Solutions of homework 1

1. Prove that the following languages are regular, either by exhibiting a regular expression representing the language, or a DFA/NFA that recognizes the language:

(a) the set of all words in the Oxford English dictionary, for $\Sigma = \{a, b, \ldots, z\}$
(b) all strings that do not contain the substring aba, for $\Sigma = \{a, b\}$ (for instance, aabaa contains the substring aba, whereas abba does not)
(c) all strings that do not contain 3 consecutive occurrences of the same letter, for $\Sigma = \{a, b\}$

Solution Outline: (5, 10, 10 points each)

(a) There are only finitely many such words. Either write down a regular expression that is the union of each of these words, or a NFA that non-deterministically (via $\epsilon$-transitions) accepts each of these words.
(b) Take $Q = \{q_0, qa, qab, qaba\}$, with $q_0$ the start state, and $F = \{q_0, qa, qab\}$. $\delta(q_0, a) = qa$, $\delta(q_0, b) = qab$, $\delta(qa, a) = qaba$, $\delta(qaba, b) = qaba$, $\delta(qaba, b) = q_0$, $\delta(qa, a) = qa$, $\delta(qab, b) = q_0$.
(c) Take $Q = \{q_0, qa, qaa, qaaa, qb, qbb, qbbb\}$, with $q_0$ the start state, and $F = \{q_0, qa, qaa, qb, qbb\}$. We have $\delta(q_0, a) = qa$, $\delta(q_0, a) = qaa$, $\delta(qaa, a) = qaaa$, and similarly, $\delta(q_0, b) = qb$, $\delta(qb, b) = qbb$, $\delta(qbb, b) = qbbb$. Add a $b$-transition from each of $qa, qaa, qaaa$ to $q_0$, and an $a$-transition from each of $qb, qbb, qbbb$ to $q_0$. Finally, $\delta(qaaa, a) = qaaa$, $\delta(qaaa, b) = qaaa$ and $\delta(qbbb, a) = qbbb$, $\delta(qbbb, b) = qbbb$.

2. (Sipser, problem 1.24) For any strong $w = w_1 w_2 \cdots w_n$, the reverse of $w$, written as $w^R$ is the string $w$ in reverse order, $w_n w_{n-1} \cdots w_2 w_1$. For any language $A$, let $A^R = \{w^R \mid w \in A\}$. Show that if $A$ is regular, so is $A^R$.

Solution Outline: (20 points)

One solution is recursively (or inductively) define a reversing operation on regular expressions, and apply that operation on the regular expression for $A$. In particular, given a regular expression $R$, reverse($R$) is:

- $a$ for some $a \in \Sigma$
- $\epsilon$ if $R = \epsilon$
- $\emptyset$ if $R = \emptyset$
- reverse($R_1 \cup R_2$), if $R = R_1 \cup R_2$
- reverse($R_2 \circ R_1$) if $R = R_1 \circ R_2$, or
- $\text{reverse}(R_1)^*$, if $R = (R_1^*)$.
Another solution is to start with a DFA $M$ for $A$, and build a NFA $M'$ for $A^R$ as follows: reverse all the arrows of $M$, and designate the start state for $M$ as the only accept state $q'_{acc}$ for $M'$. Add a new start state $q'_0$ for $M'$, and from $q'_0$, add $\epsilon$-transitions to each state of $M'$ corresponding to accept states of $M$.

It is easy to verify that for any $w \in \Sigma^*$, there is a path following $w$ from the state start to an accept state in $M$ iff there is a path following $w^R$ from $q'_0$ to $q'_{acc}$ in $M'$. It follows that $w \in A$ iff $w^R \in A^R$.

(7 points for saying reversing the arrows; 3 points for explaining the new accept state, and 5 points for explaining the new start state and the $\epsilon$-transitions. 5 points for explaining, or at least making the final observation about the paths/connectivity.)

3. For any language $A$ with alphabet $\Sigma$, let

$$A^{sub} = \{ w \in \Sigma^* \mid w \text{ is a substring of } x, \text{ for some } x \in A \}$$

Show that if $A$ is regular, so is $A^{sub}$.

**Solution Outline:** (20 points)

Again, we start with a DFA $M$ for $A$, and build a NFA $M'$ for $A^{sub}$. Copy the states and transitions of $M$ to $M'$. The accept states of $M'$ are those corresponding to every state in $M$ that is connected to an accept state in $M$ (via a directed path ending at an accept state). In addition, we add a new start state $q'_0$ for $M'$, and add $\epsilon$-transitions from $q'_0$ to every other state in $M'$ corresponding to a state in $M$ that is reachable from the start state $q_0$ of $M$ (via a directed path starting at $q_0$). Also designate $q'_0$ as an accept state.

It is easy to verify that for any $w \in \Sigma^*$, $M'$ accepts $w$ iff there exists $u, v \in \Sigma^*$ such that $M$ accepts $uwv$. For the “$\Rightarrow$” statement, the existence of $u$ is guaranteed by the “reachable from start state $q_0$” condition, and the existence of $v$ is guaranteed by the “connected to an accept state” condition.

(10 points for getting the accept states right, and 10 points for getting the start state and the $\epsilon$-transitions right).

4. Let $k$ be a positive integer. Let $\Sigma = \{0, 1\}$, and $L$ be the language consisting of all strings over $\{0, 1\}$ containing a 1 in the $k$th position from the end (in particular, all strings of length less than $k$ are not in $L$).

(a) Construct a DFA with exactly $2^k$ states that recognizes $L$.

(b) Construct a NFA with exactly $k + 1$ states that recognizes $L$.

**Solution Outline:** (20, 15 points)

(a) The idea is that we just need to keep track of the $k$ last input symbols that we’ve read, and designate a state for each of the $2^k$ possibilities.

---

1. The condition “reachable from the start state $q_0$” is essential. 2 points are taken off otherwise.
2. This is necessary because $M$ might have no accept state, and without doing this, $M'$ would not accept $\epsilon$. 


More precisely, consider the DFA $M$ with $Q = \{q_y \mid y \in \{0,1\}^k\}$, $q_0 = q_0^k$, and $F = \{q_y \mid y \text{ starts with a 1}\}$. The transition function is given by:

$$\delta(q_{y'}, b) = q_{y'ob}, \quad \forall y' \in \{0,1\}^{k-1}, \forall b, b' \in \{0,1\}$$

For the proof of correctness, it is easy to see by induction on the length of $x \in \{0,1\}^*$ that $M$ on input $x$ ends up in state $q_y$, where $y$ is the last $k$ symbols of $0^k \circ x$.

(10 points for specifying the right DFA, 10 points for the idea of keeping track of the last $k$ symbols and some explanation/proof that the DFA works.)

It is also in fact possible to derive a DFA with exactly $2^k$ states from the NFA for (b) using the transformation provided in the text. A careful accounting of the number of states is needed: the states in the DFA correspond to subsets of states in the NFA. In this specific case, we can eliminate all subsets except those containing $q_0$, which gives us $2^k$ instead of $2^{k+1}$ states in all.

(b) This is a straight-forward generalization of Example 1.14 in Sipser. Consider $Q = \{q_0, q_1, \ldots, q_k\}$. The start state is $q_0$ and the only accept state is $q_k$. Transition from $q_0$ to $q_0$ on input $0,1$; from $q_0$ to $q_1$ on input $1$, and from $q_i$ to $q_{i+1}$ on input $0,1$, for $i = 1,2,\ldots,k.$