Automata and Formal Languages — Homework 5

Due 20.11.2018

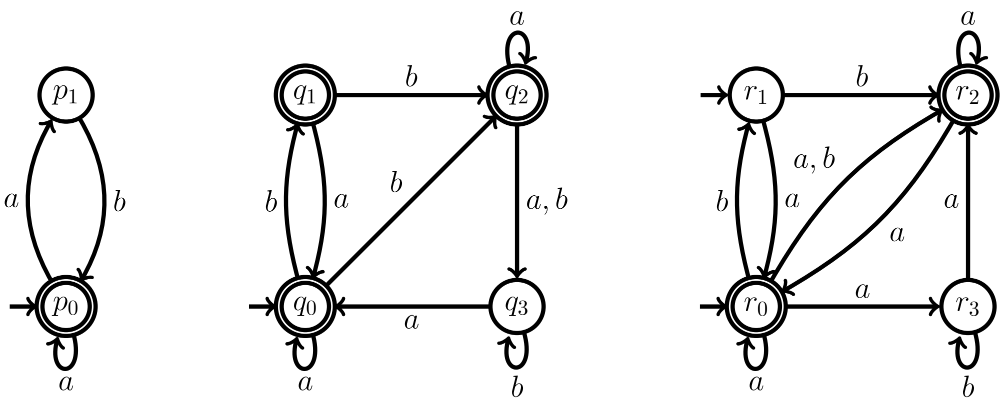
Exercise 5.1
For every $n \in \mathbb{N}$, let $L_n \subseteq \{a,b\}^*$ be the language described by the regular expression $(a+b)^*a(a+b)^n b(a+b)^*$.

(a) Give an NFA A_n with $n+3$ states that accepts L_n.

(b) Decide *algorithmically* whether $baabba \in L(A_2)$ and $baabaa \in L(A_2)$.

(c) If you make final and non final states of A_n respectively non final and final, do you obtain an NFA that accepts $\overline{L_n}$? Justify your answer.

Exercise 5.2
Consider the following NFAs A, B and C:



(a) Use algorithm *UnivNFA* to determine whether $L(B) = \{a,b\}^*$ and $L(C) = \{a,b\}^*$.

(b) Use algorithm *InclNFA* to determine whether $L(A) \subseteq L(B)$ and $L(A) \subseteq L(C)$.

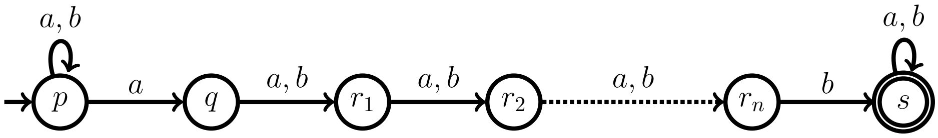
Exercise 5.3
(a) We have seen that testing whether two NFAs accept the same language can be done by using algorithm *InclNFA* twice. Give an alternative algorithm, based on pairings, for testing equality.

(b) Give two NFAs A and B for which exploring only the minimal states of $[\mathit{NFAtoDFA}(A), \mathit{NFAtoDFA}(B)]$ is not sufficient to determine whether $L(A) = L(B)$.

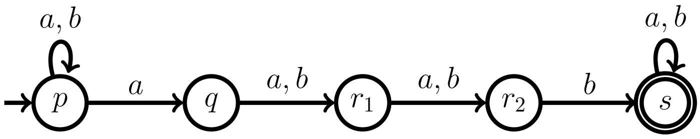
(c) Show that the problem of determining whether an NFA and a DFA accept the same language is PSPACE-hard.

Solution 5.1

(a)



(b) The automaton A_2 is as follows:



Automaton A_2 accepts $w = baabba$ since reading w in the DFA obtained from A_2 yields:

$$\{p\} \xrightarrow{b} \{p\} \xrightarrow{a} \{p, q\} \xrightarrow{a} \{p, q, r_1\} \xrightarrow{b} \{p, r_1, r_2\} \xrightarrow{b} \{p, r_2, s\} \xrightarrow{a} \{p, q, s\}$$

where s is final. However, A_2 rejects $w' = baabaa$ since reading w' in the DFA obtained from A_2 yields:

$$\{p\} \xrightarrow{b} \{p\} \xrightarrow{a} \{p, q\} \xrightarrow{a} \{p, q, r_1\} \xrightarrow{b} \{p, r_1, r_2\} \xrightarrow{a} \{p, q, r_2\} \xrightarrow{a} \{p, q, r_1\}$$

where none of p, q and r_1 are final.

(c) No, it would accept $\{a, b\}^*$ since every word could be accepted in state p.

Solution 5.2

(a) The trace of the execution is as follows:

Iter.	$\mathcal{Q}$	$\mathcal{W}$
0	$\emptyset$	$\{\{q_0\}\}$
1	$\{\{q_0\}\}$	$\{\{q_1, q_2\}\}$
2	$\{\{q_0\}, \{q_1, q_2\}\}$	$\{\{q_2, q_3\}\}$
3	$\{\{q_0\}, \{q_1, q_2\}, \{q_2, q_3\}\}$	$\emptyset$

At the third iteration, the algorithm encounters state $\{q_3\}$ which is non final, and hence it returns *false*. Therefore, $L(B) \neq \{a, b\}^*$.

(b) The trace of the algorithm is as follows:

Iter.	$\mathcal{Q}$	$\mathcal{W}$
0	$\emptyset$	$\{[p_0, \{q_0\}]\}$
1	$\{[p_0, \{q_0\}]\}$	$\{[p_1, \{q_0\}]\}$
2	$\{[p_0, \{q_0\}], [p_1, \{q_0\}]\}$	$\{[p_0, \{q_1, q_2\}]\}$
3	$\{[p_0, \{q_0\}], [p_1, \{q_0\}], [p_0, \{q_1, q_2\}]\}$	$\emptyset$

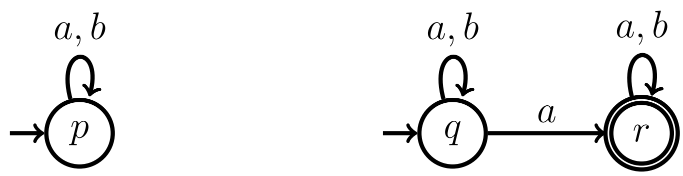
At the third iteration, $\mathcal{W}$ becomes empty and hence the algorithm returns *true*. Therefore $L(A) \subseteq L(B)$.

```
Input: NFAs A = (Q, Σ, δ, Q_0, F) and A' = (Q', Σ, δ', Q'_0, F').
Output: L(A) = L(A')?
Q ← ∅
W ← {[Q_0, Q'_0]}
while W ≠ ∅ do
    pick [P, P'] from W
    if (P ∩ F = ∅) ≠ (P' ∩ F' = ∅) then
        return false
    for a ∈ Σ do
        q ← [δ(P, a), δ'(P', a)]
        if q ∉ Q ∧ q ∉ W then
            add q to W
return true
```

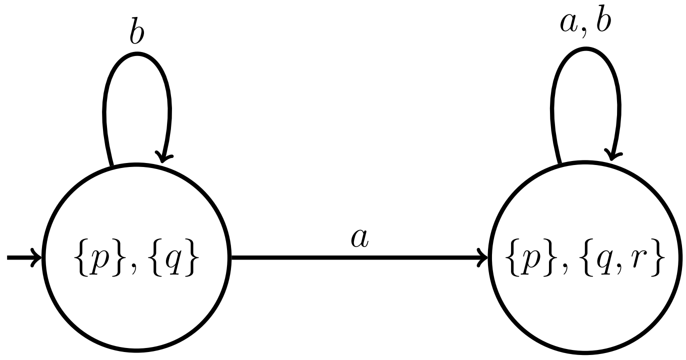
Solution 5.3

(a) We construct the pairing $[\mathit{NFAtoDFA}(A), \mathit{NFAtoDFA}(B)]$ on the fly. The algorithm returns *false* if it encounters a state $[P, P']$ such that only one of P and P' contains a final state. If no such state is encountered, the algorithm returns *true*.

(b) Let A and B be the following NFAs:



The pairing of A and B is as follows:



State $[\{p\}, \{q\}]$ does not allow us to conclude anything since both p and q are non final. However, state $[\{p\}, \{q, r\}]$, which is not minimal, allows us to conclude that $L(A) \neq L(B)$ since r is final.

(c) To show PSPACE-hardness, it suffices to give a reduction from NFA universality. Let A be an NFA. Let B the one state DFA that accepts Σ^*. The following holds:

$$L(A) = \Sigma^* \iff L(A) = L(B).$$

Therefore, $\langle A \rangle \mapsto \langle A, B \rangle$ is a reduction from NFA universality to NFA/DFA equality.