Technische Universität München **Winter term 21/22**
I7
Prof. J. Esparza / A. R. Balasubramanian / C. Weil-Kennedy

Automata and Formal Languages — Exercise Sheet 2

Exercise 2.1
Consider the regular expression $r = (b + bab)^*$.

(a) Convert r into an equivalent NFA-ε A.

(b) Convert A into an equivalent NFA B. (It is not necessary to use algorithm *NFAεtoNFA*)

(c) Convert B into an equivalent DFA C.

(d) By inspecting B, merge two states to get an equivalent DFA D with less states *if possible* (depending on how you answered previous questions, this may not be possible). No algorithm needed.

(e) Convert D into an equivalent regular expression r'.

(f) Prove formally that $L(r) = L(r')$.

Exercise 2.2
Prove that if L is a finite language, then the complement of L is a regular language.

Exercise 2.3
Let $\Sigma = \{a, b, c\}$. Show that the language described by the regular expression $(((b+c)^*a + c^*) + (bc^*)^*)^*$ is the set of all words over Σ.

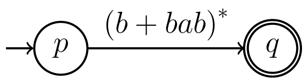
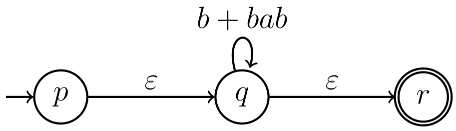
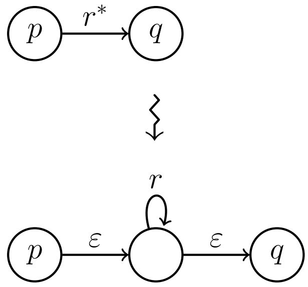
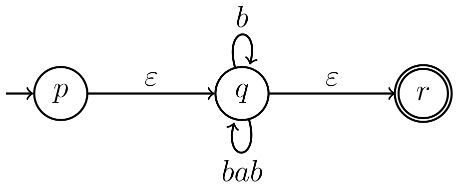
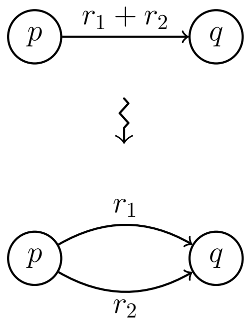
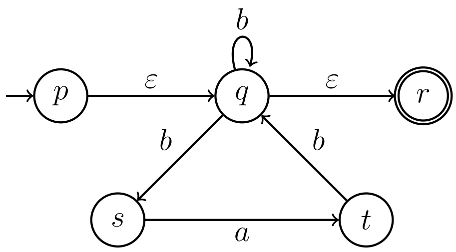
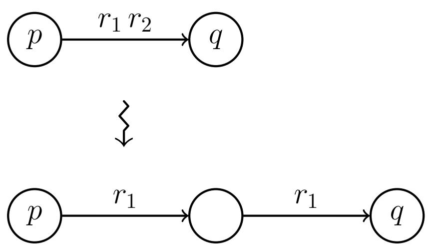
Exercise 2.4
Let $n \geq 1$ be some natural number and let $\Sigma = \{a : 1 \leq a \leq n\}$. Consider the following language over Σ:

$$L = \{aaa : a \in \Sigma\}$$

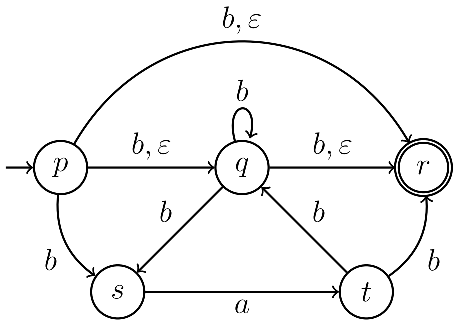
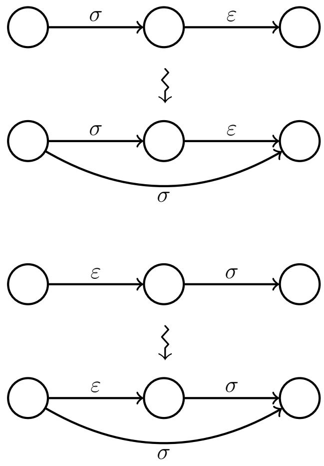
- Show that there is a NFA with $2n + 2$ states which recognizes L.
- Show that any NFA recognizing L must have at least $2n + 2$ states.

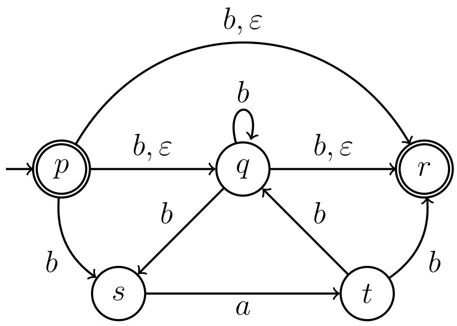
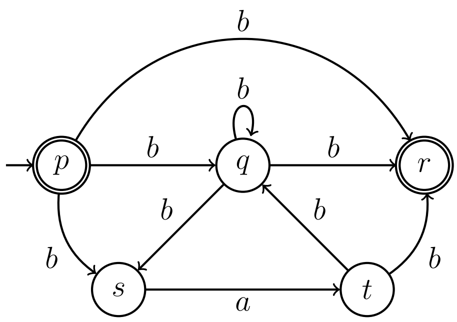
Solution 2.1
There are different correct answers for the following exercises, the following is one possible set of answers.

(a)

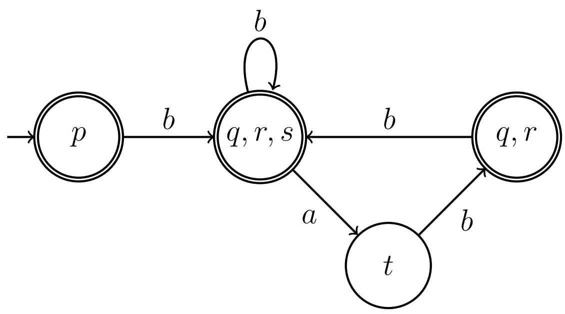
Iter.	Automaton obtained	Rule applied
1		Initial automaton from reg. expr.
2		
3		
4		

(b)

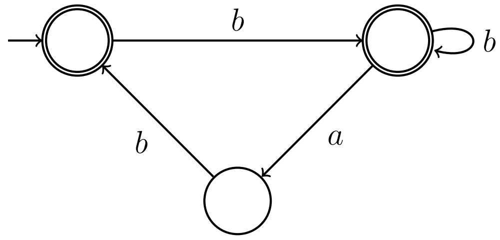
<table>
<tr><th>Iter.</th><th>Automaton obtained</th><th>Rule applied</th></tr>
<tr><td>1</td><td>
</td><td>

where $\sigma \in \Sigma \cup \{\varepsilon\}$</td></tr>
<tr><td>2</td><td>
</td><td>Initial states that can reach a final state through ε-transitions are made final.</td></tr>
<tr><td>3</td><td>
</td><td>Remove ε-transitions.
Remove states non reachable from initial state.</td></tr>
</table>

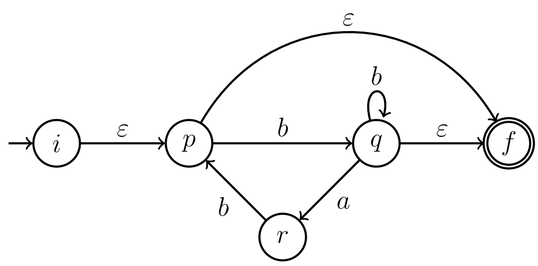
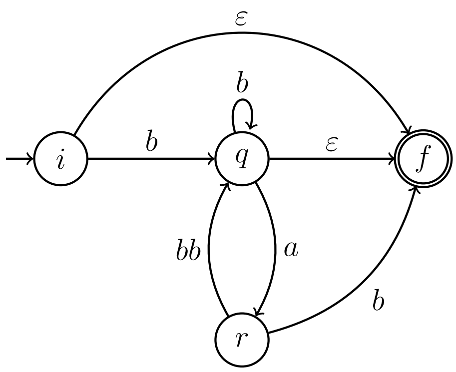
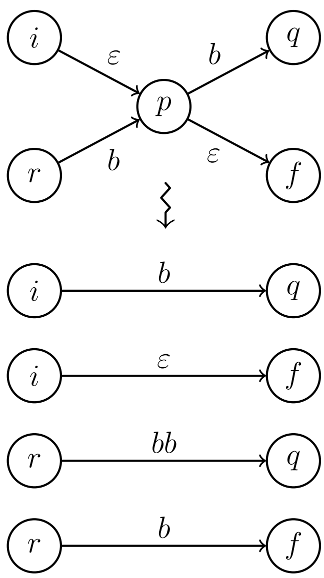
(c)

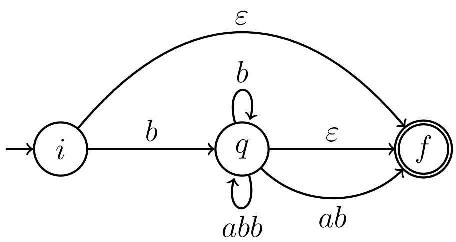
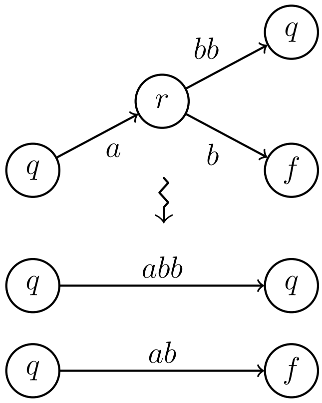
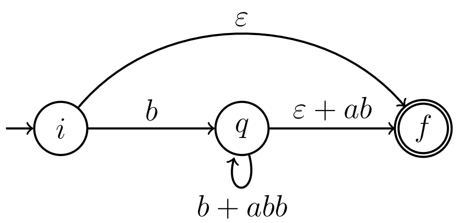
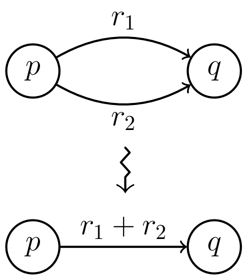
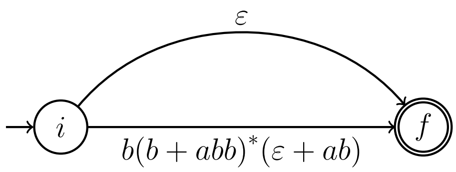
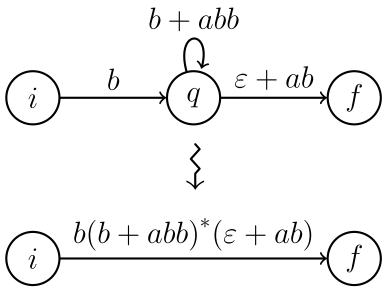
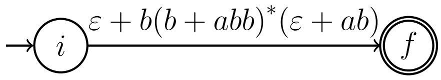
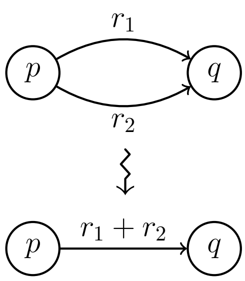


(d) States $\{p\}$ and $\{q, r\}$ have the exact same behaviours, so we can merge them. Indeed, both states are final and $\delta(\{p\}, \sigma) = \delta(\{q, r\}), \sigma)$ for every $\sigma \in \{a, b\}$. We obtain:



(e)

Iter.	**Automaton obtained**	**Rule applied**
1		Add single initial and final states.
2		

3		
4		
5		
6		
7	$\varepsilon + b(b + abb)^*(\varepsilon + ab)$	Extract regular expression from the unique transition.

(f) Let us first show that $b(b + abb)^i = (b + bab)^i b$ for every $i \in \mathbb{N}$. We proceed by induction on i. If $i = 0$, then the claim trivially holds. Let $i > 0$. Assume the claims holds at $i - 1$. We have

$$\begin{aligned}
b(b + abb)^i &= b(b + abb)^{i-1}(b + abb) \\
&= (b + bab)^{i-1} b(b + abb) && \text{(by induction hypothesis)} \\
&= (b + bab)^{i-1}(bb + babb) && \text{(by distributivity)} \\
&= (b + bab)^{i-1}(b + bab)b && \text{(by distributivity)} \\
&= (b + bab)^i b.
\end{aligned}$$

This implies that

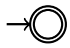
$$b(b + abb)^* = (b + bab)^* b. \tag{1}$$

We may now prove the equivalence of the two regular expressions:

$$
\begin{aligned}
\varepsilon + b(b+abb)^*(\varepsilon + ab) &= \varepsilon + (b+bab)^* b(\varepsilon + ab) && \text{(by (1))}\\
&= \varepsilon + (b+bab)^*(b+bab) && \text{(by distributivity)}\\
&= \varepsilon + (b+bab)^+ \\
&= (b+bab)^*. && \square
\end{aligned}
$$

Solution 2.2
Suppose L is a finite language. We shall first show that L is a regular language, by providing an NFA for L.

Let the alphabet of L be Σ and let $L = \{w_1, \dots, w_n\}$. For each w_i, we will construct an NFA A_i that accepts only the word w_i. If $w_i = \epsilon$ then the following NFA satisfies the required property:



Suppose w_i is not the empty word. Let $w_i = a_1, a_2, \dots, a_m$. Then the following NFA satisfies the required property:

Hence we have an NFA $A_i := (Q^i, \Sigma, \delta^i, Q_0^i, F^i)$ for each word w_i. Now, we will construct an NFA A which recongnizes the "union" $L = \cup_{1 \leq i \leq n} w_i$. Let $Q := \cup_{1 \leq i \leq n} Q^i$, $Q_0 := \cup_{1 \leq i \leq n} Q_0^i$, $F := \cup_{1 \leq i \leq n} F^i$. Further, let $\delta : Q \times \Sigma \to 2^Q$ be the function given by $\delta(q, a) = \delta^j(q, a)$ for every $q \in Q^j$ and let $A := (Q, \Sigma, \delta, Q_0, F)$. Then, A is an NFA which recongizes the language L.

Hence, we have shown that if L is a finite language, then it is regular. Hence, there must be a DFA $B = (Q, \Sigma, \delta, Q_0, F)$ such that B recognizes the language L. Consider the DFA $\overline{B} = (Q, \Sigma, \delta, Q_0, Q \setminus F)$ obtained from B by "swapping" the final and non-final states of B. By construction, $\overline{B}$ accepts a word if and only if it is rejected by B and hence $\overline{B}$ recognizes the complement of the language L.

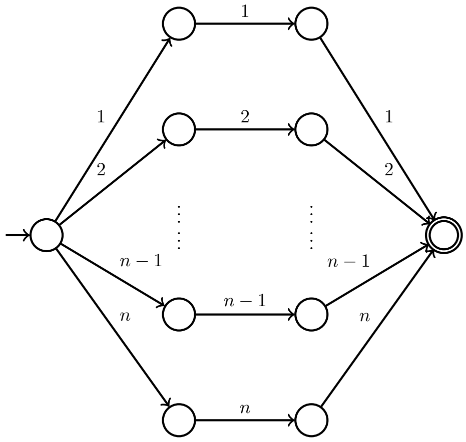
Solution 2.3
Let $r := (((b+c)^*a + c^*) + (bc^*)^*)^*$. Let $w = a_1, a_2, \dots, a_n$ be any word over Σ. We have to show that $w \in L(r)$.

Let $r' := ((b+c)^*a + c^*) + (bc^*)^*$. We will first show that each $a_i \in L(r')$. Indeed, if $a_i = a$, then $a \in L((b+c)^*a) \subseteq L(r')$. If $a_i = b$, then $b \in L((bc^*)^*) \subseteq L(r')$. Finally, if $a_i = c$ then $c \in L(c^*) \subseteq L(r')$. Hence, for each i, the letter $a_i \in L(r')$.

Notice that $L(r) = (L(r'))^*$. Since each $a_i \in L(r')$, it follows that $w \in L(r)$. Hence, we have shown that any word over Σ is included in $L(r)$, which is what we wanted to prove.

Solution 2.4

- The following is an NFA with $2n + 2$ states which recognizes L.



- We shall now show that any NFA recognizing L must have at least $2n+2$ states.

  Let A be any NFA recognizing L.

  For every $a \in \Sigma$, let $q_0^a, q_1^a, q_2^a, q_3^a$ be an accepting run of the word aaa over the NFA A. We claim that if $a \neq b$, then $q_i^a \neq q_j^b$ for any $i, j \in \{1, 2\}$. Indeed if $q_1^a = q_1^b$ (resp. $q_2^a = q_2^b$) then the word abb (resp. aab) has an accepting run given by $q_0^a, q_1^a, q_2^b, q_3^b$ (resp. $q_0^a, q_1^a, q_2^a, q_3^b$). On the other hand, if $q_1^a = q_2^b$ (resp. $q_2^a = q_1^b$) then the word ab (resp. $aabb$) has an accepting run given by q_0^a, q_1^a, q_3^b (resp. $q_0^a, q_1^a, q_2^a, q_2^b, q_3^b$). It then follows that the NFA A must have at least $2n$ states.

  We now claim that for any $a \in \Sigma$ and any $i \in \{1, 2\}$, the state q_i^a cannot be an initial or a final state. Indeed, if q_1^a (resp. q_2^a) is a final state, then the word a (resp. aa) is accepted by A. On the other hand, if q_1^a (resp. q_2^a) is an initial state, then the word aa (resp. a) is accepted by A. Hence, there is at least one initial state and one final state of A which is not in the set $\{q_i^a : i \in \{1, 2\}, a \in \Sigma\}$.

  Notice that no initial state of A can be a final state, as otherwise A would accept ϵ. It follows that there are at least two states which are not in the set $\{q_i^a : i \in \{1, 2\}, a \in \Sigma\}$. Hence, A has at least $2n+2$ states.