Middle East Technical University Department of Computer Engineering

CENG 280

Formal Languages and Abstract Machines

Spring 2022-2023

Homework 2

Name Surname: Berk Ulutaş
Student ID: 2522084

Answer for Q1

a. $(a(b+c)^*a+b+aa)(a+b)^*$

b.
A=: 0
B=: 1
C=: 0,1
D=: 2
E=: 1
F=: 0,2

Answer for Q2

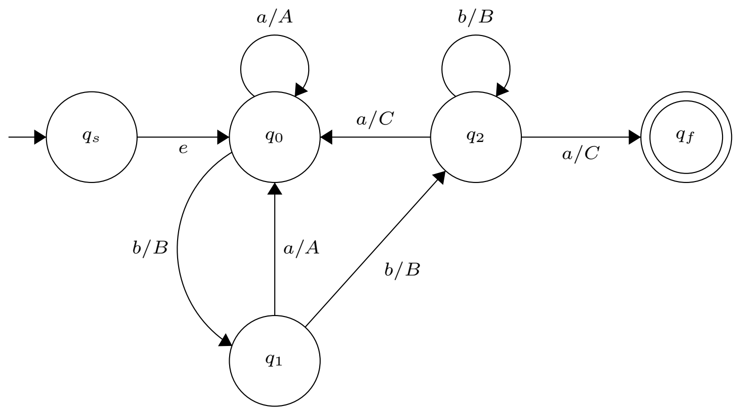
a. The algorithm which is used for convert DFA/NFA to a Regular Expression (State elimination).

b.

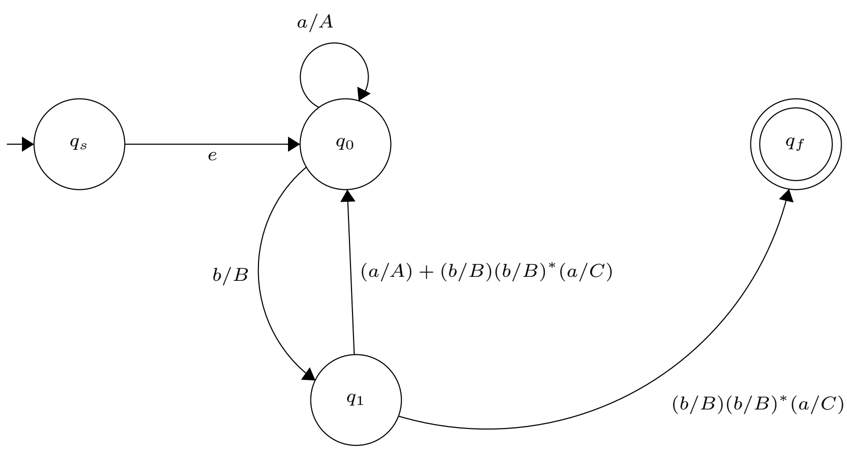
- Create a new starting state and connect it the previous starting state with empty transition. If machine do not read anything it cannot output anything.
- Create a final state and connect all states to the final state with empty transition. Empty transitions do not output anything. Since, the input can end in any state. all outputs will be in the language regardless of which state it ends in.
- After this modifications, we have an NFA-like machine. If we can treat (input/output) transitions to just like a normal transition. We can convert this NFA-like machine to a regular expression which gives the set of (input/output) strings in regular expression format that can be produced by a Mealy Machine. We can easily produce just set of output strings by ignoring inputs in (input/output).

c. Create two new states q_s and q_f. Add empty transition from q_s to q_0 and make q_s new starting state. Create 'C' transition from q_2 to q_f and make q_f accepting state.

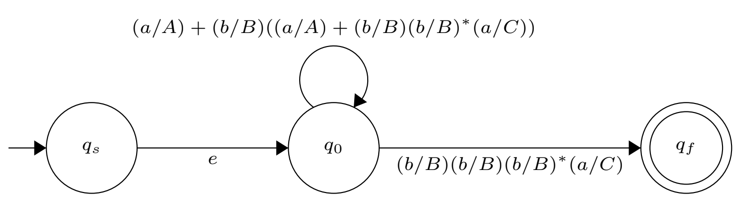
Step 1



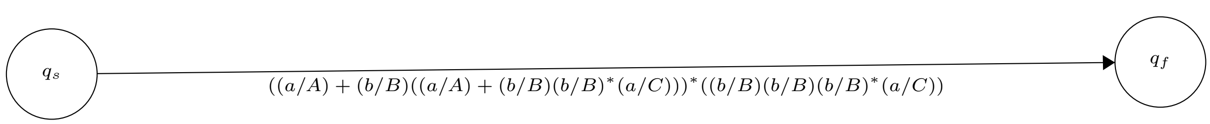
Step 2 eliminate q_2



Step 3 eliminate q_1



Step 4 eliminate q_0



Set of output strings of M_1 which are ending with C as a regular expression:

$$(A + B(A + BB^*C))^*BBB^*C$$

Answer for Q3

P machine I/O is $[(a/A) + (b/B)(a/A) + (b/B)(b/B)(a/C)]^* + [(b/B)(b/B)]^*$. Let's analyze which input/output expression we can accept, and write a general regular expression for strings we can accept with N_2 or N_3.

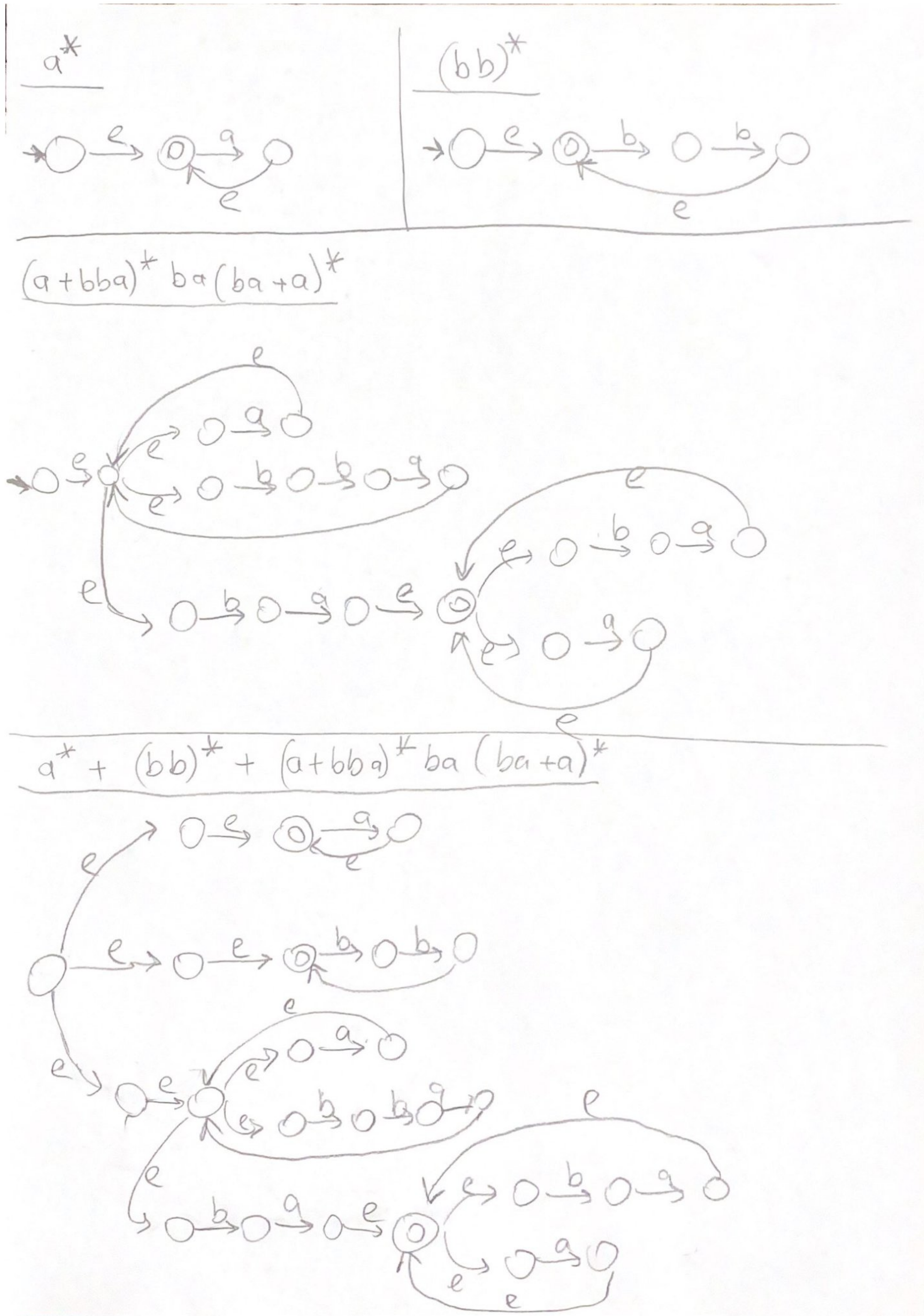
Start with N_2 machine

- We accept strings that can reach q_2 state.
- A^* or $(BB)^*$ strings can be generated by P machine and accepted by N_2 machine.
- To generate A^* or $(BB)^*$ we need $(a)^* + (bb)^*$ as input.

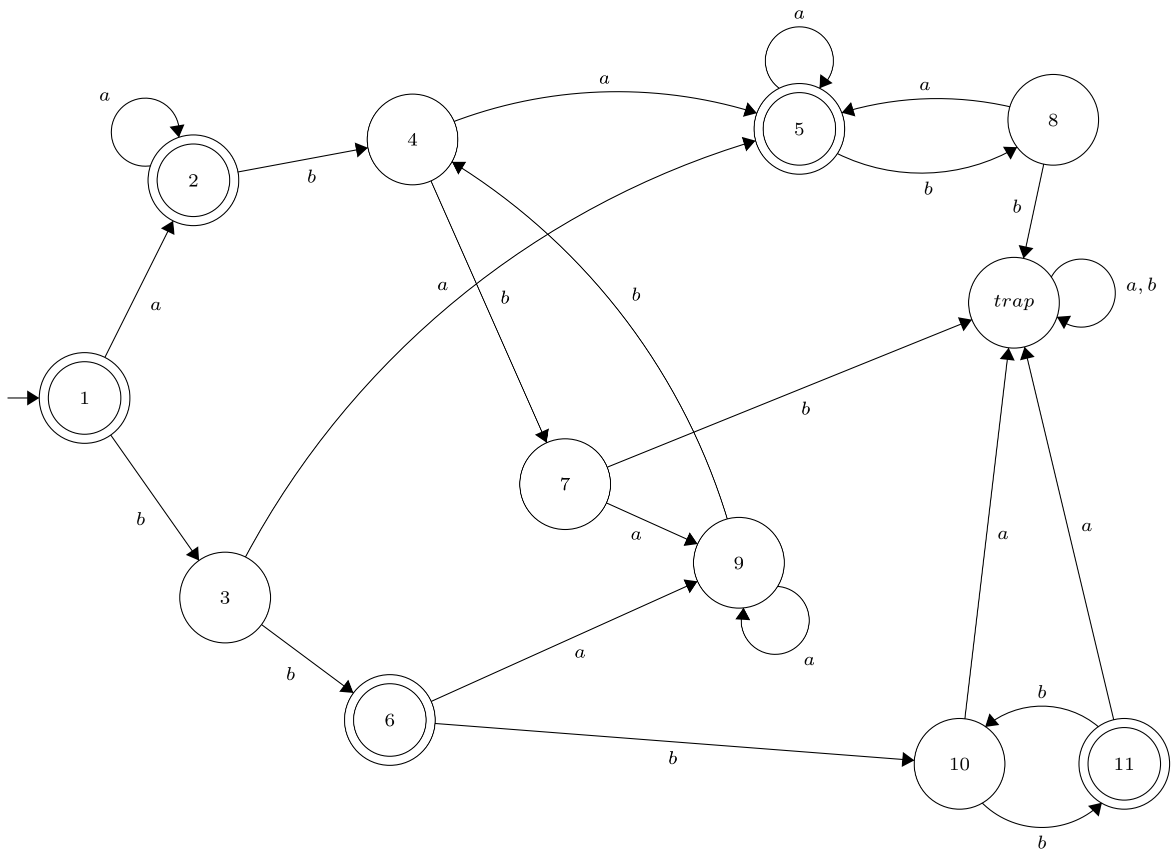
For N_3 machine

- We accept strings that can reach q_3 state. To reach the q_3 state, it is necessary to be in the q_0 or q_4 state. However, a C output is required to reach q_3 while in q4. This is not possible as the P machine does not produce an output starting with C. In this case, we cannot reach from q_4 to q_3.
- Now let's look at all the cases that can be produced by the machine P which we can reach q_0 from the starting state. $(A + BBC)^*$ to generate this output we need $(a + bba)^*$ as a input.
- We need output B to go from state q_0 to q_3. Output B can be produced by machine P in 2 ways either $(b/B)(a/A)$ or $(b/B)(b/B)(a/C)$. But we cannot go with $(b/B)(b/B)(a/C)$ case since there is no C transition in q_3. In this case output should be $(A + BBC)^*(BA)$ to create this output we need $(a + bba)^*ba$ as input.
- We are now in the q_3 state of the N_3 machine. We can continue to stay in the q_3 state by taking the $(A + BA)^*$ output. $(a + ba)^*$ input is required to get this output.
- As a result, from the start $(A + BBC)^*BA(A + BA)^*$ output can be produced by machine P and accepted by N_3. To produce this output we need $(a + bba)^*ba(a + ba)^*$ as input.

After all these steps the regular expression $a^* + (bb)^* + (a + bba)^*ba(a + ba)^*$ is inputs accepted by the whole machine. We can convert this regular expression to a DFA. First create a NFA from regular expression then convert it to a DFA.



Since the steps of converting from NFA to DFA take too long, I could not draw all of them one by one. The latest version of DFA is below


1
2
3
4
5
6
7
8
9
10
11
trap
a
b
a, b