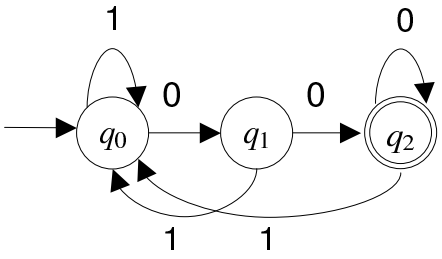
CS5371 Theory of Computation

Homework 1 (Solution)

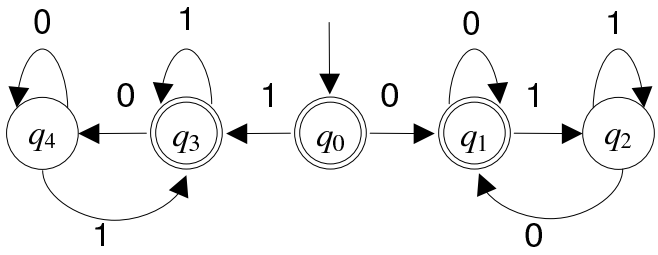
1. Assume that the alphabet is $\{0, 1\}$. Give the state diagram of a DFA that recognizes the language $\{w \mid w \text{ ends with } 00\}$.

   Answer: The key idea is to design three states q_0, q_1, q_2, where q_0 specifies the input string does not end with 0, q_1 specifies the input string ends with exactly one 0, and q_2 specifies the input string ends with at least two 0s.



2. Assume that the alphabet is $\{0, 1\}$. Give the state diagram of a DFA that recognizes the language $\{w \mid w \text{ contains an equal number of occurrences of the substrings } 01 \text{ and } 01\}$.

   Answer: The key observation is that the number of occurrences of the substrings 01 and 01 can differ by at most 1. In the figure below, the states q_2 and q_4 refer to the cases when the difference is exactly 1 (q_2 when 01 has one more occurrences, q_4 when 10 has one more occurrences).



3. Prove that the language $\{w^p \mid p \text{ is prime}\}$ is not regular. (You may assume that the number of primes is infinite.)

   Answer: Suppose on the contrary that this language L is regular. Let p be its pumping length and t be a prime greater than p. (Note that t exists since the number of primes is infinite.) Then, $w^t \in L$, and by the pumping lemma, we can write w^t as $w^t = xyz$ for some x, y, z, such that $|y| > 0$, $|xy| \leq p$, and for each $i \geq 0$, $xy^iz \in L$.

   By setting $i = t + 1$, the pumping lemma implies that $xy^{t+1}z$ is in L. On the other hand, $xy^{t+1}z = w^{t+t|y|}$, which implies that $xy^{t+1}z \notin L$, as $t + t|y| = t(1 + |y|)$ is not a prime. Thus, a contradiction occurs, so that L is not regular.

4. Consider the language $F = \{\mathtt{a}^i\mathtt{b}^j\mathtt{c}^k \mid i, j, k \geq 0 \text{ and if } i = 1 \text{ then } j = k\}$.

   (a) Show that F is not regular.

(b) Show that F acts like a regular language in the pumping lemma. In other words, give a pumping length p and demonstrate that F satisfies the three conditions of the pumping lemma for this value of p.

(c) Explain why parts (a) and (b) do not contradict the pumping lemma.

Answer:

(a) Suppose on the contrary that F is regular. Let $L = \{x \mid x \text{ begins with one } \mathtt{a}\}$. Obviously, L is regular. Recall that in the tutorial, we have proved that the intersection of two regular languages is regular, so the language $L' = F \cap L$ is regular.

Let p be the pumping length of L'. Note that $L' = \{\mathtt{ab}^n\mathtt{c}^n \mid n \geq 0\}$, so that $\mathtt{ab}^p\mathtt{c}^p$ is in L'. By pumping lemma, $\mathtt{ab}^p\mathtt{c}^p$ can be written as xyz such that $|y| > 0$, $|xy| \leq p$, and for each $i \geq 0$, $xy^iz \in L'$. However, (i) if y includes $\mathtt{a}$, the string $xyyz$ has at least two $\mathtt{a}$'s; (ii) else, if y includes both $\mathtt{b}$ and $\mathtt{c}$, the string $xyyz$ has at least two substrings $\mathtt{bc}$; (iii) else, y includes only $\mathtt{b}$ or only $\mathtt{c}$, and so the number of $\mathtt{b}$ and the number of $\mathtt{c}$ in $xyyz$ will not be equal. In all the above three cases, $xyyz$ cannot be in L', so that a contradiction occurs (why?). Thus, F is not regular.

(b) Let p' be the pumping length. Let $p = \max\{p', 3\}$ and let s be a string in F with $|s| > p$. We divide the discussion into two cases such that Case 1 corresponds to those string s that has exactly two '$\mathtt{a}$'s (i.e., when $i = 2$), and Case 2 corresponds to the other kinds of s (i.e., when $i \neq 2$). For Case 1, we can divide s into xyz, such that $x = \varepsilon$, $y = \mathtt{aa}$, and z is the remaining string. For Case 2, we can divide s into xyz, such that $x = \varepsilon$, y is the first character, and z is the remainder string. We can easily verify that in both cases, $|y| > 0$, $|xy| \leq p$, and for all $i \geq 0$, $xy^iz \in F$.

(c) All regular language satisfies the pumping lemma, but the converse is not true. That is, if a language satisfies the pumping lemma, the language may not be regular.

5. For languages A and B, let the *perfect shuffle* of A and B be the language

$$\{w \mid w = a_1b_1 \cdots a_kb_k, \text{ where } a_1 \cdots a_k \in A \text{ and } b_1 \cdots b_k \in B, \text{ each } a_i, b_i \in \Sigma\}.$$

Show that the class of regular languages is closed under perfect shuffle.

Answer: Let $D_A = (Q_A, \Sigma, \delta_A, q_A, F_A)$ and $D_B = (Q_B, \Sigma, \delta_B, q_B, F_B)$ be two DFAs that recognize A and B, respectively. Here, we shall construct a DFA $D = (Q, \Sigma, \delta, q, F)$ that recognizes the perfect shuffle of A and B.

The key idea is to design D to alternately switch from running D_A and running D_B after each character is read. Therefore, at any time, D needs to keep track of (i) the current states of D_A and D_B and (ii) whether the next character of the input string should be matched in D_A or in D_B. Then, when a character is read, depending on which DFA should match the character, D makes a move in the corresponding DFA accordingly. After the whole string is processed, if both DFAs are in the accept states, the input string is accepted; otherwise, the input string is rejected.

Formally, the DFA D can be defined as follows:

(a) $Q = Q_A \times Q_B \times \{A, B\}$, which keeps track of all possible current states of D_A and D_B, and which DFA to match.

(b) $q = (q_A, q_B, A)$, which states that D starts with D_A in q_A, D_B in q_B, and the next character read should be in D_A.

(c) $F = F_A \times F_B \times \{A\}$, which states that D accepts the string if both D_A and D_B are in accept states, and the next character read should be in D_A (i.e., last character was read in D_B).

(d) δ is as follows:

i. $\delta((x, y, A), a) = (\delta_A(x, a), y, B)$, which states that if current state of D_A is x, the current state of D_B is y, and the next character read is in D_A, then when a is read as the next character, we should change the current state of A to $\delta_A(x, a)$, while the current state of B is not changed, and the next character read will be in D_B.

ii. Similarly, $\delta((x, y, B), b) = (x, \delta_B(y, b), A)$.

6. For languages A and B, let the *shuffle* of A and B be the language

$$\{w \mid w = a_1 b_1 \cdots a_k b_k, \text{ where } a_1 \cdots a_k \in A \text{ and } b_1 \cdots b_k \in B, \text{ each } a_i, b_i \in \Sigma^*\}.$$

Show that the class of regular languages is closed under shuffle.

Answer: Let $D_A = (Q_A, \Sigma, \delta_A, q_A, F_A)$ and $D_B = (Q_B, \Sigma, \delta_B, q_B, F_B)$ be two DFAs that recognize A and B, respectively. Similar to the previous question, we shall prove by construction. However, the key difference is that D *may* now switch from running D_A and running D_B after each character is read. To allow this flexibility and simplify the construction, we design an NFA $N = (Q, \Sigma, \delta, q, F)$ that recognizes the shuffle of A and B instead of directly designing a DFA.

At any time, N needs to keep track of the current states of D_A and D_B. Then, when a character is read, N may make a move in D_A or D_B accordingly. After the whole string is processed, if both DFAs are in the accept states, the input string is accepted; otherwise, the input string is rejected. In addition, N should also accept the empty string.

Formally, the NFA N can be defined as follows:

(a) $Q = (Q_A \times Q_B) \cup \{q_0\}$, where $Q_A \times Q_B$ keeps track of all possible current states of D_A and D_B, and q_0 denotes the state when nothing is read.

(b) $q = q_0$.

(c) $F = (F_A \times F_B) \cup \{q_0\}$, which states that N accepts the string if both D_A and D_B are in accept states, or N accepts the empty string.

(d) δ is as follows:

i. $\delta(q_0, \varepsilon) = (q_A, q_B)$, which states that at the start state q_0, N can make D_A in q_A and D_B in q_B without reading anything.

ii. $(\delta_A(x, a), y) \in \delta((x, y), a)$, which states that if current state of D_A is x, the current state of D_B is y, then when a is read as the next character, we can change the current state of A to $\delta_A(x, a)$, while the current state of B is not changed.

iii. Similarly, $(x, \delta_B(y, a)) \in \delta((x, y), a)$.

7. **(Myhill-Nerode Theorem.)** Let L be any language.

Definition 1. *Let x and y be strings. We say that x and y are* distinguishable by L *if some string z exists whereby exactly one of the strings xz and yz is a member of L; otherwise, for every string z, we have $xz \in L$ whenever $yz \in L$.*

Definition 2. *Let X be a set of strings. We say X is* pairwise distinguishable by L *if every two distinct strings in X are distinguishable by L.*

Definition 3. *Define the* index *of L to be the maximum number of elements in any set of strings that is pairwise distinguishable by L. The index of L may be finite or infinite.*

(a) Show that if L is recognized by a DFA with k states, L has index at most k.

(b) Show that, if the index of L is a finite number k, then it is recognized by a DFA with k states.

(c) Conclude that L is regular if and only if it has finite index. Moreover, its index is the size of the smallest DFA recognizing it.

Answer: The following answers are extracted from the textbook.

(a) Let M be the DFA with k states recognizing L. Suppose on the contrary that L has index greater than k. That means, some set X with at least $k+1$ strings is pairwise distinguishable by L. By pigeonhole's principle, we can find two distinct strings x and y from X, such that the state of M after reading x as input is the same as the state of M after reading y as input. Therefore, both xz and yz are in L or neither are in L. This implies x and y are not distinguishable by L. Contradiction occurs, so that L has index at most k.

(b) Let $X = \{x_1, x_2, \ldots, x_k\}$ be pairwise distinguishable by L. We construct DFA $M = (Q, \Sigma, \delta, q_0, F)$ with k states recognizing L as follows. Let $Q = \{q_1, q_2, \ldots, q_k\}$, and define $\delta(q_i, a)$ to be q_j, if $x_i a$ and x_j are not distinguishable. Let $F = \{q_i \mid x_i \in L\}$. Let the start state q_0 be the (unique) state such that x_i and and the empty string ε are not distinguishable by L.

We can show that if a string s and x_j are not distinguishable by L, the state of M after reading s as input will be q_j (how to show?). Then, by the definition of F, M accepts s if and only if s is in L (why?). Thus, M recognizes L.

(c) Suppose that L is regular and let k be the number of states in a DFA recognizing L. Then from part (a), L has index at most k. On the other hand, if L has index k, then from part (b) we can construct a DFA with k states recognizing L. Thus, L is regular if and only if it has finite index.

To see why the index k is the size of the smallest DFA recognizing it, suppose on the contrary that it is not true. Then, from part (a) we would conclude that L has index fewer than k, which contradicts with the fact that L has index equal to k.

8. (Bonus Question.) If A is a language, let $A_{\frac{1}{2}}$ be the set of all first halves of strings in A, so that

$$A_{\frac{1}{2}} = \{x \mid \text{ for some } y, |x| = |y| \text{ and } xy \in A\}.$$

Show that if A is regular, then so is $A_{\frac{1}{2}}$.

Answer: Let $D = (Q_A, \Sigma, \delta_A, q_A, F_A)$ be a DFA recognizing A. We shall construct an NFA N that recognizes $A_{\frac{1}{2}}$. The idea is that, when we have processed the ith input characters, N is able to keep track both the state of D when processed the string so far, and the possible states which can reach some accept state of D in i steps. Then, a string is accepted by N when the current state is in one of these possible states.

Formally, we let $N = (Q, \Sigma, \delta, q, F)$ such that

(a) $Q = (Q_A \times Q_A) \cup \{q_0\}$, where $Q_A \times Q_A$ keeps track of the current state of D, and the state that can reach an accept state of D in i steps, where i is the length of the input string processed so far. In addition, we create a state q_0, which denotes the state when nothing is read.

(b) $q = q_0$.

(c) $F = \{(x, x) \mid x \in Q_A\}$, which states that a string is accepted when the current state of D is at x, and x is i steps from some accept state of D, where i is the length of the input string processed so far.

(d) δ is as follows:

i. $(q_A, x) \in \delta(q_0, \varepsilon)$ for $x \in F_A$, which states that without reading anything, we make D to start at q_A, and keep track that $x \in F$ is 0 steps from some accept state of D.

ii. $(\delta_A(x, a), z) \in \delta((x, y), a)$ for any z such that there exists some $c \in \Sigma$ with $\delta_A(z, c) = y$. This states that when D advances one step from state x to $\delta_A(x, a)$, we update the state y to some state z which is one more step further from the accept state of D.