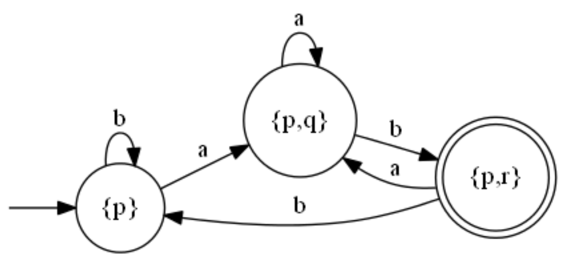
# Informatics 2A 2018–19.
# Tutorial Sheet 1 - SOLUTIONS

Mary Cryan

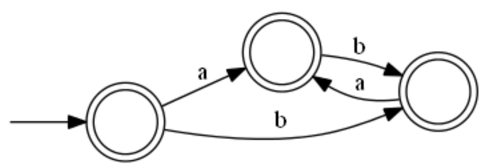
1. Three subsets of $\{p, q, r\}$ suffice:




The best way to produce this is to start with the DFA start state $\{p\}$, and then explore the result of applying $a$ and $b$ transitions to states so far constructed, until no new DFA states (i.e. subsets of $\{p, q, r\}$) arise.

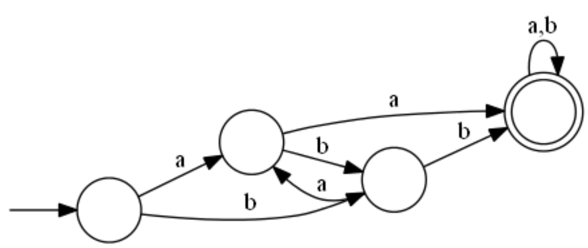
2. The NFAs given here are the 'simplest possible' – however, many other choices of regular expresssions would be equally reasonable.

   (a) NFA:



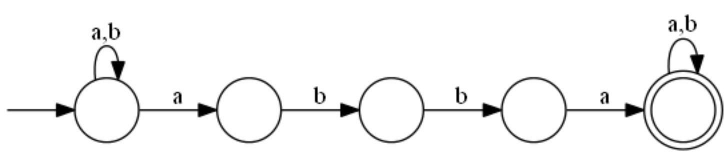
   Regular Expression: $(a + \epsilon)(ba)^*(b + \epsilon)$

   (b) NFA:



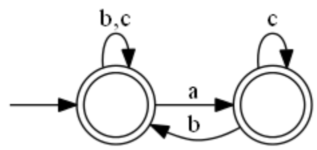
   Regular Expression: $(a + b)^*(aa + bb)(a + b)^*$

   (c) NFA:



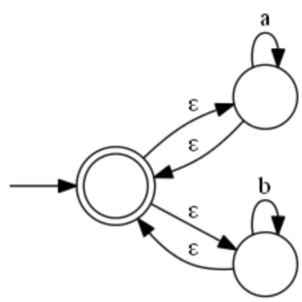
Regular Expression: $(a+b)^*abba(a+b)^*$

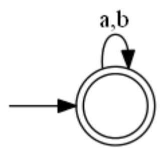
(d) NFA:




Regular Expresssion: $\mathcal{Z}(\epsilon + a(\mathcal{Z}b\mathcal{Z}a)^*)\mathcal{Z}$, where $\mathcal{Z} = (b+c)^*$

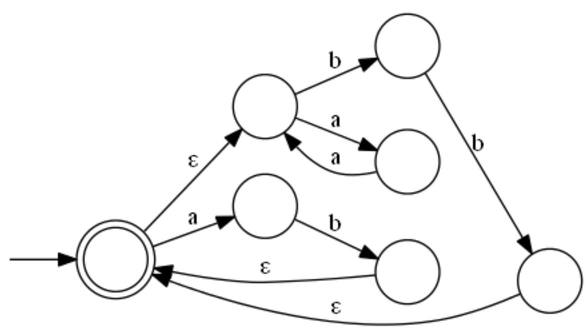
(e)



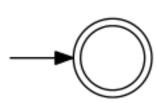

Actually 

would accept the same language, but I wouldn't regard it as following the structure of the regular expression.

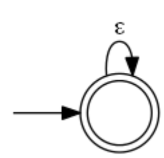
(f)




(g)

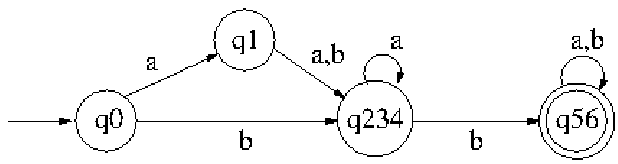


or



[Note that $\epsilon$ is the only string accepted.]

3. The minimized DFA is:




Please obtain this using the algorithm presented in Lecture 5. I have suggested inserting the separating strings discovered by the algorithm into the chart, rather than just ticks. In this case, the resulting chart will look like this:

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| $q0$ | | | | | | | |
| $q1$ | $ab$ | | | | | | |
| $q2$ | $b$ | $b$ | | | | | |
| $q3$ | $b$ | $b$ | . | | | | |
| $q4$ | $b$ | $b$ | . | . | | | |
| $q5$ | $\epsilon$ | $\epsilon$ | $\epsilon$ | $\epsilon$ | $\epsilon$ | | |
| $q6$ | $\epsilon$ | $\epsilon$ | $\epsilon$ | $\epsilon$ | $\epsilon$ | . | |
| | $q0$ | $q1$ | $q2$ | $q3$ | $q4$ | $q5$ | $q6$ |

Notice that the $\epsilon$ entries get added in 'Round 0' of the algorithm, the $b$ entries in Round 1, and the $ab$ entry in Round 2, when we detect a pair which goes under $a$ to a pair that already has a $b$ entry.

As an aside, the minimal DFA can also be obtained in an *ad hoc* way by observing the following.

- States $q5$ and $q6$ may be collapsed, since **any** string takes us from either of these to an accepting state.
- States $q2$,$q3$ and $q4$ may all be collapsed, since the strings that takes us from these to an accepting state are those matching $a^*b(a+b)^*$.
- Any other pair of states are differentiated by their behaviour on at least one of the strings: $a$, $\epsilon$, $b$, $ab$.

4. (a) Different: 01 is in $\mathcal{L}((0+1)^*)$ but not $\mathcal{L}(0^*+1^*)$.

   (b) The same: by the third identity with $a=1$, $b=20$,

$$\begin{array}{rrcl} & (120)^*1 & = & 1(201)^* \\ \text{where} & 0(120)^*12 & = & 01(201)^*2 \end{array}$$

   (c) Different: 0 is in $\mathcal{L}((0^*1^*)^*)$ but not $\mathcal{L}((0^*1)^*)$.

   (d) The same: $(01+0)^*0$
   $= (0(1+\epsilon))^*0$ by second identity and '$a\epsilon = a$'.
   $= 0((1+\epsilon)0)^*$ by third identity.
   $= 0(10+0)^*$ by first identity and '$\epsilon a = a$'.

5. (a) The required language is $X_p$, where

$$X_p = aX_p + bX_q \quad (1)$$
$$X_q = (a+b)X_q + \epsilon \quad (2)$$

Solving these:

| | | |
|---|---|---|
| $X_q$ | $= (a+b)^*$ | from (2) by Arden's rule |
| $X_p$ | $= aX_p + b(a+b)^*$ | substituting in (1) |
| $X_p$ | $= a^*b(a+b)^*$ | by Arden's rule. |

(b) The required language is $X_p$, where

$$X_p = bX_p + aX_q + \epsilon \quad (3)$$
$$X_q = bX_p + aX_r \quad (4)$$
$$X_r = (a+b)X_q \quad (5)$$

Solving these:

| | | |
|---|---|---|
| $X_q$ | $= bX_p + a(a+b)X_q$ | substituting (3) in (2) |
| $X_q$ | $= (a(a+b))^*bX_p$ | by Arden's rule |
| $X_p$ | $= bX_p + a(a(a+b))^*bX_p + \epsilon$ | substituting in (1) |
| | $= (b + a(a(a+b))^*b)X_p + \epsilon$ | by distributivity law |
| | $= (b + a(a(a+b))^*b)^*$ | by Arden's rule. |

6. (a) The DFA has 21 states in all (I won't draw it here). There are 16 states corresponding to all possible scorelines $x/y$ where $x, y \in \{0, 15, 30, 40\}$. (Except that the state for 40/40 is known as Deuce.) The start state is 0/0. The transitions between the above states are as expected, e.g. from 15/30 there is an $f$-transition to 30/30 and an $m$-transition to 15/40.

There is also state 'Advantage Federer' with an $f$-transition from Deuce and an $m$-transition to Deuce. There is a state 'Game Federer' with $f$-transitions from the states 40/0, 40/15, 40/30 and Advantage Federer. Similarly on Murray's side, except that 'Game Murray' is designated as an accepting state.

Finally, there should also be a 'garbage state' which we enter (and stay in) if input symbols continue after a game has been completed.

(b) This DFA is not minimal. The main thing to note is that the states 40/30 and Advantage Federer can be identified: from either of these states, $f$ would take us to Game Federer and $m$ would take us to Deuce. Likewise, 30/40 and Advantage Murray can be identified.

Less interestingly, we can identify Game Federer with the garbage state (but this is just a consequence of our biased decision only to accept wins by Murray).

(c) *Yes*, an entire tennis match can indeed be modelled using a DFA.

One can build such a DFA in a hierarchical way. First, we can model a set as a sequence of games (say $F$ for Game Federer, $M$ for Game Murray), and build a DFA over $\{F, M\}$ to process complete sets. We then refine this by replacing each state along with its $F$ and $M$ transitions by (essentially) a complete copy of the DFA constructed in (a), or by a suitably adapted version of this if a tiebreak is required. This gives a DFA that processes complete sets at the level of individual points. Repeating the process, we now build a DFA that models a complete match at the level of sets, and then insert lots of copies of the DFA for a single set to obtain a DFA for a complete match at the level of points.