Automata and Formal Languages — Exercise Sheet 5

Exercise 5.1
Consider the following languages over alphabet $\Sigma = \{a, b\}$:

- $L_1$ is the set of all words where $a$ occurs only at odd positions;
- $L_2$ is the set of all words with an even number of $a$’s;
- $L_3$ is the set of all words where between any two occurrences of $b$’s there is at least one $a$;
- $L_4$ is the set of all words of odd length.

Construct an NFA for the language $(L_1 \setminus L_2) \cup (L_3 \triangle L_4)$, where $L \triangle L'$ denotes the symmetric difference of $L$ and $L'$, i.e. $(L \setminus L') \cup (L' \setminus L)$, while sticking to the following rules:

- Start from DFAs for $L_1, \ldots, L_4$;
- Any further automaton must be constructed from already existing automata via an algorithm introduced in the lecture, e.g. Comp, BinOp, UnionNFA, NFAtoDFA, etc.

Exercise 5.2
Let $\Sigma$ be a finite alphabet. For every $u, v \in \Sigma^*$, we say that $u \preceq v$ if and only if $u$ can be obtained by deleting zero or more letters of $v$. For example, $abc \preceq abca$, $abc \preceq acbac$, $abc \preceq abc$, $\varepsilon \preceq abc$ and $aab \not\preceq acbac$.

Let $L \subseteq \Sigma^*$ be a language accepted by an NFA $A$. Give an NFA-$\varepsilon$ for each of the following languages:

(a) $\downarrow L = \{w \in \Sigma^* \mid w \preceq w' \text{ for some } w' \in L\}$,
(b) $\uparrow L = \{w \in \Sigma^* \mid w' \preceq w \text{ for some } w' \in L\}$,
(c) $\sqrt{L} = \{w \in \Sigma^* \mid ww \in L\}$,
(d) $\star \text{ Cyc}(L) = \{vu \in \Sigma^* \mid uu \in L\}$.

Exercise 5.3
Let $L \neq \{\varepsilon\}$ be an arbitrary non-empty language over a 1-letter alphabet. Prove that there exists words $v_1, v_2, \ldots, v_n, w$ such that $L^* = (v_1 + v_2 + \cdots + v_n)w^*$.

(Hint: Consider the shortest non-empty word $w \in L$. If $L^* = w^*$, then we are done. Otherwise, pick the shortest word $v_1 \in L^* \setminus w^*$. If $L^* = v_1w^*$, then we are done. Otherwise, pick the shortest word $v_2 \in L^* \setminus v_1w^*$ and so on).
**Solution 5.1**

We start from the following deterministic automata:

- $L_1$: \[ a, b \]
- $L_2$: \[ a \]
- $L_3$: \[ a, b \]
- $L_4$: \[ a, b \]

By applying $\text{BinOp}$ (and omitting the trap state on $L_1 \setminus L_2$), we obtain:

- $L_1 \setminus L_2$: \[ a, b, b, a, b \]
- $L_3 \triangle L_4$: \[ a, a, b, a, b, a, b, a, b \]

By using $\text{Comp}$ on the rightmost automaton, we obtain:

- $L_3 \triangle L_4$: \[ a, a, b, a, b, a, b, a, b \]

By considering the NFA for $L_1 \setminus L_2$ and the above NFA as a single automaton, we obtain an NFA for $(L_1 \setminus L_2) \cup (L_3 \triangle L_4)$.

**Solution 5.2**

Let $A = (Q, \Sigma, \delta, Q_0, F)$ be an NFA that accepts $L$.

(a) We add a $\varepsilon$-transition “parallel” to every transition of $A$. This simulates the deletion of letters from words of $L$. More formally, let $B = (Q, \Sigma, \delta', Q_0, F)$ be such that, for every $q \in Q$ and $a \in \Sigma \cup \{\varepsilon\}$,

\[
\delta'(q, a) = \begin{cases} 
\delta(q, a) & \text{if } a \in \Sigma, \\
\{q \in Q : q \in \delta(q, b) \text{ for some } b \in \Sigma\} & \text{if } a = \varepsilon.
\end{cases}
\]

(b) For every state of $Q$, we add self-loops for each letter of $\Sigma$. This corresponds to the insertion of letters in words of $L$. More formally, let $B = (Q, \Sigma, \delta', Q_0, F)$ be such that $\delta'(q, a) = \delta(q, a) \cup \{q\}$ for every $q \in Q$ and $a \in \Sigma$.

(c) Intuitively, we construct an automaton $B$ that guesses an intermediate state $p$ and then reads $w$ simultaneously from an initial state $q_0$ and from $p$. The automaton accepts if it simultaneously reaches $p$ and an accepting state $q_F$. More formally, let $B = (Q', \Sigma, \delta', Q'_0, F')$ be such that

\[
Q' = Q \times Q \times Q, \\
Q'_0 = \{(p, q, p) : p \in Q, q \in Q_0\}, \\
F' = \{(p, p, q) : p \in Q, q \in F\}.
\]
and, for every $p,q,r \in Q$ and $a \in \Sigma$,
\[
\delta'(\langle p,q,r \rangle, a) = \{ (p,q',r') : q' \in \delta(q,a), r' \in \delta(r,a) \}.
\]

(d) Intuitively, we construct an automaton $B$ that guesses a state $p$ and reads a prefix $v$ of the input word until it reaches a final state. Then, $B$ moves non deterministically to an initial state from which it reads the remainder $u$ of the input word, and it accepts if it reaches $p$. More formally, let $B = (Q', \Sigma, \delta', Q'_0, F')$ be such that
\[
\begin{align*}
Q' &= Q \times \{0,1\} \times Q, \\
Q'_0 &= \{ (p,0,p) \mid p \in Q \}, \\
F' &= \{ (p,1,p) \mid p \in Q \},
\end{align*}
\]
and, for every $p,q \in Q$ and $a \in \Sigma \cup \{\varepsilon\}$,
\[
\delta'((p,b,q),a) = \begin{cases} 
\{ (p,b,q') : q' \in \delta(q,a) \} & \text{if } a \in \Sigma, \\
\{ (p,1,q') : q' \in Q_0 \} & \text{if } a = \varepsilon, b = 0 \text{ and } q \in F, \\
\emptyset & \text{otherwise.}
\end{cases}
\]

\textbf{Solution 5.3}

Without loss of generality, we can assume that the alphabet is $\{a\}$. As the hint suggests, we first consider the shortest non-empty word $w \in L$. If $L^* = w^*$, then we are done. Otherwise, there must be a shortest word $v_1 \in L^* \setminus w^*$. If $L^* = v_1 w^*$, then we are done again. Otherwise, there must be a shortest word $v_2 \in L^* \setminus v_1 w^*$ and so on.

We claim that in at most $p = |w|$ steps, this process will terminate and we will find words $v_1, \ldots, v_n, w$ that satisfy the required claim. Indeed, suppose this process does not terminate in at most $p$ steps and so we have constructed words $v_1, v_2, \ldots, v_{p+1}$. By the pigeonhole principle, there exists $1 \leq i < j \leq p + 1$ such that $|v_i| \equiv |v_j| \pmod{p}$. Notice that $|v_i| \neq |v_j|$ as otherwise $v_i = v_j$, because both of them are words over a singleton alphabet. Hence we have two cases.

Suppose $|v_i| < |v_j|$. Since $|v_i| \equiv |v_j| \pmod{p}$, there must be a $k > 0$ such that $|v_j| = |v_i| + k \cdot p$. Hence, $v_j = a^{|v_i|} = a^{|v_j|+k \cdot p} = v_i w^k \in v_i w^*$, contradicting the way $v_j$ was picked.

Suppose $|v_j| < |v_i|$. Since $|v_i| \equiv |v_j| \pmod{p}$, there must be a $k > 0$ such that $|v_i| = |v_j| + k \cdot p$. Hence, $v_i = a^{|v_i|} = a^{|v_j|+k \cdot p} = v_j w^k \in v_j w^*$. If $v_j \in (v_1 + v_2 + \cdots + v_{i-1})w^*$, this would then mean that $v_i \in (v_1 + v_2 + \cdots + v_{i-1})w^*$ as well, contradicting the way $v_i$ was picked. Otherwise, $v_j \notin (v_1 + v_2 + \cdots + v_{i-1})w^*$, but then $|v_j| < |v_i|$, which also contradicts the choice of $v_i$. It follows that in either case, we arrive at a contradiction.

Hence, the process terminates in at most $p$ steps. Since the process terminates, it means that we have found $v_1, \ldots, v_n, w$ satisfying the property that $L^* = (v_1 + v_2 + \cdots + v_n)w^*$.

\textbf{Remark:} A previous version of this question which also required that the words $v_1, \ldots, v_n, w$ belong to $L$, was wrong. This has now been corrected.