Theoretical Computer Science - Bridging Course  
Summer Term 2016  
Sample Solution for Exercise Sheet 2

Exercise 1: The Empty String (3 points)

The empty string $\varepsilon$ is defined as a string of length $|\varepsilon| = 0$. Shortly explain the difference between $\varepsilon$, $L_1 := \emptyset$ and $L_2 := \{\varepsilon\}$. Construct a corresponding DFA where applicable.

Solution:

We compare the terms pairwise:

1. The empty language $\emptyset$ is a set containing no strings, while the empty string $\varepsilon$ is a string containing no symbols.

2. $L_2 = \{\varepsilon\}$ is a set containing $\varepsilon$, while $\varepsilon$ is just a string.

3. $L_1, L_2$ are different languages since $L_2$ contains a string while $L_1$ is empty ($0 = |L_1| \neq |L_2| = 1$).

Since $\varepsilon$ is not a language, we can give DFAs only for $L_1$ and $L_2$ (see Figure 1).

![Figure 1: DFA for $L_1$ (left) and for $L_2$ (right).]

Exercise 2: Constructing DFAs (1+2+3 points)

Let $\Sigma = \{0, 1\}$ be an alphabet. Construct DFAs that recognize the following languages over $\Sigma$.

(a) $L_1 = \{w \in \Sigma^* \mid w \text{ contains at most one } 1\}$.

(b) $L_2 = \{w \in \Sigma^* \mid w \text{ contains the substring } 00\}$.

(c) $L_1 \cup L_2$.

Remark: $\Sigma^* := \{a_1 \cdots a_n \mid a_i \in \Sigma, n \in \mathbb{N}_0\}$ is the set of all strings over the alphabet $\Sigma$.

Solution:

See Figure 2. For $L_1 \cup L_2$ we constructed the product automaton according to the procedure from the lecture slides (page 31) which shows that the union of regular languages is again regular. This automaton is not minimal, i.e. there exists an automaton that accepts the same language and has fewer states.
Exercise 3: Closure under Complement and Intersection (2+3 points)

Let $L, L_1, L_2$ be regular languages. Show that both $\overline{L} := \Sigma^* \setminus L$ and $L_1 \cap L_2$ are regular as well.

Solution:

Since $L$ is regular there is a DFA $M = (Q, \Sigma, \delta, q_0, F)$ that recognizes $L$. Without loss of generality we require $M$ to be complete, meaning that for any input it always halts in a state $q \in Q$. From $M$ we construct a DFA $\overline{M} := (Q, \Sigma, \delta, q_0, \overline{F})$ that recognizes $\overline{L}$ by inverting the set of accepting states of $M$, i.e. $\overline{F} := Q \setminus F$.

If $w \in L$, then $w$ is accepted by $M$, i.e. $M$ halts in an accepting state $q \in F$. Since $\overline{M}$ halts in the same state $q \notin \overline{F}$ when processing $w$, $\overline{M}$ rejects $w$. Analogously $\overline{M}$ accepts any $w \notin L$ that are rejected by $M$. We conclude that $\overline{M}$ recognizes the language $\overline{L}$ which is therefore regular.

With the law of DeMorgan we obtain: $L_1 \cap L_2 = \neg(\neg L_1 \cup \neg L_2)$. Thus $L_1 \cap L_2$ is regular, since we already know that regularity is conserved by complementation and a finite number of unions of regular languages (cf. lecture).

Remark: Alternatively we can construct the product automaton for $L_1 \cap L_2$ analogously to the procedure in the lecture (for $L_1 \cup L_2$), with the one difference that we set $F := F_1 \times F_2$ where $F_1, F_2$ are the sets of accepting states of DFAs for $L_1, L_2$.
Exercise 4: NFA to DFA Conversion (2+4 points)

Consider the NFA given in Figure 3.

(a) Give a formal description of the above NFA, i.e. \((Q, \Sigma, \delta, q_0, F)\). Write down the transition function \(\delta\) explicitly (e.g. in the form \(\delta(q, x) = A\) for any transition from \(q \in Q\) to \(A \subseteq Q\) with the symbol \(x \in \Sigma \cup \{\epsilon\}\)).

(b) Construct a DFA which is equivalent to the above NFA.

Solution:

(a) We denote the automaton given in Figure 3 by \(M := (Q, \Sigma, \delta, q_1, F)\). From Figure 3 we conclude that \(Q := \{q_1, q_2, q_3\}\), \(\Sigma := \{a, b\}\) (Note that \(\epsilon \notin \Sigma\)) and \(F := \{q_2\}\). We formalize the transition function \(\delta : \Sigma \cup \{\epsilon\} \rightarrow \mathcal{P}(Q)\) as follows

\[
\begin{align*}
q_1: & \quad \delta(q_1, a) = \{q_3\}, \delta(q_1, b) = \emptyset, \delta(q_1, \epsilon) = \{q_2\} \\
q_2: & \quad \delta(q_2, a) = \{q_1\}, \delta(q_2, b) = \emptyset, \delta(q_2, \epsilon) = \emptyset \\
q_3: & \quad \delta(q_3, a) = \{q_2\}, \delta(q_3, b) = \{q_2, q_3\}, \delta(q_3, \epsilon) = \emptyset.
\end{align*}
\]

(b) In Figure 4 we constructed the power set automaton from the automaton given in Figure 3 according to the proof from the lecture slides (page 48) showing that for any NFA there exists a DFA which accepts the same language. We left out states which are irrelevant for the functioning of the automaton.

Figure 3: NFA for Exercise 4.

Figure 4: DFA for Exercise 4.