Technische Universität München | **Winter term 2018/19**
I7
Prof. J. Esparza / S. Sickert | **25.10.2018**

Automata and Formal Languages — Homework 2

Due 30.10.2018

Exercise 2.1
Consider the regular expression $r = (a + ab)^*$.

(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*)

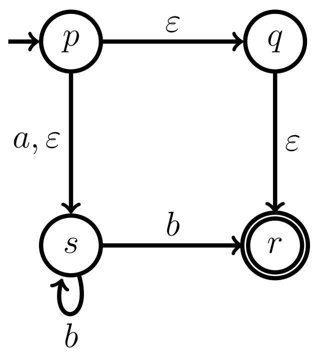
(c) Convert B into an equivalent DFA C.

(d) By inspecting B, give an equivalent minimal DFA D. (No algorithm needed).

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

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

Exercise 2.2
Convert the following NFA-ε to an NFA using the algorithm *NFAεtoNFA* from the lecture notes (see Sect. 2.3, p. 33). You may verify your answer with the Python program `nfa-eps2nfa`.



Exercise 2.3
For every $n \in \mathbb{N}$, let $L_n = \{w \in \{0,1\}^* : |w| \geq n \text{ and } w_{|w|-n+1} = 1\}$.

(a) Exhibit an NFA with $\mathcal{O}(n)$ states that accepts L_n.

(b) Exhibit a DFA with $\Omega(2^n)$ states that accepts L_n.

(c) Show that any DFA that accepts L_n has at least 2^n states.

Exercise 2.4

Recall that a nondeterministic automaton A accepts a word w if at least one of the runs of A on w is accepting. This is sometimes called the *existential* accepting condition. Consider the variant in which A accepts w if *all* runs of A on w are accepting (in particular, if A has no run on w then it accepts w). This is called the *universal* accepting condition. Notice that a DFA accepts the same language with both the existential and the universal accepting conditions.

Intuitively, we can visualize an automaton with universal accepting condition as executing all runs in parallel. After reading a word w, the automaton is simultaneously in all states reached by all runs labelled by w, and accepts if all those states are accepting.

Give an algorithm that transforms an automaton with universal accepting condition into a DFA recognizing the same language. This shows that automata with universal accepting condition recognize the regular languages.

Exercise 2.5

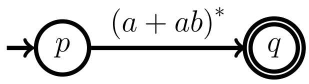
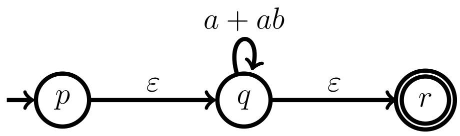
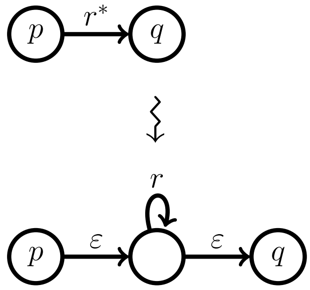
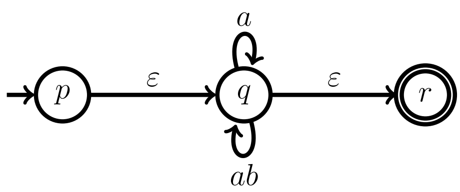
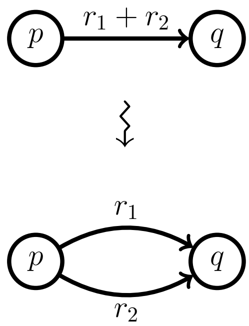
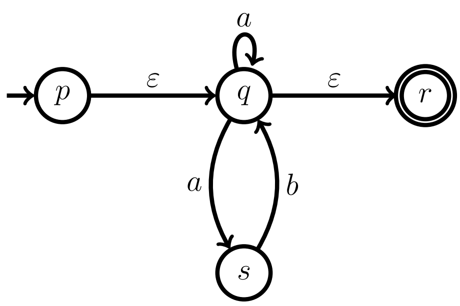
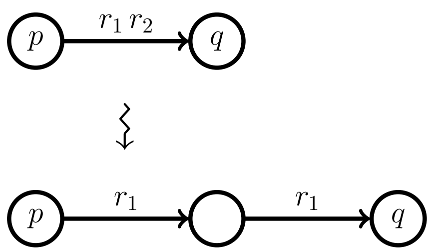
★ Prove or Disprove:

Assume we have an NFA-ε $A = (Q, \Sigma, \delta, Q_0, F)$. Then translating by removing ε-transitions and adding new transition into an NFA B adds in the worst-case $\mathcal{O}(n^2)$ transitions if the original automaton had n transitions. Formally the resulting NFA B should have the definition $B = (Q, \Sigma, (\delta \setminus \{(q, \varepsilon, p) \mid p, q \in Q\}) \cup \delta', Q_0, F')$ and the translation is only allowed to define δ' and F'.

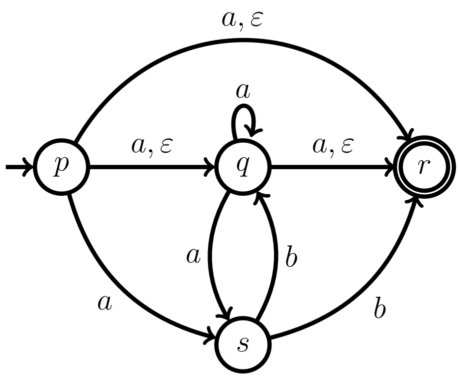
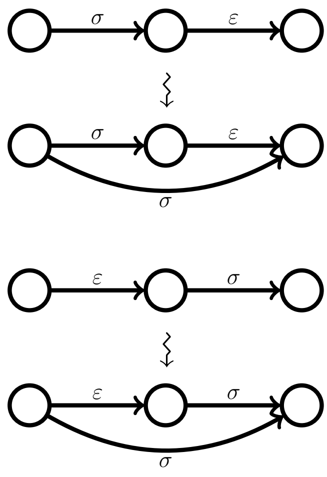
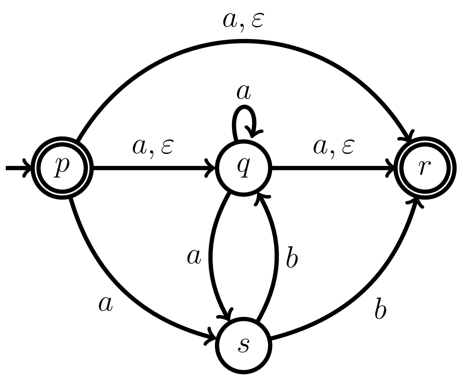
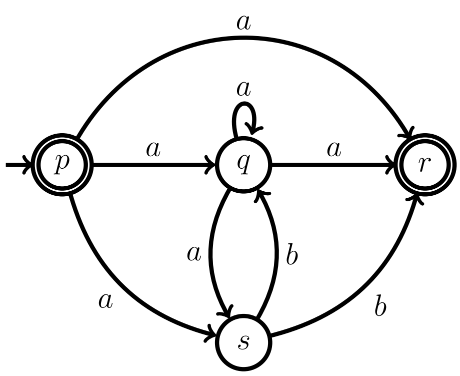
Consider the family of languages defined by the family of regular expressions $r_n = (a_1 + \epsilon)(a_2 + \epsilon) \ldots (a_2 + \epsilon)$ over the alphabet $\Sigma = \{a_1, a_2, \ldots a_n\}$.

Solution 2.1

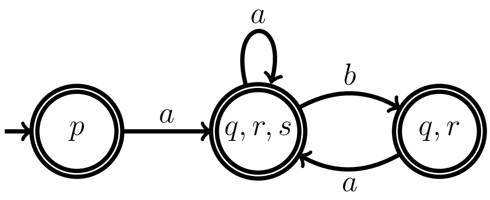
(a)

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

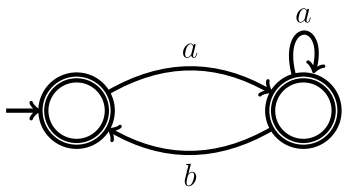
(b)

Iter.	**Automaton obtained**	**Rule applied**
1		 where $\sigma \in \Sigma \cup \{\varepsilon\}$
2		Initial states that can reach a final state through ε-transitions are made final.
3		Remove ε-transitions. Remove states non reachable from initial state.

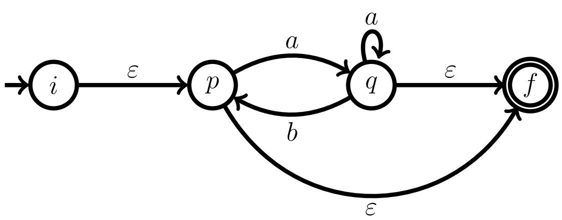
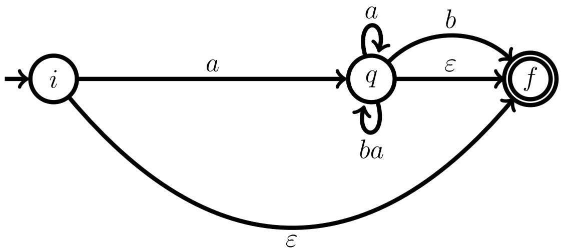
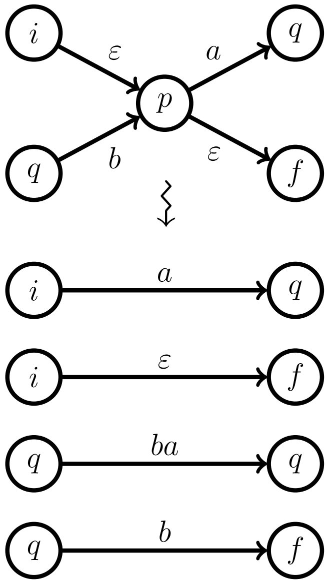
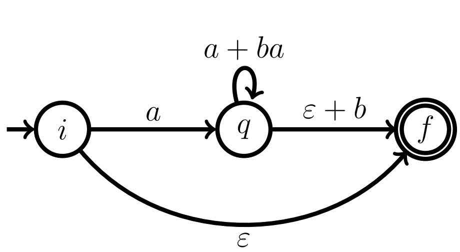
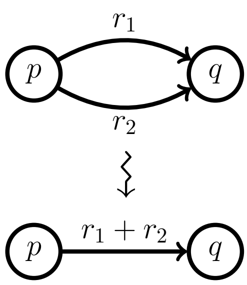
(c)

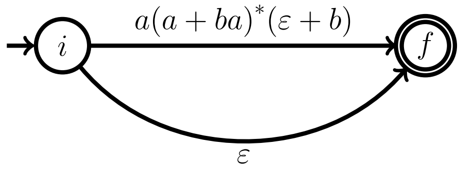
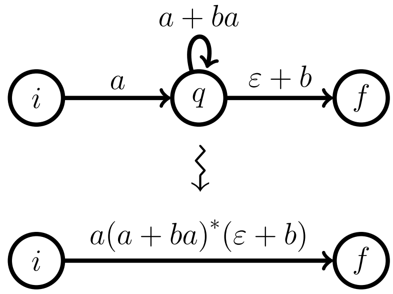
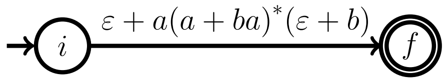
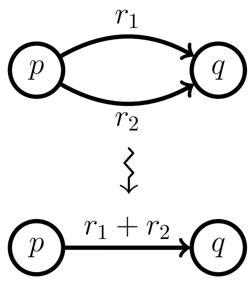


(d) States $\{p\}$ and $\{q,r\}$ have the exact same behaviours, so we can merge them. Indeed, both states are final and $\delta(\{p\},\sigma) = \delta(\{q,r\}),\sigma)$ for every $\sigma \in \{a,b\}$. We obtain:



(e)

<table>
<tr><th>Iter.</th><th>Automaton obtained</th><th>Rule applied</th></tr>
<tr><td>1</td><td>
</td><td>Add single initial and final states.</td></tr>
<tr><td>2</td><td>
</td><td>
</td></tr>
<tr><td>3</td><td>
</td><td>
</td></tr>
</table>

4		
5		
6	$\varepsilon + a(a+ba)^*(\varepsilon+b)$	Extract regular expression from the unique transition.

(f) Let us first show that $a(a+ba)^i = (a+ab)^i a$ 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

$$\begin{aligned}
a(a+ba)^i &= a(a+ba)^{i-1}(a+ba) \\
&= (a+ab)^{i-1}a(a+ba) && \text{(by induction hypothesis)} \\
&= (a+ab)^{i-1}(aa+aba) && \text{(by distributivity)} \\
&= (a+ab)^{i-1}(a+ab)a && \text{(by distributivity)} \\
&= (a+ab)^i a.
\end{aligned}$$

This implies that

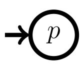
$$a(a+ba)^* = (a+ab)^*a. \tag{1}$$

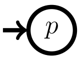
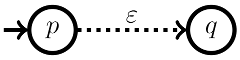
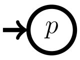
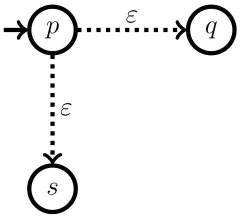
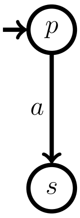
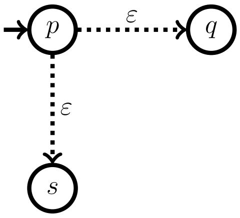
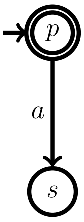
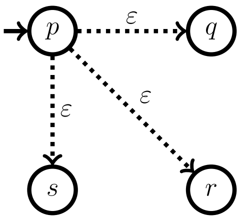
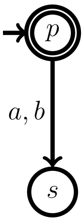
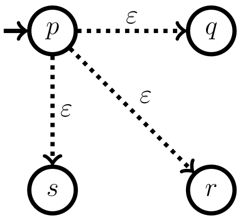
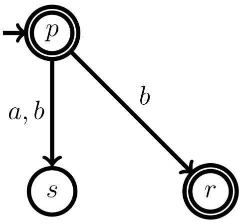
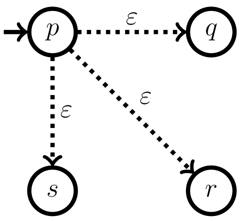
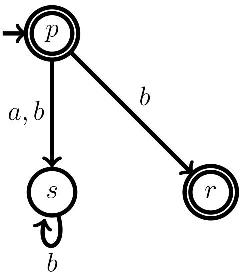
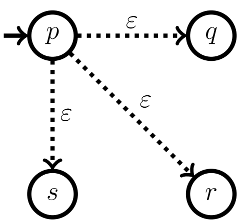
We may now prove the equivalence of the two regular expressions:

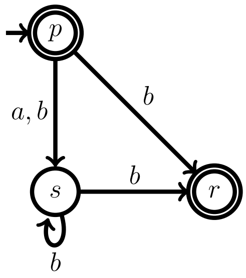
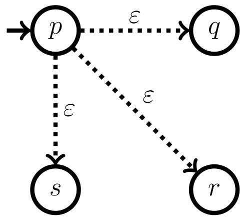
$$\begin{aligned}
\varepsilon + a(a+ba)^*(\varepsilon+b) &= \varepsilon + (a+ab)^*a(\varepsilon+b) && \text{(by (1))} \\
&= \varepsilon + (a+ab)^*(a+ab) && \text{(by distributivity)} \\
&= \varepsilon + (a+ab)^+ \\
&= (a+ab)^*.
\end{aligned}$$

□

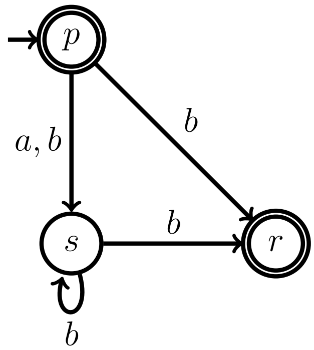
Solution 2.2

Iter.	$B = (Q', \Sigma, \delta', Q'_0, F')$	δ'' **(ε-transitions)**	**Workset** W **and next** (q_1, α, q_2)
0			$\{(p, \varepsilon, q), (p, \varepsilon, s), (p, a, s)\}$

1			$\{(p,\varepsilon,s),(p,a,s),(p,\varepsilon,r)\}$
2			$\{(p,a,s),(p,\varepsilon,r),(p,b,s),(p,b,r)\}$
3			$\{(p,\varepsilon,r),(p,b,s),(p,b,r),(s,b,s),(s,b,r)\}$
4			$\{(p,b,s),(p,b,r),(s,b,s),(s,b,r)\}$
5			$\{(p,b,r),(s,b,s),(s,b,r)\}$
6			$\{(s,b,s),(s,b,r)\}$
7			$\{(s,b,r)\}$

8			$\emptyset$

The resulting NFA is:

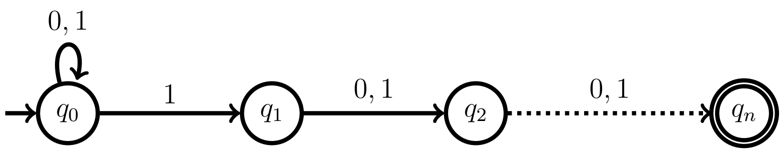


which corresponds to the output of `nfa-eps2nfa`:

```
Q'  = {'p', 'r', 's'}
S   = {'a', 'b'}
d'  = {('p', 'a', 's'), ('s', 'b', 's'), ('p', 'b', 's'), ('s', 'b', 'r'), ('p', 'b', 'r')}
Q0' = {'p'}
F'  = {'p', 'r'}
```

Solution 2.3

(a)



(b) We build a DFA that remembers the last n letters and accepts if the n to last last letter is a 1. More formally, let $A_n = (Q, \Sigma, \delta, q_0, F)$ be such that

$$\begin{aligned} Q &= \{q_u : u \in \{0,1\}^*, |u| \leq n\}, \\ \Sigma &= \{0,1\}, \\ q_0 &= q_\varepsilon, \\ F &= \{q_{1u} : u \in \{0,1\}^*, |u| = n-1\}, \end{aligned}$$

and such that

$$\delta(q_u, a) = \begin{cases} q_{ua} & \text{if } |u| < n, \\ q_{va} & \text{if } u = bv \text{ for some } b \in \{0,1\} \text{ and } v \in \{0,1\}^{n-1}. \end{cases}$$

Note that A_n has $\sum_{i=0}^{n} 2^i = 2^{n+1} - 1$ states.

(c) Let $n \in \mathbb{N}$. For the sake of contradiction, assume there exists a DFA $B = (Q, \{0,1\}, \delta, q_0, F)$ such that $L(B) = L_n$ and $|Q| < 2^n$. By the pigeonhole principle, there exist $u, v \in \{0,1\}^n$ and $q \in Q$ such that $u \neq v$ and

$$q_0 \xrightarrow{u} q \text{ and } q_0 \xrightarrow{v} q. \tag{2}$$

Since $u \neq v$, there exists $1 \leq i \leq n$ such that $u_i \neq v_i$. Without loss of generality, we may assume that $u_i = 1$ and $v_i = 0$. We have $u \cdot 0^{i-1} \in L_n$ and $v \cdot 0^{i-1} \notin L$. This is a contradiction since, by (2), $u \cdot 0^{i-1}$ and $v \cdot 0^{i-1}$ lead to the same state from q_0. □

Solution 2.4

1. We use that $v \in L_n$ iff for every $1 \leq i \leq n$ the i-th and $i+n$-th letters of v coincide. This is a conjunction of conditions. We construct a universal automaton that has a run on v for each of these conditions, and the run accepts iff the condition holds.

The automaton has a spine of states $q_0, q_1, \ldots, q_n$, with transitions $q_i 0, 1 q_{i+1}$ for every $0 \leq i \leq n-1$. At every state q_i the automaton can leave the spine remembering the $(i+1)$-th letter by means of transitions $q_i 0 \mathbf{0}_1$ and $q_i 1 \mathbf{1}_1$. The automaton then reads the next $n-1$ letters by transitions $\mathbf{0}_i 0, 1 \mathbf{0}_{i+1}$ and $\mathbf{1}_i 0, 1 \mathbf{1}_{i+1}$ for every $1 \leq i \leq n-1$, and checks whether the $(i+n)$-th letter matches the $(i+1)$-th letter by transitions $\mathbf{0}_n 0 q_f$ and $\mathbf{1}_n 1 q_f$, where q_f is the unique final state.

2. We use the same technique as in Exercise **??**. Let A be an NFA recognizing L_n. Then, for every $ww \in^{2n}$, the automaton A has at least one accepting run on ww. Let q_w be the state reached by this run (if there are several accepting runs pick anyone). We claim that for any two different words w, w' of length n the states $q_w, q_{w'}$ are also different. Assume $q_w = q_{w'}$. Then, A has an accepting run on ww', obtained by concatenating the first half of the accepting run on ww and the second half of the accepting run on ww'. But $ww' \notin L_n$, contradicting the assumption that A recognizes L_n, and the claim is proved. So A has a different state q_w for each word w of length n, and so it has at least 2^n states.

3. It suffices to replace line 6 of *NFAtoDFA* by : **if** $Q' \subseteq F$ **then add** Q' to $\mathcal{F}$.