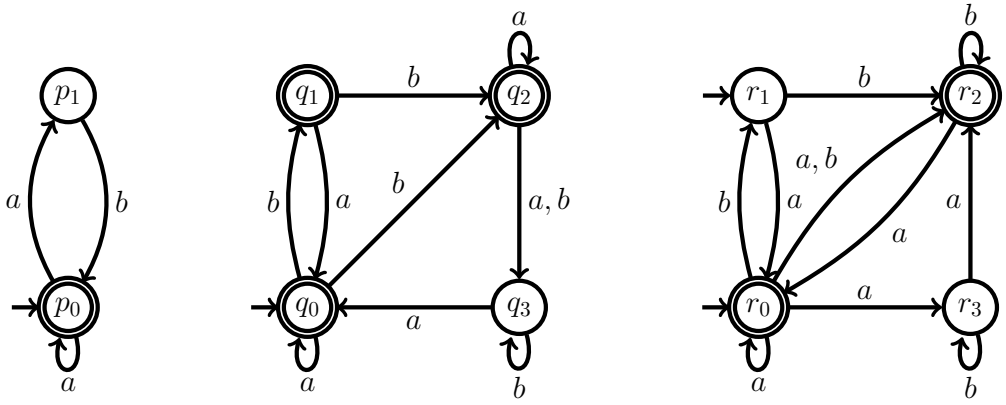
Automata and Formal Languages — Exercise Sheet 6

Exercise 6.1
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) For $D \in \{B, C\}$, if $L(D) \neq \{a, b\}^*$, use algorithm *InclNFA* to determine whether $L(A) \subseteq L(D)$.

Exercise 6.2

1. Build the automata B_p and C_p for the word pattern $p = aamaamm$.
2. How many transitions are taken when reading $t = aamm$ in B_p and C_p?
3. Let $n > 0$. Find a text $t \in \{a, b\}^*$ and a word pattern $p \in \{a, b\}^n$ such that testing whether p occurs in t takes n transitions in B_p and $2n - 1$ transitions in C_p.

Exercise 6.3
In order to make pattern-matching robust to typos we want to include also "similar" words in our results. For this we consider words with a small Levenshtein-distance (edit-distance) "similar".

We transform a word w to a new word w' using the following operations (with $a_i, b \in \Sigma$):

- *replace* (R): $a_1 \dots a_{i-1} a_i a_{i+1} \dots a_l \to a_1 \dots a_{i-1} b a_{i+1} \dots a_l$
- *delete* (D): $a_1 \dots a_{i-1} a_i a_{i+1} \dots a_l \to a_1 \dots a_{i-1} \varepsilon a_{i+1} \dots a_l$
- *insert* (I): $a_1 \dots a_{i-1} a_i a_{i+1} \dots a_l \to a_1 \dots a_{i-1} a_i b a_{i+1} \dots a_l$

The Levenshtein-distance (denoted by $\Delta(w, w')$) of w and w' is the minimal number of operations (R,D,I) needed to transform w into w'. We let $\Delta_{L,i} = \{w \in \Sigma^* \mid \exists w' \in L.\, \Delta(w', w) \leq i\}$, i.e., $\Delta_{L,i}$ is the language of all words with edit-distance at most i to some word of L.

(a) Compute $\Delta(tyrion, tywin)$ and $\Delta(buzz, zurg)$.

(b) Prove the following statement: If L is a regular language, then $\Delta_{L,n}$ is a regular language.

(c) Let p be the pattern $ROCKY$. Construct an NFA-ϵ locating the pattern or variations of it with edit-distance 1.

Exercise 6.4

Let Σ be an alphabet, and define the *shuffle operator* $\|\ :\ \Sigma^* \times \Sigma^* \to 2^{\Sigma^*}$ as follows, where $a, b \in \Sigma$ and $u, v \in \Sigma^*$:

$$\begin{array}{lcl} u \parallel \varepsilon & = & \{u\} \\ \varepsilon \parallel u & = & \{u\} \\ au \parallel bv & = & \{a\}\,(u \parallel bv) \cup \{b\}\,(au \parallel v) \end{array}$$

For example we have:

$$b||d = \{bd, db\}, \quad ab||d = \{abd, adb, dab\}, \quad ab||cd = \{cabd, acbd, abcd, cadb, acdb, cdab\}.$$

Given DFAs recognizing languages $L_1, L_2 \subseteq \Sigma^*$, construct an NFA recognizing their *shuffle*

$$L_1 \parallel L_2 := \bigcup_{u \in L_1, v \in L_2} u \parallel v\ .$$

Solution 6.1

(a) The trace of the execution for NFA B 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	$\{\{q_0\}, \{q_1, q_2\}, \{q_2, q_3\}\}$	$\{\{q_3\}\}$

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

The trace of the execution for NFA C is as follows:

Iter.	$\mathcal{Q}$	$\mathcal{W}$
0	$\emptyset$	$\{\{r_0, r_1\}\}$
1	$\{\{r_0, r_1\}\}$	$\{\{r_0, r_2, r_3\}, \{r_1, r_2\}\}$
2	$\{\{r_0, r_1\}, \{r_0, r_2, r_3\}\}$	$\{\{r_1, r_2\}\}$
3	$\{\{r_0, r_1\}, \{r_0, r_2, r_3\}, \{r_1, r_2\}\}$	$\{\{r_0\}, \{r_2\}\}$
3	$\{\{r_0, r_1\}, \{r_0, r_2, r_3\}, \{r_1, r_2\}, \{r_0\}\}$	$\{\{r_2\}\}$
3	$\{\{r_0, r_1\}, \{r_0, r_2, r_3\}, \{r_1, r_2\}, \{r_0\}, \{r_2\}\}$	$\emptyset$

At the fifth iteration, $\mathcal{W}$ becomes empty and hence the algorithm returns *true*. Therefore $L(C) = \{a, b\}^*$.

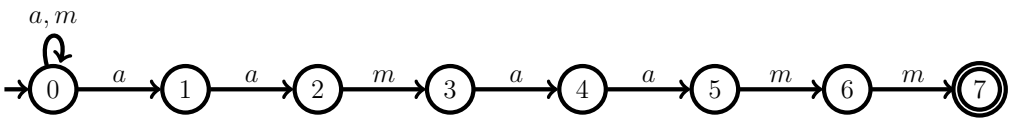
(b) The trace of the algorithm for A and B is as follows:

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	$\{[p_0, \{q_0\}], [p_1, \{q_0\}], [p_0, \{q_1, q_2\}]\}$	$\emptyset$

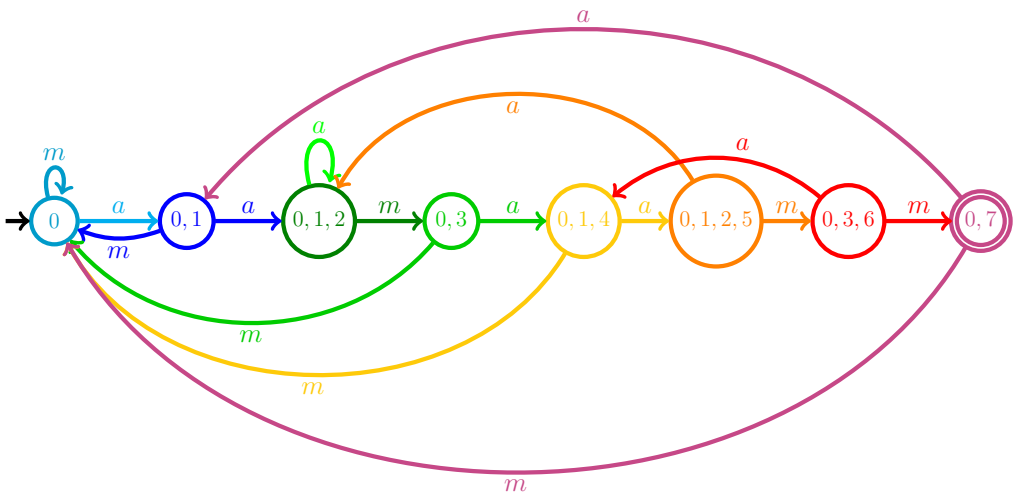
At the third iteration, $\mathcal{W}$ becomes empty and hence the algorithm returns *true*. Therefore $L(A) \subseteq L(B)$.

Solution 6.2

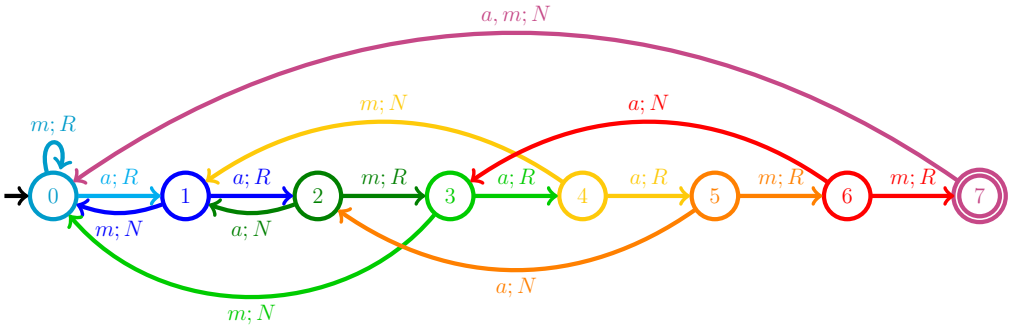
1. A_p :



B_p :

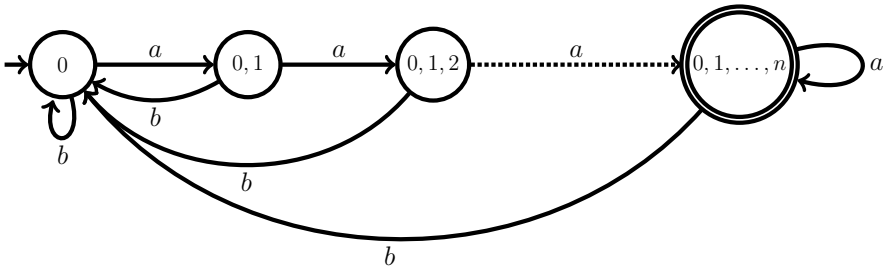


C_p :

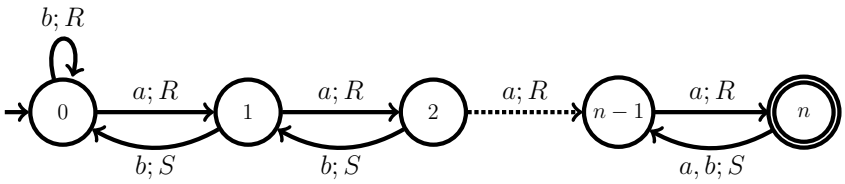


2. Four transitions taken in B_p: $\{0\} \xrightarrow{a} \{0,1\} \xrightarrow{a} \{0,1,2\} \xrightarrow{m} \{0,3\} \xrightarrow{m} \{0\}$.
   Five transitions taken in C_p: $0 \xrightarrow{a} 1 \xrightarrow{a} 2 \xrightarrow{m} 3 \xrightarrow{m} 0 \xrightarrow{m} 0$.
3. $t = a^{n-1}b$ and $p = a^n$. The automata B_p and C_p are as follows:

B_p:



C_p:



The runs over t on B_p and C_p are respectively:

$$\{0\} \xrightarrow{a} \{0,1\} \xrightarrow{a} \{0,1,2\} \xrightarrow{a} \cdots \xrightarrow{a} \{0,1,\ldots,n-1\} \xrightarrow{b} \{0\} \ ,$$

and

$$0 \xrightarrow{a} 1 \xrightarrow{a} 2 \xrightarrow{a} \cdots \xrightarrow{a} (n-1) \xrightarrow{b} (n-2) \xrightarrow{b} (n-3) \xrightarrow{b} \cdots \xrightarrow{b} 0 \ .$$

Solution 6.3

(a) $\Delta(tyrion, tywin) = 2$, because the r can be replaced by a w and the o can be removed. $\Delta(buzz, zurg) = 3$, because the u can be fixed and the others can be replaced.

(b) Let $M = (Q, \Sigma, \delta, q_0, F)$ be a DFA for L. We obtain an NFA-ϵ N for $\Delta_{L,n}$ by adding n "error-levels". Formally:

$$N = (Q \times [0,n], \Sigma, \delta', (q_0, 0), F \times [0,n])$$

with

$$\begin{aligned}
\delta' \ &= \{((q,i), a, (p,i)) \mid \delta(q,a) = p \text{ and } i \le n\} && \text{no change} \\
&\cup \{((q,i), \varepsilon, (p,i+1)) \mid \exists a \in \Sigma.\, \delta(q,a) = p \text{ and } i < n\} && \text{delete} \\
&\cup \{((q,i), a, (q,i+1)) \mid q \in \ \text{ and } i < n\} && \text{insert} \\
&\cup \{((q,i), b, (p,i+1)) \mid \exists a \in \Sigma \setminus \{b\}.\, \delta(q,a) = p \text{ and } i < n\} && \text{replace}
\end{aligned}$$

Let us prove that $\Delta_{L,n} = L(N)$.

$\Delta_{L,n} \subseteq L(N)$**.** If $w \in \Delta_{L,n}$, it means that there is $w' \in L$ such that $\Delta(w', w) = k \le n$, or in other words, starting from the word w', we can obtain w by applying k "mistakes" (delete, insert, replace). As $w' \in L$ (accepted by M) and as the 0-level of N is a copy of M, note that w' has a run in N that reaches a final state $(q_f, 0)$. By construction of the automaton N, there is a run of the word w that follows the run of w' where each "mistake" can be seen as moving to the next error-level, using the corresponding transition from δ' (delete, insert, replace) depending on a mistake. It is easy to see that if the word w' reaches a final state $(q_f, 0)$ in N, then w can reach (q_f, k), and thus $w \in L(N)$.

$L(N) \subseteq \Delta_{L,n}$**.** If $w \in L(N)$, this means there is a run of w in N that reaches a final state $(q_f, k) \in F \times [0,n]$. Intuitively, for each transition of that run that changes the level, we modify w so that it "stays in the same level". Formally, we check the nature of the transition that changes the level and modify w as follows:

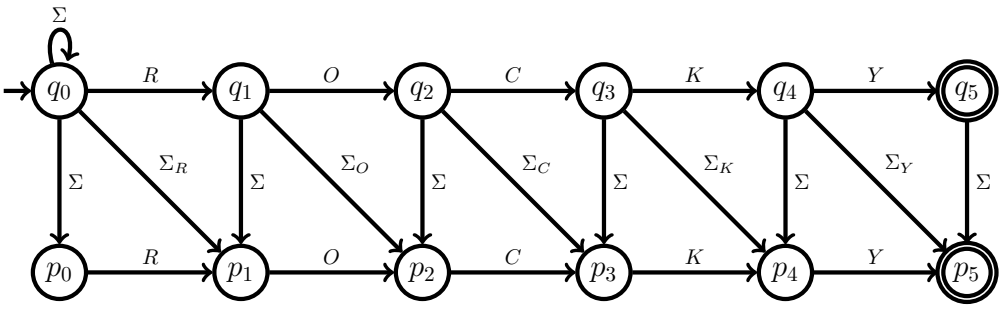
(i) If $(p,i) \xrightarrow{a} (p,i+1)$ is an insert edge, this occurrence of the letter a will be removed from w.

(ii) If $(p,i) \xrightarrow{a} (q,i+1)$ is a replace edge, then there exists a $(p,i) \xrightarrow{b} (q,i)$ edge for some letter b, and we replace this occurrence of a in w with b.

(iii) If $(p,i) \xrightarrow{\epsilon} (q,i+1)$ is a delete edge, then there exists a $(p,i) \xrightarrow{a} (q,i)$ edge for some letter a, and we add the letter a at this place in w.

Denote the obtained word by w'. It is easy to see that w' is obtained from w by applying mistakes (delete, insert, replace) k times, as in the run of w there are exactly k transitions that change the level. Therefore, $\Delta(w', w) \le k \le n$. Moreover, it is easy to see that if w reaches (q_f, k), then w' reaches $(q_f, 0)$. As the 0-level is a copy of M, we have $w' \in L$. To summarize, there exists $w' \in L$ such that $\Delta(w', w) \le n$, that is, $w \in \Delta_{L,n}$.

(c) We use the same construction as in (b) with the automaton A_p for pattern $p = ROCKY$. Let $\Sigma = \{R, O, C, K, Y\}$ and $\Sigma_a = \Sigma \setminus \{a\} \cup \{\epsilon\}$ for every $a \in \Sigma$.



Solution 6.4

Let $A_i = (Q_i, \Sigma, \delta_i, q_0^{(i)}, F_i)$ be a DFA with $L_i = L(A_i)$ (for $i = 1, 2$). We use a variation of the pairing construction, i.e., we construct an automaton with states $Q_1 \times Q_2$. While in the standard pairing construction both automata move when a symbol is read, we now choose nondeterministically one of the two automata, which is to move accordingly to the symbol read, while the other one does not change its state, i.e.,

$$\delta((q, q'), a) := \{(\delta_1(q, a), q'), (q, \delta_2(q', a))\}.$$

It is left to the reader to show that this automaton indeed accepts exactly $L_1 \parallel L_2$.