Exercise 1

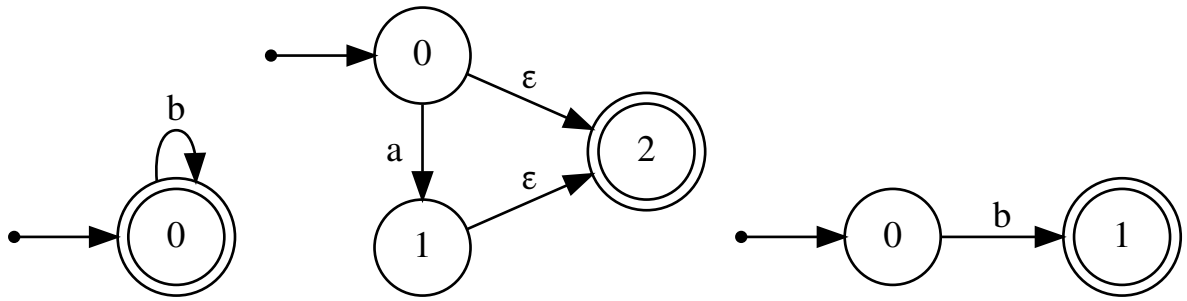
Halvor Linder Henriksen

1 Regular languages, NFAs and DFAs

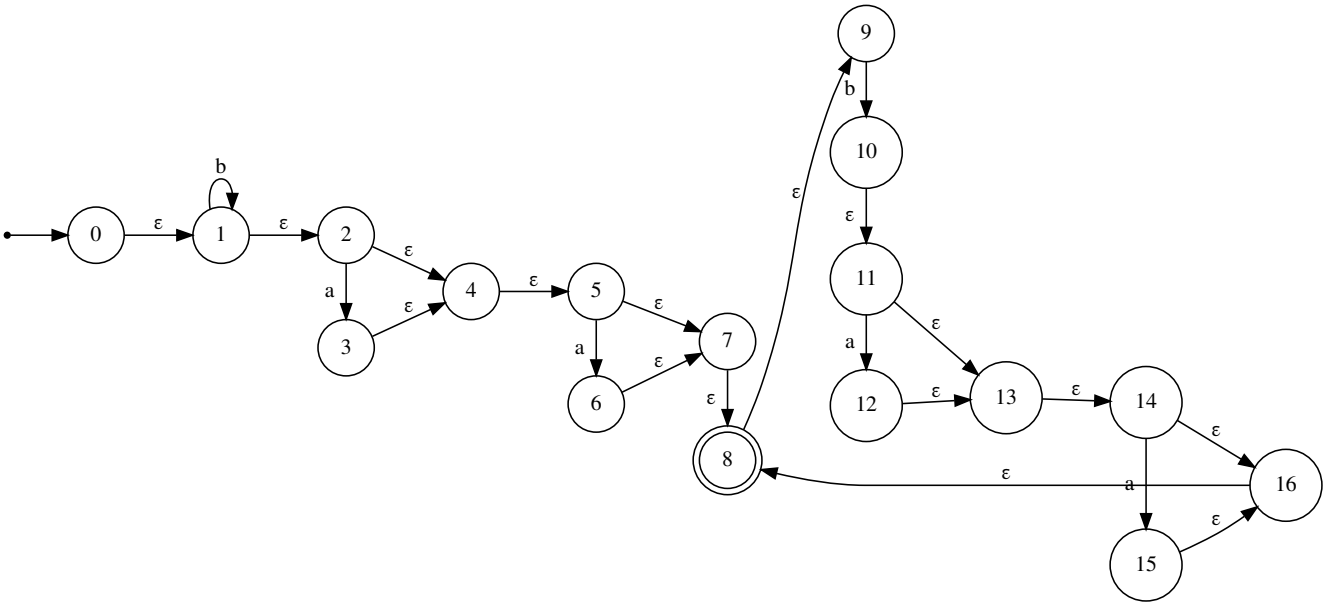
1.1

b*a?a?(ba?a?)*

1.2



State machines for b*, a?, b respectively.



The full NFA obtained form using the MYT-algorithm on the regular expression.

1.3

epsilon-closure(0) = {0,1,2,4,5,7,8,9}

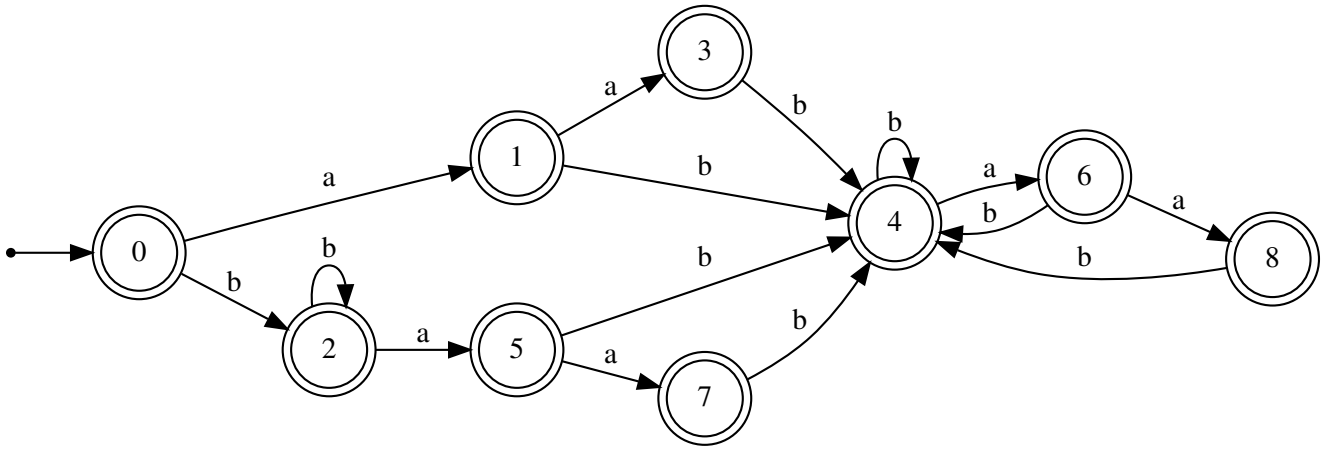
State	a	b
{0,1,2,4,5,7,8,9} (0)	{3,4,5,6,7,8,9} (1)	{1,2,4,5,7,8,9,10,11,13,14,16} (2)
{3,4,5,6,7,8,9} (1)	{6,7,8,9} (3)	{10,11,13,14,16,8} (4)
{1,2,4,5,7,8,9,10,11,13,14,16} (2)	{3,4,5,6,7,8,9,13,14,15,16} (5)	{1,2,4,5,7,8,9,10,11,13,14,16} (2)
{6,7,8,9} (3)	{} (9)	{10,11,13,14,16,8} (4)
{10,11,13,14,16,8} (4)	{12,13,14,15,16,8,9} (6)	{10,11,13,14,16,8} (4)
{3,4,5,6,7,8,9,13,14,15,16} (5)	{6,7,8,9,15,16} (7)	{10,11,13,14,16,8} (4)
{12,13,14,15,16,8,9} (6)	{15,16,8,9} (8)	{10,11,13,14,16,8} (4)
{6,7,8,9,15,16} (7)	{} (9)	{10,11,13,14,16,8} (4)
{15,16,8,9} (8)	{} (9)	{10,11,13,14,16,8} (4)

The above table provides the iterations of the subset construction algorithm

Transition table for the resulting DFA:

State	a	b
0	1	2
1	3	4
2	5	2
3	9	4
4	6	4
5	7	4
6	8	4
7	9	4
8	9	4
9	9	9

Graph representation:



1.4

To find the minimum DFA, we can find the N-equivalences until fixed point.

N = 0

The 0 equivalence is given by the non-final and final states

{9} {0,1,2,3,4,5,6,7,8}

N = 1

{9} {0,1,2,4,5,6} {3,7,8}

N = 2

{9} {0,2,4} {1,5,6} {3,7,8}

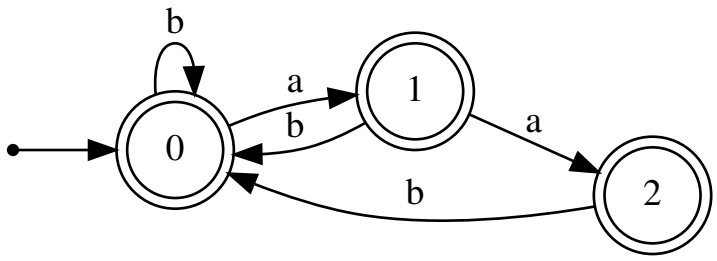
N = 3

{9} {0,2,4} {1,5,6} {3,7,8}

We see that a fixed point is reacherd in the iterations, giving the states of the minimum DFA.

Transition table for the resulting DFA:

State	a	b
0	1	0
1	2	0
2	3	0
3	3	3



Graph representation

1.5

In the DFA, all final states could be turned into non-final states and vice versa to produce a DFA recognizing the "opposite" language.

Regex that matches the expression:

(b*a*)*(aaa)(b*a*)*

The DFA was the easiest

2 DFA for a small language

2.1

<integer> ::= "-"?[0-9]+

<statement> ::= ("dx="<integer> | "dy="<integer> | "go") "\n"

2.2