CS321 Introduction to Theory of Computation
Assignment No. 2, Due: Friday February 2, 2024

1. Suppose that a bank only permits passwords that are strings from the alphabets $\sum = \{a, b.c, d, 1, 2, 3, 4, \#, \$, \&\}$. The passwords follow the rules

   (a) The length can be 5, 6 or 7.

   (b) The first alphabet must be from $\{a, b, c, d\}$.

   (c) The last two alphabets must be from $\{1, 2, 3, 4\}$

   (d) Exactly one alphabet should be from $\{\#, \$, \&\}$.

   The set of legal passwords forms a regular language L. Construct a NFA for L.

2. Design an NFA with no more than five states for the set $\{abab^n : n \geq 0\} \cup \{aba^n : n \geq 0\}$

3. Find an NFA with four states for $L = \{a^n : n \geq 0\} \cup \{b^m a : m \geq 1\}$.

4. Convert the NFA defined by

   $\delta(q_0, a) = \{q_0, q_1\}$.
   $\delta(q_1, b) = \{q_1, q_2\}$
   $\delta(q_2, a) = \{q_2\}$
   $\delta(q_0, \lambda) = \{q_2\}$

   with initial state q_0 and final state q_2 into an equivalent DFA.

5. Show that if L is regular, so is L^R.

1. Suppose that a bank only permits passwords that are strings from the alphabets $\Sigma = \{a, b.c, d, 1, 2, 3, 4, \#, \$, \&\}$. The passwords follow the rules

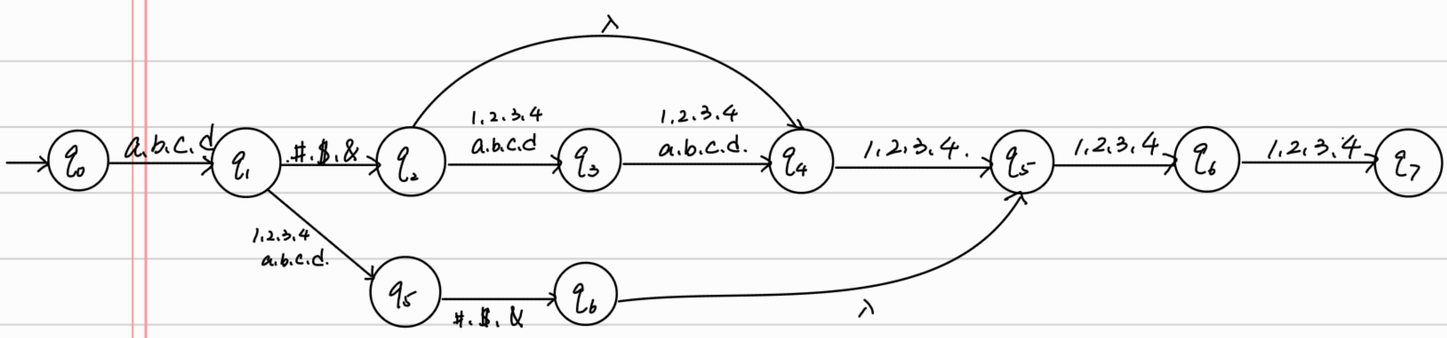
   (a) The length can be 5, 6 or 7.

   (b) The first alphabet must be from $\{a, b, c, d\}$.

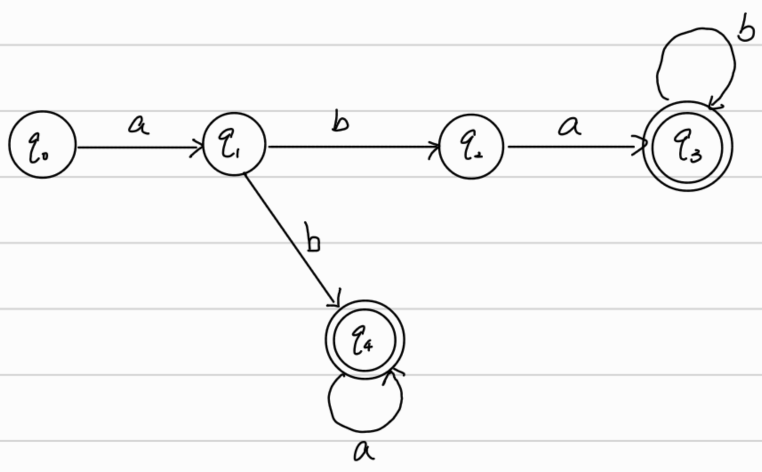
   (c) The last two alphabets must be from $\{1, 2, 3, 4\}$

   (d) Exactly one alphabet should be from $\{\#, \$, \&\}$.

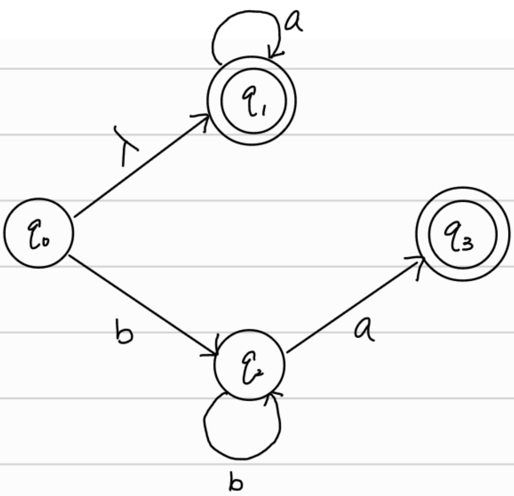
   The set of legal passwords forms a regular language L. Construct a NFA for L.



2. Design an NFA with no more than five states for the set $\{abab^n : n \geq 0\} \cup \{aba^n : n \geq 0\}$



3. Find an NFA with four states for $L = \{a^n : n \geq 0\} \cup \{b^m a : m \geq 1\}$.

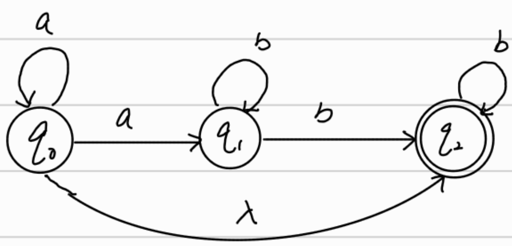


4. Convert the NFA defined by

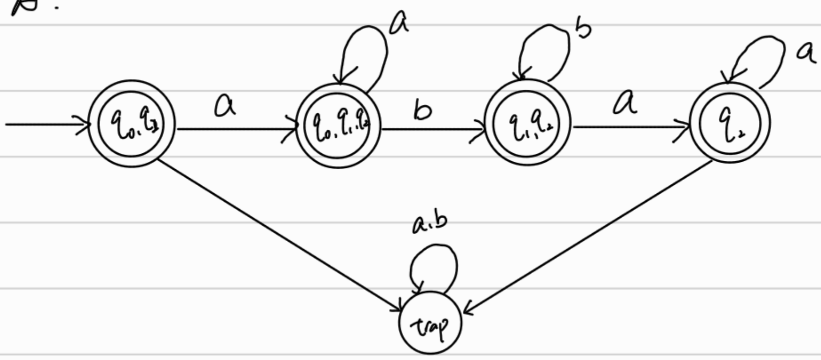
$\delta(q_0, a) = \{q_0, q_1\}$.
$\delta(q_1, b) = \{q_1, q_2\}$
$\delta(q_2, a) = \{q_2\}$
$\delta(q_0, \lambda) = \{q_2\}$

with initial state q_0 and final state q_2 into an equivalent DFA.

Given NFA. :



DFA.



5. Show that if L is regular, so is L^R.

① Express L as a regular expression : Since L is regular, there exists a regular expression r

② Reverse the order of the symbols and operations in r to get a new regular expression r'

③ Show that r' generates LR, it can be proven that for any regular expression r. $L(r') = (L(r))^R$, The reversed regular expression r' generates the reversal of the language generated by r

④ Since r' generates LR, and regular expressions are closed under reversal, So we can calculate that LR is also regular.