) = \emptyset$
$\delta'(\emptyset,a) = \delta(\emptyset,b) = \emptyset = x_4$



2) (Quiz 4 Solution) Design a pushdown automaton (PDA) that recognizes the following language and draw the finite automaton that shows the state-transition relation of this PDA.

$$L = \{a^m b^n \mid 0 < m \le n \le 2m\}$$

Solution:



3) Claim if the following language can be recognized by a pushdown automaton (PDA). Prove/rationalize your claim.

$$L(G) = \{a^i b^k a^i b^k \mid i, k > 0\}$$

Solution:

The given language is not a context-free language, thus it cannot be recognized by a PDA. This can be proved using pumping lemma as follows.

Assumptions:

Suppose that L has a CFG.

Choose $u = vwxyz = a^n b^n a^n b^n$ satisfying $u \in L$ and $|u| \geq n$.

By pumping lemma:

$|wxy| \leq n$ and $|wy| > 0$ imply that the string wxy contains at least one symbol and can overlap at most two of the four contiguous groups of symbols. The following cases are possible for wy:

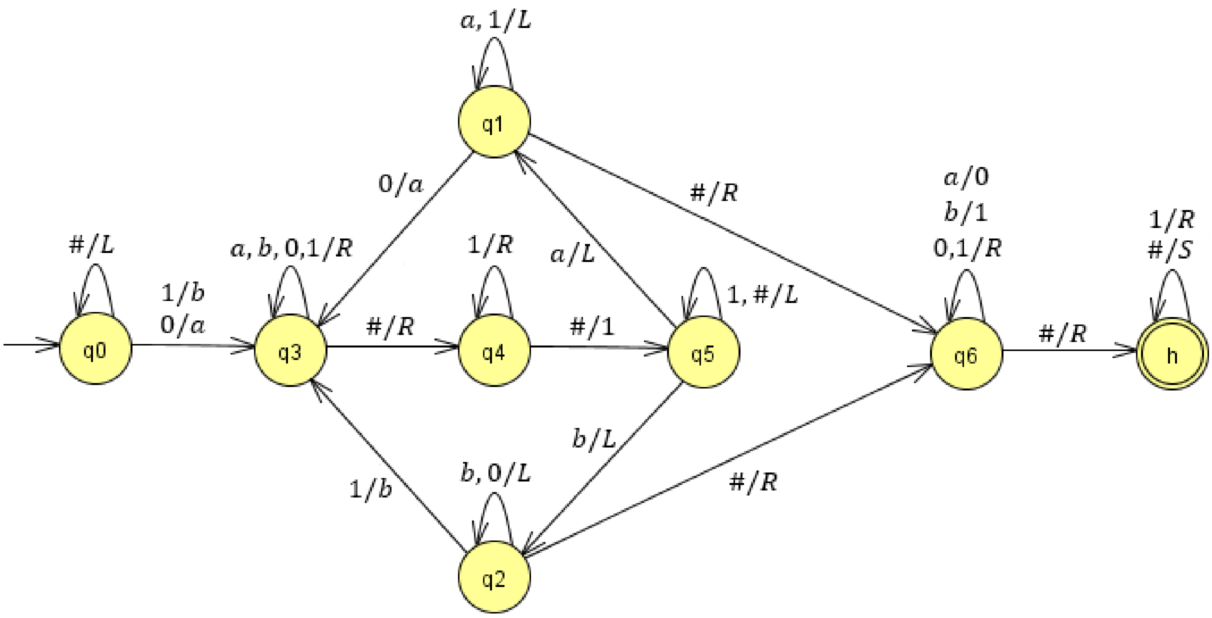
1. wy contains only a's from either the first or the second group of a's:
   $vw^m xy^m z \notin L$ for $m \neq 1$ as it will change the number of a's in one group while the other group remains the same.
2. wy contains only b's from either the first or the second group of b's:
   $vw^m xy^m z \notin L$ for $m \neq 1$ as it will change the number of b's in one group while the other group remains the same.
3. wy contains a's from the first group of a's and b's from the first group of b's:
   $vw^m xy^m z \notin L$ for $m \neq 1$ as it will change the number of a's in one group while the other group remains the same and it will also change the number of b's in one group while the other group remains the same.
4. wy contains b's from the first group of b's and a's from the second group of a's:
   $vw^m xy^m z \notin L$ for $m \neq 1$ due to the same reason in case 3.
5. wy contains a's from the second group of a's and b's from the second group of b's:
   $vw^m xy^m z \notin L$ for $m \neq 1$ due to the same reason in case 3.

4) Design a Turing Machine that can count 0's and 1's given a binary string as an input. If the input string ends with a 0, it counts the 0's in the string and writes as many 1's after the blank symbol (#) following the input. If the last symbol of the input string is 1, then your machine will count the number of 1's in the string and again will write as many 1's after the blank following the input. You can see how the machine works from the following examples. As you can see from the examples, initially the tape head is on the first blank after the input. You can assume that input will never be an empty string. (Underlined symbol represents the position of the tape head).

001110110<u>#</u> =>* 001110110#1111<u>#</u>

1101<u>#</u> =>* 1101#111<u>#</u>

Solution:



5) Design a Deterministic Turing Machine that deletes the 0s from a given binary string. Binary string will start after the first blank (#) and initially, the tape head is over the first blank after the input (underlined symbol shows tape head position). Write your TM as a **state diagram**. Consider the following examples carefully. You may use additional symbols if necessary.

$\#0011001010111011\underline{\#} \vdash *_M \#111111111\underline{\#}$

$\#0010010\underline{\#} \vdash *_M \#11\underline{\#}$

$\#0000000\underline{\#} \vdash *_M \#\underline{\#}$

Solution: