1

(25 points) Give the state diagram of an NFA that recognizes the language $1^*(101^+)^*$, where the alphabet is $\{0, 1\}$. To receive full credits, your NFA needs to use at most three states (otherwise you will get at most 20 points). Note: You will get more partial credits if you design an NFA that recognizes the correct language rather than an NFA that recognizes the wrong language.

State Diagram:

Formal definition:
$M = (Q, \Sigma, \delta, q_s, F)$ where
$Q = \{q_0, q_1, q_2\}$
$\Sigma = \{0, 1\}$
$q_s = q_0$
$F = \{q_0\}$
Let $\Sigma = \{0, 1\}$. For any string $w \in \{0, 1\}^*$, let $w^R$ be $w$ written backwards. Give a regular expression for the following languages:

(a) $B = \{w | w$ has 10 as a substring and $w$ does not have two consecutive 1s}\}.

(b) $C = \{wtw^R | w, t \in \{0, 1\}^+\}$. Explain (informally) why your regular expression gives the correct language.

(a) 

$0^* (10^+)^+ (\epsilon \cup 1)$

(b) 

$0\Sigma^+ 0 \cup 1\Sigma^+ 1$

This is a correct construction because the language $C$ is equivalent to the following language:

$D = \{x | x$ has length at least 3 and starts and ends with the same bit\}

To see this, first notice that any string in $D$ is also in $C$ by just taking $w$ to be the first/last bit of $x$. Conversely, if we have some $w = w_0 \ldots w_n$ where $wtw^R \in C$ for some $t$, then this string $wtw^R$ also satisfies the property that it has length at least 3 and starts and ends with the same bit, so it’s also in $D$. 

3

(25 points) Let $\Sigma = \{0, 1\}$, consider the language $L = \{111\}$, i.e., $L$ contains a single string with three 1’s.

(a) Give an NFA with 4 states that recognizes $L$ (You can use either a state diagram or a formal description. You don’t need to prove it formally).

State Diagram:
(b) Show that no DFA with 4 states can recognize $L$. **Hint:** show that any DFA with 4 states that accepts 111 must also accept some other strings.

Suppose there exists a DFA $M = (Q, \{0, 1\}, \delta, q_0, F)$ which recognizes $L$ with 4 states.

Consider running $M$ on input 111. Assume $\delta(q_0, 1) = q_1, \delta(q_1, 1) = q_2, \delta(q_2, 1) = q_3$ for some $q_1, q_2, q_3$. Because $M$ accepts 111, $q_3 \in F$.

We claim that $q_0, q_1, q_2, q_3$ must be distinct. If not, then $M$ visits one state at least twice on reading 111. So there is a cycle on the path that $M$ goes through when reading 111. Assume the cycle has length $t$ for some $t > 0$. Then $1^{3+t}$ is in $L$, but $1^{3+t} \neq 111$, which is a contradiction. So the claim holds and we must have $Q = \{q_0, q_1, q_2, q_3\}$.

Now assume $\delta(q_3, 0) = q_i$ for some $i \in \{0, 1, 2, 3\}$. Then 11101$3-i$, which is not in $L$, is accepted by $M$. This is a contradiction. Thus, there is no DFA which recognizes $L$ with 4 states.

4

(25 points) Let $\Sigma = \{0, 1\}$. We define an operation **remove** on any string in $\Sigma^*$ that removes the leftmost bit if such a bit exists. For example, **remove**(010) is 10 and **remove**($\epsilon$) is $\epsilon$. Now for any language $L \subseteq \Sigma^*$, extend the operation as

$$\text{REMOVE}(L) = \{\text{remove}(w) | w \in L\}$$

Show that the class of regular languages is closed under the operation $\text{REMOVE}$.

Suppose $A$ is a regular language. Then there must exist some DFA $M = (Q, \{0, 1\}, \delta, q_0, F)$ which recognizes $A$, e.g. $L(M) = A$. We use $M$ to construct a NFA $N = (Q', \{0, 1\}, \delta', q_{\text{new}}, F'')$ where $q_{\text{new}}$ is a new state that we define, $Q' = Q \cup \{q_{\text{new}}\}$, $F'' = F \cup \{q_{\text{new}}\}$ if $q_0 \in F$ (if not, then just $F' = F$), and

$$\delta'(q, \sigma) = \begin{cases} 
\{\delta(q_0, 0), \delta(q_0, 1)\} & \text{if } \sigma = \epsilon \text{ and } q = q_{\text{new}} \\
\{\delta(q, \sigma)\} & \text{if } q \neq q_{\text{new}} \text{ and } \sigma \neq \epsilon \\
\emptyset & \text{all other cases}
\end{cases}$$

We now show that $L(N) = \text{REMOVE}(A)$ via $L(N) \subseteq \text{REMOVE}(A)$ and $\text{REMOVE}(A) \subseteq L(N)$.

**Proof.** ($L(N) \subseteq \text{REMOVE}(A)$)

If $w \in L(N)$ then $\exists r_0, \ldots, r_n$ where $r_0 = q_{\text{new}}, r_n \in F$, and $r_{i+1} \in \delta'(r_i, y_{i+1})$, where $y_i = \epsilon, z_{i+1} = w_i$ for $i \in \{1, \ldots, n\}$. $r_i \in \delta'(q_{\text{new}}, \sigma)$ for any $\sigma$ (only defined for $\sigma = \epsilon$) implies that $\delta(q_0, 0) = r_1$ or $\delta(q_0, 1) = r_1$ as per our definition of transition on $q_{\text{new}}$. Thus, we have that $r_0 = q_0, r_1, \ldots, r_n$ is a valid computation for the original DFA $M$, i.e., we have that $r_0 = q_0$ is the start state of $M$ and $r_n \in F$, but now $\delta'(r_i, z_{i+1}) = r_{i+1}$ for at least one of $z = 1w$ or $z = 0w$. Either way, we have $w \in \text{REMOVE}(A)$. Note that if $w = \epsilon$ is accepted by $N$, then
then one of \(\{q_{\text{new}}, \delta(q_0, 0), \delta(q_0, 1)\}\) has to be an accept state in \(N\) by our definition. Thus one of \(\{q_0, \delta(q_0, 0), \delta(q_0, 1)\}\) has to be an accept state in \(M\). In any case this will imply that \(\epsilon \in \text{REMOVE}(A)\) as well.

\[\square\]

\textbf{Proof.} \((\text{REMOVE}(A) \subseteq L(N))\)

Suppose \(w \in \text{REMOVE}(A)\). Then either \(0w\), \(1w\), or \(\epsilon\) is in \(A\) (the last case happens when \(w = \epsilon\) and is obtained by \(\text{remove}(\epsilon)\)). If \(\epsilon\) is in \(A\), then \(q_{\text{new}}\) is an accept state (since \(q_0\) was an accept state), so \(\epsilon\) would be accepted by \(N\) as well. Otherwise, \(\exists r_0, \ldots, r_{n+1}\) where \(r_0 = q_0, r_{n+1} \in F\), and \(\delta(r_i, w_i) = r_{i+1}\) for \(i \geq 1\). Furthermore, either \(r_1 = \delta(q_0, 0)\) or \(r_1 = \delta(q_0, 1)\), either way, \(r_{1} \in \{\delta(q_0, 0), \delta(q_0, 1)\}\). This means that \(r_{1} \in \delta'(q_{\text{new}}, \epsilon)\) which gives us that \(q_{\text{new}}, r_1, \ldots, r_{n+1}\) is a sequence where \(q_{\text{new}}\) is the start state of \(N\), \(r_{n+1} \in F'\) and \(r_{i+1} \in \delta'(r_i, w_i)\) for all \(i \in \{1, \ldots, n\}\). This is a valid computation for \(w\) on the machine \(N\) (since we take the first transition with an \(\epsilon\)). Thus \(w \in L(N)\). \(\square\)