CS5371 Theory of Computation

Homework 2 (Suggested Solution)

1. **Ans.** We completed the proof by showing Claim 1 and Claim 2 as follows:

   Claim 1. *If s is a string accepted by D, then V_0 derives s.*

   Proof. Let $k = |s|$. When $k = 0$, $s = \varepsilon$, which is accepted by D only if q_0 is an accept state. In this case, the CFG contains the rule $V_0 \to \varepsilon$ so that V_0 derives s.

   When $k \geq 1$, let $s = s_1 s_2 \cdots s_k$. Since s is accepted by D, there will be a sequence of states $q_{i_0}, q_{i_1}, \ldots, q_{i_k}$ in D such that $q_{i_0} = q_0$, q_{i_k} is an accept state, and for each j, $q_{i_j} = \delta(q_{i_{j-1}}, s_i)$. By the construction of CFG, we know that $V_{i_0} = V_0$ is the start variable, and there is a rule $V_{i_k} \to \varepsilon$. Also, there is a rule $V_{i_{j-1}} \to s_i V_{i_j}$ for each j. This implies that

   $$V_0 \Rightarrow s_1 V_{i_1} \Rightarrow s_1 s_2 V_{i_2} \Rightarrow \cdots \Rightarrow s_1 s_2 \cdots s_k V_{i_k} \Rightarrow s_1 s_2 \cdots s_k.$$

   In other words, V_0 derives s. □

   Claim 2. *If V_0 derives s, then s is a string accepted by D.*

   Proof. Let $k = |s|$. When $k = 0$, $s = \varepsilon$, which is derived from V_0 only if there is a rule $V_0 \to \varepsilon$. In this case, q_0 is an accept state in D, so s is accepted by D.

   When $k \geq 1$, let $s = s_1 s_2 \cdots s_k$. Since s is derived from V_0, and since each rule replaces a variable by either `(i)` a terminal followed by another variable, or `(ii)` an empty string, the derivation of s must be in the following form:

   $$V_0 \Rightarrow s_1 V_{i_1} \Rightarrow s_1 s_2 V_{i_2} \Rightarrow \cdots \Rightarrow s_1 s_2 \cdots s_k V_{i_k} \Rightarrow s_1 s_2 \cdots s_k,$$

   for some $i_1, i_2, \ldots, i_k$. By letting $i_0 = 0$, the above form implies that q_{i_k} is an accept state, and for each j, $q_{i_j} = \delta(q_{i_{j-1}}, s_i)$. Since $q_{i_0}, q_{i_1}, \ldots, q_{i_k}$ is exactly the sequence of states visited in D when we process s, therefore s is accepted by D. □

2. (a) **Ans.** A string in the complement of the language $\{\mathtt{a}^n\mathtt{b}^n \mid n \geq 0\}$ must be in one of the following forms: `(i)` contains a substring `ba`; or `(ii)` equals to $\mathtt{a}^i\mathtt{b}^j$ for some $i \neq j$. Accordingly, the CFG we use is as follows:

   $$\begin{array}{rcl} S & \to & S_1 \mid S_2 \\ S_1 & \to & \mathtt{ba} \mid XS_1 \mid S_1X \\ X & \to & \mathtt{a} \mid \mathtt{b} \\ S_2 & \to & AC \mid CB \\ A & \to & \mathtt{a}A \mid \mathtt{a} \\ B & \to & \mathtt{b}B \mid \mathtt{b} \\ C & \to & \mathtt{ab} \mid \mathtt{a}C\mathtt{b} \end{array}$$

(b) **Ans.** A string in the language

$$\{x_1\#x_2\#\cdots\#x_k \mid k \geq 1,\ \text{each } x_i \in \{\mathtt{a},\mathtt{b}\}^*,\ \text{and for some } i \text{ and } j, x_i = x_j^R\}$$

must be in one of the following forms: (i) contains a palindrome x_i for some i; (ii) contains distinct i and j such that $x_i = x_j^R$. Accordingly, the CFG we use is as follows:

$$\begin{aligned}
S &\rightarrow S_1 \mid S_2 \\
S_1 &\rightarrow P \mid X\#P \mid P\#X \\
P &\rightarrow \mathtt{a}P\mathtt{a} \mid \mathtt{b}P\mathtt{b} \mid \mathtt{a} \mid \mathtt{b} \mid \varepsilon \\
X &\rightarrow \mathtt{a} \mid \mathtt{b} \mid \# \mid \varepsilon \\
S_2 &\rightarrow M \mid X\#M \mid M\#X \\
M &\rightarrow \mathtt{a}M\mathtt{a} \mid \mathtt{b}M\mathtt{b} \mid \#X\#
\end{aligned}$$

3. **Ans.** An equivalent CFG in Chomsky normal form is as follows:

$$\begin{aligned}
S &\rightarrow AB \mid BA \mid BB \mid BC \mid DD \mid \varepsilon \\
A &\rightarrow AB \mid BA \mid BB \mid BC \mid DD \\
B &\rightarrow DD \\
C &\rightarrow AB \\
D &\rightarrow \mathtt{0}
\end{aligned}$$

4. **Ans.** A string is in the language $C = \{x\#y \mid x, y \in \{\mathtt{0},\mathtt{1}\}^* \text{ and } x \neq y\}$ if and only if $|x| \neq |y|$, or for some i, the ith bit of x is different with the ith bit of y. Accordingly, we design a CFG for C, thus showing that C is context-free:

$$\begin{aligned}
S &\rightarrow M \mid N\mathtt{1}A \mid P\mathtt{0}A \\
M &\rightarrow XMX \mid U \\
X &\rightarrow \mathtt{0} \mid \mathtt{1} \\
U &\rightarrow \#\mathtt{0}A \mid \#\mathtt{1}A \mid A\mathtt{0}\# \mid A\mathtt{1}\# \\
A &\rightarrow \mathtt{0}A \mid \mathtt{1}A \mid \varepsilon \\
N &\rightarrow XNX \mid \mathtt{0}A\# \\
P &\rightarrow XPX \mid \mathtt{1}A\#
\end{aligned}$$

In the above CFG, A generates strings of any length, M generates x and y of unequal length by first matching characters in the shorter of them with the longer one, and then generates the remaining characters of the longer (using U). N generates strings in the form of $\{\mathtt{0},\mathtt{1}\}^i\mathtt{0}\{\mathtt{0},\mathtt{1}\}^*\#\{\mathtt{0},\mathtt{1}\}^i$ so that when combined with $\mathtt{1}A$, we obtain a string with ith bit of x is $\mathtt{0}$, while ith bit of y is $\mathtt{1}$. P generates strings similar to N, except the ith bit is $\mathtt{1}$.

5. (a) **Ans.** Assume that the language is context-free. Let p be the pumping length. Consider the string $s = \mathtt{0}^p\mathtt{1}^p\mathtt{0}^p\mathtt{1}^p$ which is in the language with length at least p.

Suppose that s is partitioned into $s = uvxyz$ with $|vxy| \leq p$ and $|vy| > 0$. There are two cases:

(i) Both v and y contains only one type of characters.

(ii) Either v or y contains two types of characters.

For the first case, if we consider s as four maximal substrings with contiguous 0s or 1s, we can see that deleting v and y from s can change the number of 0s or 1s in one or two such substrings, so that uxz is not in the language.

For the second case, since $|vxy| \leq p$, we see that v and y cannot both contain two types of characters. Without loss of generality, let v to be the string with two types of characters. Then, the string $uvvxyyz$ will be in the form $\mathtt{0}^a\mathtt{1}^b\mathtt{0}^c\mathtt{1}^d\mathtt{0}^e\mathtt{1}^f$, so that it is not in the language.

Thus, we find a string in the language which does not satisfy the pumping lemma. Contradiction occurs, so that we conclude the language is not context-free.

(b) **Ans.** Assume that the language is context-free. Let p be the pumping length. Consider the string $s = \mathtt{0}^p\mathtt{1}^p\#\mathtt{0}^p\mathtt{1}^p$ which is in the form $x_1\#x_1$. Thus, s is in the language, and also with length at least p.

Suppose that s is partitioned into $s = uvxyz$ with $|vxy| \leq p$ and $|vy| > 0$. There are two cases:

(i) One of v or y contains #

(ii) Both v and y does not contain #

For case (i), the string uxz must not be in the language, since it does not contain #.

For case (ii), since v and y are within p characters, deleting v and y will not delete the same characters in the corresponding portions of x_1 in s. As a result, the string uxz must not be in the language.

Thus, we find a string in the language which does not satisfy the pumping lemma. Contradiction occurs, so that we conclude the language is not context-free.