Exercise 1: Drawing DFAs and NFAs (8 Points)

Consider the following three languages over the alphabet \{0, 1\}.

\[ L_1 = \{ w \mid |w| \geq 2 \text{ and } w \text{ contains an even number of zeros} \} \]

\[ L_2 = \{ w \mid w \text{ contains exactly two ones} \} \]

\[ L_3 = \{ w \mid w \text{ has an odd number of zeros and ends with 1} \} \]

First draw a DFA for each of the languages \( L_1, L_2 \) and \( L_3 \). Then, for each of the following languages, provide an NFA that recognizes the given language.

(a) \( L_1^* \)

(b) \( L_3 \circ L_2 \)

(c) \( L_2 \cup L_3 \)

Sample Solution

Here are the DFAs for the three languages:

(a) \( L_1 \):

\[ \text{start} \rightarrow q_0 \rightarrow q_1 \rightarrow q_2 \rightarrow q_3 \rightarrow q_4 \rightarrow q_5 \rightarrow q_6 \]

[Diagram of DFA for \( L_1 \)]

(b) \( L_3 \circ L_2 \):

[Diagram of NFA recognizing \( L_3 \circ L_2 \)]

(c) \( L_2 \cup L_3 \):

[Diagram of NFA recognizing \( L_2 \cup L_3 \)]
For constructing the NFAs regarding the given three languages in (a), (b), and (c), it is enough to reuse the drawn DFAs and insert proper epsilon transitions. Let $N_1$ and $N_2$ denote two DFAs. Then the following figures show how to utilize the DFAs to construct $L(N_1) \cup L(N_2)$, $L(N_1) \circ L(N_2)$, and $L(N_1)^*$ respectively. The figures are taken from the lecture slides.

Figure 1: $L(N_1) \cup L(N_2)$

Figure 2: $L(N_1) \circ L(N_2)$
Exercise 2: Regular Languages

Let \( L, L_1, L_2 \) be regular languages. Show that both \( L := \Sigma^* \setminus L \) and \( L_1 \cap L_2 \) are regular as well by constructing the corresponding DFAs.

**Remark:** No need for drawing state diagrams. Show how a DFA for the language in question can be constructed presuming the existence of DFAs for \( L, L_1, L_2 \).

**Sample Solution**

Let \( M = (Q, \Sigma, \delta, q_0, F) \) be the DFA recognizing \( L \). We define the DFA \( \overline{M} := (Q, \Sigma, \delta, q_0, \overline{F}) \) by inverting the set of accepting states of \( M \), i.e. \( \overline{F} := Q \setminus F \). We show that \( \overline{M} \) recognizes \( \overline{L} \).

If \( w \in L \), then \( w \notin \overline{L} \) and \( M \) halts in an non accepting state \( q \) when processing \( w \). \( \overline{M} \) will halt in the same state (because we only changed the set of accepting states), but here \( q \) is an accepting state. Analogously, if \( w \notin L \), then \( w \in \overline{L} \) and so \( M \) halts in an accepting state when processing \( w \). \( \overline{M} \) will again halt in the same state, but here \( q \) is a non accepting state. So we have that \( \overline{M} \) halts in an accepting state when processing \( w \) if and only if \( w \in \overline{L} \). Thus \( \overline{M} \) recognizes the language \( \overline{L} \) which is therefore regular.

For proving the regularity of \( L_1 \cap L_2 \), we construct the product automaton like done in the lecture (Theorem 1.25. p. 30) for \( L_1 \cup L_2 \), with the difference that we set \( F := F_1 \times F_2 \) as the set of accepting states, where \( F_1 \) and \( F_2 \) are the sets of accepting states of the DFAs for \( L_1 \) and \( L_2 \).
Exercise 3: NFA to DFA

Consider the following NFA.

(a) Give a formal description of the NFA by giving the alphabet, state set, transition function, start state and the set of accept states.

(b) Construct a DFA which is equivalent to the above NFA by drawing the corresponding state diagram.

(c) Explain what language the automaton recognizes.

Sample Solution

(a) The set of states is $Q = \{q_0, q_1, q_2\}$; the alphabet $\Sigma = \{0, 1\}$; the initial state is $q_0$; the set of accept states is $F = \{q_1\}$; the transition function is shown in the following table.

<table>
<thead>
<tr>
<th></th>
<th>$q_0$</th>
<th>$q_1$</th>
<th>$q_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>$q_0$</td>
<td>$\emptyset$</td>
<td>$\emptyset$</td>
</tr>
<tr>
<td>1</td>
<td>$q_2$</td>
<td>$\emptyset$</td>
<td>$q_1, q_2$</td>
</tr>
<tr>
<td>$\epsilon$</td>
<td>$q_1$</td>
<td>$\emptyset$</td>
<td>$\emptyset$</td>
</tr>
</tbody>
</table>

(b)
If we leave out nodes with no path leading into it, we have

```
0, 1

∅ -> {q2}

start -> {q0, q1} -> {q1, q2}
```

0
0
1
1