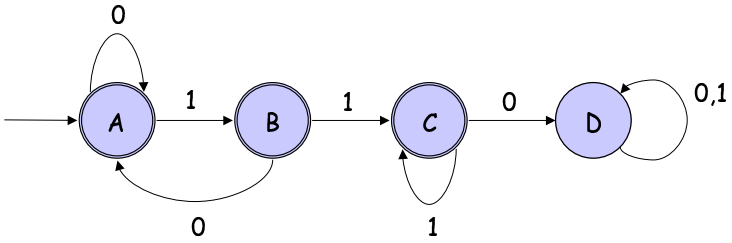
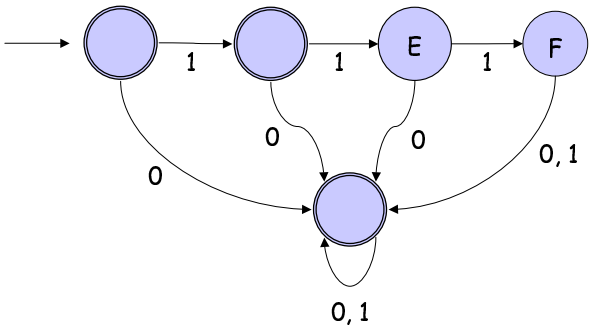
CS5371 Theory of Computation

Homework 1 (Suggested Solution)

1. (a) **Ans**: The state diagram for $\{w \mid w$ does not contain the substring `110`$\}$ is as follows. In the diagram, the states A, B, and C keep track of the ending characters of the current input until the substring `110` is seen; state D indicates that the input string contains the substring `110`.



(b) **Ans**: The state diagram for $\{w \mid w$ is any string except `11` and `111`$\}$ is as follows. In the diagram, state E is exactly reached when input string is `11`, and state F is exactly reached when the input string is `111`. Since E and F are the only non-accepting states, all strings except `11` and `111` are thus accepted.



2. (a) **Ans**: Let M' denote the DFA constructed by swapping the accept and nonaccept state in M. For any string $w \in B$, w will be accepted by M, so that processing w in M will exactly reach an accept state of M in the end. Thus, processing w in M' will exactly reach a nonaccept state of M' in the end, and w is therefore rejected by M'.

For any string $w \notin B$, w will be rejected by M, so that processing w in M will exactly reach a nonaccept state of M in the end. Thus, processing w in M' will exactly reach an accept state of M' in the end, and w is therefore rejected by M'.

In other words, M' is a DFA recognizing the complement of B. Thus, for any regular language L, there exists a DFA that recognizes its complement $\bar{L}$, which implies that $\bar{L}$ is also regular. We can therefore conclude that the class of regular language is closed under complement.

(b) **Ans**: Suppose that $\Sigma = \{0, 1\}$, and consider the NFA in Fig. 2(a). It will accept all strings in Σ^*. By swapping the accept and nonaccept states in this NFA, we obtain the NFA in Fig. 2(b). This new NFA will also accept all strings in Σ^*. The above result shows that swapping the accept and nonaccept states of the NFA for L is not

a proper way to obtain an NFA that recognizes its complement $\bar{L}$. However, the class of languages recognized by NFAs is still closed under complement, because it is equivalent to the class of regular languages.

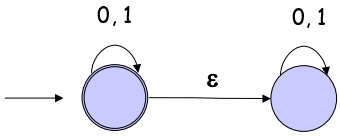


Fig. 2(a)

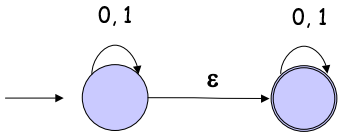


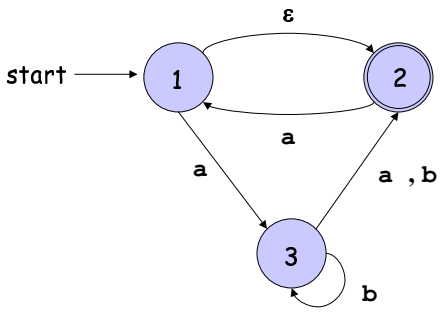
Fig. 2(b)

3. (a) **Ans**: The formal description of the NFA $= (\{1, 2, 3\}, \{\mathtt{a}, \mathtt{b}\}, \delta, 1, \{2\})$, where:

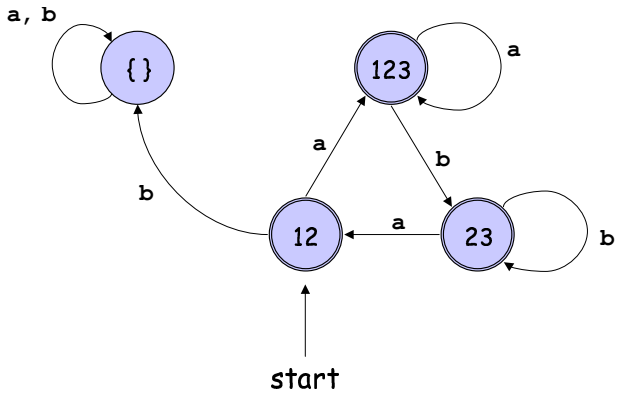
$$\delta(1, \mathtt{a}) = \{3\}, \quad \delta(1, \mathtt{b}) = \{\},$$
$$\delta(1, \varepsilon) = \{2\},$$

$$\delta(2, \mathtt{a}) = \{1\}, \quad \delta(2, \mathtt{b}) = \{\},$$
$$\delta(2, \varepsilon) = \{\},$$

$$\delta(3, \mathtt{a}) = \{2\}, \quad \delta(3, \mathtt{b}) = \{2, 3\},$$
$$\delta(3, \varepsilon) = \{\};$$



(b) **Ans**: The equivalent DFA (after minor simplifications) is as follows:



4. Suppose on the contrary that the language $A = \{www \mid w \in \{\mathtt{a}, \mathtt{b}\}^*\}$ is regular. Then A must satisfy the pumping lemma. In particular, let p be the pumping length for A.

   Consider the string $w = \mathtt{a}^p\mathtt{b}\mathtt{a}^p\mathtt{b}\mathtt{a}^p\mathtt{b}$, which is clearly a string in A. By the pumping lemma, there exists a way of partitioning w into x, y, z such that $w = xyz$, $|xy| \leq p$, $|y| > 0$, and for all $i \geq 0$, xy^iz is in A.

   By the above conditions, we know that y must be in the form $\mathtt{a}^t$ for some $1 \leq t \leq p$. Unfortunately, $xyyz = \mathtt{a}^{p+t}\mathtt{b}\mathtt{a}^p\mathtt{b}\mathtt{a}^p\mathtt{b}$ can never be a string in A, for any $t \geq 1$. Thus, a contradiction occurs (where??), so we can conclude that A is not regular.

5. (a) **Ans**: We claim that the language $\texttt{PAL} = \{w \mid w \in \{0,1\}^*$ is a palindrome$\}$ is not regular. Otherwise, the PAL must satisfy the pumping lemma. In particular, let p be the pumping length for PAL.

   Consider the string $w = 0^p10^p$, which is clearly a string in PAL. By the pumping lemma, there exists a way of partitioning w into x, y, z such that $w = xyz$, $|xy| \leq p$, $|y| > 0$, and for all $i \geq 0$, xy^iz is in PAL.

   By the above conditions, we know that y must be in the form 0^t for some $1 \leq t \leq p$. Unfortunately, for any $t \geq 1$, $xyyz = 0^{p+t}10^p$ can never be a palindrome, and thus not in PAL. Thus, a contradiction occurs (where??), so we can conclude that PAL is not regular.

   Since PAL is not regular, its complement must not be regular (why??). The completes the proof of the question.

   (b) **Ans**: Suppose on the contrary that the language $W = \{wtw \mid w, t \in \{0,1\}^+\}$ is regular. Then W must satisfy the pumping lemma. In particular, let p be the pumping length for W.

   Consider the string $s = 0^p110^p1$, which is clearly a string in W (setting $w = 0^p1, t = 1$). By the pumping lemma, there exists a way of partitioning s into x, y, z such that $s = xyz$, $|xy| \leq p$, $|y| > 0$, and for all $i \geq 0$, xy^iz is in W.

   By the above conditions, we know that y must be in the form 0^r for some $1 \leq r \leq p$. Then, the string $xyyz$ must be in the form $0^{p+r}110^p1$, for some $1 \leq r \leq p$. Now, for such a string to be in W so that it is in the form wtw, w needs to start with 0 and end with 1. From the prefix of $xyyz$, we conclude that the length of w must be at least $p + r + 1$, which is at least half the length of the whole string. Consequently, the length of t is at most 0. Thus, contradiction occurs (as $t \in \{0,1\}^+$ which implies $|t| > 0$), so we conclude that W is not regular.

6. (a) **Ans**: Consider the DFA for language A, and consider processing strings starting from the start state. Then, strings in $0, 1$ can be classified into three classes: (i) those that will exactly reach State 1, (ii) those that will exactly reach State 2, and (iii) those that will exactly reach State 3.

   For strings reaching State 1 or State 3, any one of them will reach an accept state of the DFA if 01 is further read from the input. As 01 is in B, by definition, all such strings reaching State 1 or State 3 are in A/B.

   For strings reaching State 2, any one of them will not reach an accept state of the DFA *no matter which string of B is further read from the input.* By definition, all such strings are not in A/B.

   Based on the above reasoning, we see that if we modify the DFA for language A by setting State 1 and State 3 to be accept state, and State 2 to be nonaccept state, the resulting DFA will recognize A/B.

   (b) **Ans**: For a state q in a DFA and a string w, we extend the notation of δ such that $\delta(q, w)$ denotes the state exactly reach when we process w starting at state q. In general, we can construct a DFA $M_{A/B}$ for A/B from the DFA M_A for A just by modifying the set of accept states. In particular, a state q is an accept state of $M_{A/B}$ if and only if for some $w \in B$, $\delta(q, w)$ is an accept state of M_A. Then, it is easy to check that this constructed DFA M_A recognizes the language A/B.