CS 341: Foundations of Computer Science II
Prof. Marvin Nakayama

Homework 5 Solutions

1. Give context-free grammars that generate the following languages. Each context-free language has infinitely many correct CFGs, but you only need to provide one.

   (a) $\{\, w \in \{0,1\}^* \mid w \text{ contains at least three } 1\text{s} \,\}$

   Answer: $G = (V, \Sigma, R, S)$ with set of variables $V = \{S, X\}$, where S is the start variable; set of terminals $\Sigma = \{0,1\}$; and rules

   $$\begin{aligned} S &\rightarrow X1X1X1X \\ X &\rightarrow 0X \mid 1X \mid \varepsilon \end{aligned}$$

   (b) $\{\, w \in \{0,1\}^* \mid w = w^{\mathcal{R}} \text{ and } |w| \text{ is even} \,\}$

   Answer: $G = (V, \Sigma, R, S)$ with set of variables $V = \{S\}$, where S is the start variable; set of terminals $\Sigma = \{0,1\}$; and rules

   $$S \rightarrow 0S0 \mid 1S1 \mid \varepsilon$$

   (c) $\{\, w \in \{0,1\}^* \mid \text{the length of } w \text{ is odd and the middle symbol is } 0 \,\}$

   Answer: $G = (V, \Sigma, R, S)$ with set of variables $V = \{S\}$, where S is the start variable; set of terminals $\Sigma = \{0,1\}$; and rules

   $$S \rightarrow 0S0 \mid 0S1 \mid 1S0 \mid 1S1 \mid 0$$

   (d) $\{\, a^i b^j c^k \mid i,j,k \geq 0, \text{ and } i = j \text{ or } i = k \,\}$

   Answer: $G = (V, \Sigma, R, S)$ with set of variables $V = \{S, W, X, Y, Z\}$, where S is the start variable; set of terminals $\Sigma = \{a, b, c\}$; and rules

   $$\begin{aligned} S &\rightarrow XY \mid W \\ X &\rightarrow aXb \mid \varepsilon \\ Y &\rightarrow cY \mid \varepsilon \\ W &\rightarrow aWc \mid Z \\ Z &\rightarrow bZ \mid \varepsilon \end{aligned}$$

(e) $\{\, a^i\, b^j\, c^k \mid i, j, k \geq 0 \text{ and } i + j = k \,\}$

Answer: $G = (V, \Sigma, R, S)$ with set of variables $V = \{S, X\}$, where S is the start variable; set of terminals $\Sigma = \{a, b, c\}$; and rules

$$\begin{array}{rcl} S & \rightarrow & aSc \mid X \\ X & \rightarrow & bXc \mid \varepsilon \end{array}$$

(f) $\{\, a^i\, b^j\, c^k \mid i, j, k \geq 0 \text{ and } i + k = j \,\}$ [Hint: use problem 3b.]

Answer: Let $L = \{\, a^i\, b^j\, c^k \mid i, j, k \geq 0 \text{ and } i + k = j \,\}$ be the language given in the problem, and define other languages

$$L_1 = \{\, a^i\, b^i \mid i \geq 0 \,\},$$
$$L_2 = \{\, b^k\, c^k \mid k \geq 0 \,\}.$$

Note that $L = L_1 \circ L_2$ because concatenating any string $a^i b^i \in L_1$ with any string $b^k c^k \in L_2$ results in a string $a^i b^i b^k c^k = a^i b^{i+k} c^k \in L$. Thus, if L_1 has a CFG $G_1 = (V_1, \Sigma, R_1, S_1)$, and L_2 has a CFG $G_2 = (V_2, \Sigma, R_2, S_2)$, we can construct a CFG for $L = L_1 \circ L_2$ by using the approach in problem 3b, as suggested in the hint. Specifically,

- L_1 has a CFG $G_1 = (V_1, \Sigma, R_1, S_1)$, with $V_1 = \{S_1\}$, $\Sigma = \{a, b, c\}$, S_1 as the starting variable, and rules $S_1 \rightarrow aS_1b \,|\, \varepsilon$ in R_1;
- L_2 has a CFG $G_2 = (V_2, \Sigma, R_2, S_2)$, with $V_2 = \{S_2\}$, $\Sigma = \{a, b, c\}$, S_2 as the starting variable, and rules $S_2 \rightarrow bS_2c \,|\, \varepsilon$ in R_2.

Even though $\Sigma = \{a, b, c\}$ for both CFGs G_1 and G_2, CFG G_1 never generates a string with c, and CFG G_2 never generates a string with a. Then from problem 3b, a CFG $G_3 = (V_3, \Sigma, R_3, S_3)$ for L has $V_3 = V_1 \cup V_2 \cup \{S_3\} = \{S_1, S_2, S_3\}$ with S_3 the starting variable, $\Sigma = \{a, b, c\}$, and rules

$$S_3 \rightarrow S_1 S_2$$
$$S_1 \rightarrow aS_1b \,|\, \varepsilon$$
$$S_2 \rightarrow bS_2c \,|\, \varepsilon$$

(g) $\{\, ab^n acab^n a \mid n \geq 0 \,\}$.

Answer: $G = (V, \Sigma, R, S)$ with set of variables $V = \{S, T\}$, where S is the start variable; set of terminals $\Sigma = \{a, b, c\}$; and rules

$$\begin{array}{rcl} S & \rightarrow & aTa \\ T & \rightarrow & bTb \mid aca \end{array}$$

(h) $\emptyset$

Answer: $G = (V, \Sigma, R, S)$ with set of variables $V = \{S\}$, where S is the start variable; set of terminals $\Sigma = \{0, 1\}$; and rules

$$S \rightarrow S$$

Note that if we start a derivation, it never finishes, i.e., $S \Rightarrow S \Rightarrow S \Rightarrow \cdots$, so no string of terminals is ever produced. Thus, $L(G) = \emptyset$.

(i) The language A of strings of properly balanced left and right brackets: every left bracket can be paired with a unique subsequent right bracket, and every right bracket can be paired with a unique preceding left bracket. Moreover, the string between any such pair has the same property. For example, $[\,]\,[\,[\,[\,]\,[\,]\,]\,[\,]\,] \in A$.

Answer: $G = (V, \Sigma, R, S)$ with set of variables $V = \{S\}$, where S is the start variable; set of terminals $\Sigma = \{[,]\}$; and rules

$$S \rightarrow \varepsilon \mid SS \mid [S]$$

2. Let $T = \{\, 0,\ 1,\ (,\ ),\ \cup,\ ^*,\ \emptyset,\ e \,\}$. We may think of T as the set of symbols used by regular expressions over the alphabet $\{0, 1\}$; the only difference is that we use e for symbol ε, to avoid potential confusion in what follows.

(a) Your task is to design a CFG G with set of terminals T that generates exactly the regular expressions with alphabet $\{0, 1\}$.

Answer: $G = (V, \Sigma, R, S)$ with set of variables $V = \{S\}$, where S is the start variable; set of terminals $\Sigma = T$; and rules

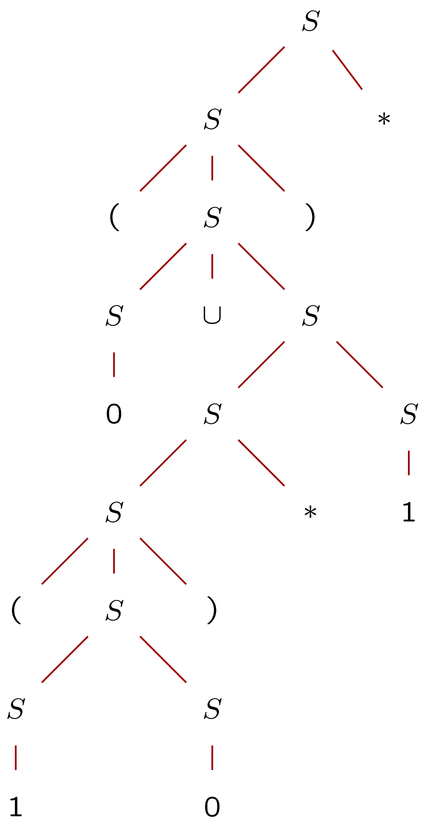
$$S \rightarrow S \cup S \mid SS \mid S^* \mid (S) \mid 0 \mid 1 \mid \emptyset \mid e$$

(b) Using your CFG G, give a derivation and the corresponding parse tree for the string $(0 \cup (10)^*1)^*$.

Answer: A derivation for $(0 \cup (10)^*1)^*$ is

$$\begin{aligned} S &\Rightarrow S^* \Rightarrow (S)^* \Rightarrow (S \cup S)^* \Rightarrow (0 \cup S)^* \Rightarrow (0 \cup SS)^* \Rightarrow (0 \cup S^*S)^* \\ &\Rightarrow (0 \cup (S)^*S)^* \Rightarrow (0 \cup (SS)^*S)^* \Rightarrow (0 \cup (1S)^*S)^* \\ &\Rightarrow (0 \cup (10)^*S)^* \Rightarrow (0 \cup (10)^*1)^* \end{aligned}$$

and the corresponding parse tree is



3. (a) Suppose that language A_1 has a context-free grammar $G_1 = (V_1, \Sigma, R_1, S_1)$, and language A_2 has a context-free grammar $G_2 = (V_2, \Sigma, R_2, S_2)$, where, for $i = 1, 2$, V_i is the set of variables, R_i is the set of rules, and S_i is the start variable for CFG G_i. The CFGs have the same set of terminals Σ. Assume that $V_1 \cap V_2 = \emptyset$. Define another CFG $G_3 = (V_3, \Sigma, R_3, S_3)$ with $V_3 = V_1 \cup V_2 \cup \{S_3\}$, where $S_3 \notin V_1 \cup V_2$, and $R_3 = R_1 \cup R_2 \cup \{\, S_3 \to S_1,\ S_3 \to S_2 \,\}$. Argue that G_3 generates the language $A_1 \cup A_2$. Thus, conclude that the class of context-free languages is closed under union.

Answer: Let $A_3 = A_1 \cup A_2$, and we need to show that $L(G_3) = A_3$. To do this, we need to prove that $L(G_3) \subseteq A_3$ and $A_3 \subseteq L(G_3)$. To show that $L(G_3) \subseteq A_3$, first consider any string $w \in L(G_3)$. Since $w \in L(G_3)$, we have that $S_3 \overset{*}{\Rightarrow} w$. Since the only rules in R_3 with S_3 on the left side are $S_3 \to S_1 \mid S_2$, we must have that $S_3 \Rightarrow S_1 \overset{*}{\Rightarrow} w$ or $S_3 \Rightarrow S_2 \overset{*}{\Rightarrow} w$. Suppose first that $S_3 \Rightarrow S_1 \overset{*}{\Rightarrow} w$. Since $S_1 \in V_1$ and we assumed that $V_1 \cap V_2 = \emptyset$, the derivation $S_1 \overset{*}{\Rightarrow} w$ must only use variables in V_1 and rules in R_1, which implies that $w \in A_1$. Similarly, if $S_3 \Rightarrow S_2 \overset{*}{\Rightarrow} w$, then we must have that $w \in A_2$. Thus, $w \in A_3 = A_1 \cup A_2$, so $L(G_3) \subseteq A_3$.

To show that $A_3 \subseteq L(G_3)$, first suppose that $w \in A_3$. This implies $w \in A_1$ or $w \in A_2$. If $w \in A_1$, then $S_1 \overset{*}{\Rightarrow} w$. But then $S_3 \Rightarrow S_1 \overset{*}{\Rightarrow} w$, so $w \in L(G_3)$.

Similarly, if $w \in A_2$, then $S_2 \overset{*}{\Rightarrow} w$. But then $S_3 \Rightarrow S_2 \overset{*}{\Rightarrow} w$, so $w \in L(G_3)$. Thus, $A_3 \subseteq L(G_3)$, and since we previously showed that $L(G_3) \subseteq A_3$, it follows that $L(G_3) = A_3$; i.e., the CFG G_3 generates the language $A_1 \cup A_2$.

(b) Prove that the class of context-free languages is closed under concatenation.

Answer: Suppose that language A_1 has a context-free grammar $G_1 = (V_1, \Sigma, R_1, S_1)$, and language A_2 has a context-free grammar $G_2 = (V_2, \Sigma, R_2, S_2)$, where, for $i = 1, 2$, V_i is the set of variables, R_i is the set of rules, and S_i is the start variable for CFG G_i. The CFGs have the same set of terminals Σ. Assume that $V_1 \cap V_2 = \emptyset$. Then a CFG for $A_1 \circ A_2$ is $G_3 = (V_3, \Sigma, R_3, S_3)$ with $V_3 = V_1 \cup V_2 \cup \{S_3\}$, where $S_3 \notin V_1 \cup V_2$, and $R_3 = R_1 \cup R_2 \cup \{ S_3 \to S_1 S_2 \}$.

To understand why $L(G_3) = A_1 \circ A_2$, note that any string $w \in A_1 \circ A_2$ can be written as $w = uv$, where $u \in A_1$ and $v \in A_2$. It follows that $S_1 \overset{*}{\Rightarrow} u$ and $S_2 \overset{*}{\Rightarrow} v$, so $S_3 \Rightarrow S_1 S_2 \overset{*}{\Rightarrow} u S_2 \overset{*}{\Rightarrow} uv$, so $w = uv \in L(G_3)$. This proves that $A_1 \circ A_2 \subseteq L(G_3)$.

To prove that $L(G_3) \subseteq A_1 \circ A_2$, consider any string $w \in L(G_3)$. Since $w \in L(G_3)$, it follows that $S_3 \overset{*}{\Rightarrow} w$. The only rule in R_3 with S_3 on the left side is $S_3 \to S_1 S_2$, so $S_3 \Rightarrow S_1 S_2 \overset{*}{\Rightarrow} w$. Since $V_1 \cap V_2 = \emptyset$, any derivation starting from S_1 can only generate a string in A_1, and any derivation starting from S_2 can only generate a string in A_2. Thus, since $S_3 \Rightarrow S_1 S_2 \overset{*}{\Rightarrow} w$, it must be that w is the concatenation of a string from A_1 with a string from A_2. Therefore, $w \in A_1 \circ A_2$, which establishes that $L(G_3) \subseteq A_1 \circ A_2$.

(c) Prove that the class of context-free languages is closed under Kleene-star.

Answer: Suppose that language A has a context-free grammar $G_1 = (V_1, \Sigma, R_1, S_1)$. Then a CFG for A^* is $G_2 = (V_2, \Sigma, R_2, S_2)$ with $V_2 = V_1 \cup \{S_2\}$, where $S_2 \notin V_1$, and $R_2 = R_1 \cup \{ S_2 \to S_1 S_2,\ S_2 \to \varepsilon \}$.

To show that $L(G_2) = A^*$, we first prove that $A^* \subseteq L(G_2)$. Consider any string $w \in A^*$. We can write $w = w_1 w_2 \cdots w_n$ for some $n \geq 0$, where each $w_i \in A$. (Here, we interpret $w = w_1 w_2 \cdots w_n$ for $n = 0$ to be $w = \varepsilon$.) Since each $w_i \in A$, we have that $S_1 \overset{*}{\Rightarrow} w_i$. To derive the string w using CFG G_2, we first apply the rule $S_2 \to S_1 S_2$ a total of n times, followed by one application of the rule $S_2 \to \varepsilon$. Then for the ith S_1, we use $S_1 \overset{*}{\Rightarrow} w_i$. Thus, we get

$$S_2 \overset{*}{\Rightarrow} \underbrace{S_1 S_1 \cdots S_1}_{n \text{ times}} S_2 \Rightarrow \underbrace{S_1 S_1 \cdots S_1}_{n \text{ times}} \overset{*}{\Rightarrow} w_1 w_2 \cdots w_n = w$$

Therefore, $w \in L(G_2)$, so $A^* \subseteq L(G_2)$.

To show that $L(G_2) \subseteq A^*$, suppose we apply the rule $S \to S_1 S_2$ a total of $n \geq 0$ times, followed by an application of the rule $S_2 \to \varepsilon$. This gives

$$S_2 \overset{*}{\Rightarrow} \underbrace{S_1 S_1 \cdots S_1}_{n \text{ times}} S_2 \Rightarrow \underbrace{S_1 S_1 \cdots S_1}_{n \text{ times}}.$$

Now each of the variables S_1 can be used to derive a string $w_i \in A$, i.e., from the ith S_1, we get $S_1 \overset{*}{\Rightarrow} w_i$. Thus,

$$S_2 \overset{*}{\Rightarrow} \underbrace{S_1 S_1 \cdots S_1}_{n \text{ times}} \overset{*}{\Rightarrow} w_1 w_2 \cdots w_n \in A^*$$

since each $w_i \in A$. Therefore, we end up with a string in A^*. To convince ourselves that the productions applied to the various separate S_1 terms do not interfere in undesired ways, we need only think of the parse tree. Each S_1 is the root of a distinct branch, and the rules along one branch do not affect those on another. Here, we assumed that we first applied the rule $S_2 \to S_1 S_2$ a total of n times, then applied the rule $S_2 \to \varepsilon$, and then applied rules to change each S_1 into strings. However, we could have applied the rules in a different order, as long as the rule $S_2 \to \varepsilon$ is applied only after the n applications of $S_2 \to S_1 S_2$. By examining the parse tree, we can argue as before that the order in which we applied the rules doesn't matter.

4. Convert the following CFG into an equivalent CFG in Chomsky normal form, using the procedure given in Theorem 2.9.

$$\begin{aligned} S &\to BSB \mid B \mid \varepsilon \\ B &\to \mathsf{00} \mid \varepsilon \end{aligned}$$

Answer: First introduce new start variable S_0 and the new rule $S_0 \to S$, which gives

$$\begin{aligned} S_0 &\to S \\ S &\to BSB \mid B \mid \varepsilon \\ B &\to \mathsf{00} \mid \varepsilon \end{aligned}$$

Then we remove ε rules:

- Removing $B \to \varepsilon$ yields

$$\begin{aligned} S_0 &\to S \\ S &\to BSB \mid BS \mid SB \mid S \mid B \mid \varepsilon \\ B &\to \mathsf{00} \end{aligned}$$

- Removing $S \to \varepsilon$ yields

$$\begin{aligned} S_0 &\to S \mid \varepsilon \\ S &\to BSB \mid BS \mid SB \mid S \mid B \mid BB \\ B &\to \mathsf{00} \end{aligned}$$

- We don't need to remove the ε-rule $S_0 \to \varepsilon$ since S_0 is the start variable and that is allowed in Chomsky normal form.

Then we remove unit rules:

- Removing $S \to S$ yields

$$\begin{array}{rcl} S_0 & \to & S \mid \varepsilon \\ S & \to & BSB \mid BS \mid SB \mid B \mid BB \\ B & \to & \mathsf{00} \end{array}$$

- Removing $S \to B$ yields

$$\begin{array}{rcl} S_0 & \to & S \mid \varepsilon \\ S & \to & BSB \mid BS \mid SB \mid \mathsf{00} \mid BB \\ B & \to & \mathsf{00} \end{array}$$

- Removing $S_0 \to S$ gives

$$\begin{array}{rcl} S_0 & \to & BSB \mid BS \mid SB \mid \mathsf{00} \mid BB \mid \varepsilon \\ S & \to & BSB \mid BS \mid SB \mid \mathsf{00} \mid BB \\ B & \to & \mathsf{00} \end{array}$$

Then we replaced ill-placed terminals $\mathsf{0}$ by variable U with new rule $U \to \mathsf{0}$, which gives

$$\begin{array}{rcl} S_0 & \to & BSB \mid BS \mid SB \mid UU \mid BB \mid \varepsilon \\ S & \to & BSB \mid BS \mid SB \mid UU \mid BB \\ B & \to & UU \\ U & \to & \mathsf{0} \end{array}$$

Then we shorten rules with a long RHS to a sequence of RHS's with only 2 variables each. So the rule $S_0 \to BSB$ is replaced by the 2 rules $S_0 \to BA_1$ and $A_1 \to SB$. Also the rule $S \to BSB$ is replaced by the 2 rules $S \to BA_2$ and $A_2 \to SB$. Thus, our final CFG in Chomsky normal form is

$$\begin{array}{rcl} S_0 & \to & BA_1 \mid BS \mid SB \mid UU \mid BB \mid \varepsilon \\ S & \to & BA_2 \mid BS \mid SB \mid UU \mid BB \\ B & \to & UU \\ U & \to & \mathsf{0} \\ A_1 & \to & SB \\ A_2 & \to & SB \end{array}$$

To be precise, the CFG in Chomsky normal form is $G = (V, \Sigma, R, S_0)$, where the set of variables is $V = \{S_0, S, B, U, A_1, A_2\}$, the start variable is S_0, the set of terminals is $\Sigma = \{\mathsf{0}\}$, and the rules R are given above.

5. Consider the CFG $G = (V, \Sigma, R, S)$, where $V = \{ S \}$ is the set of variables with S as the starting variable, alphabet $\Sigma = \{ +, -, \times, /, (,), 0, 1, 2, \ldots, 9 \}$, and rules R as

$$S \;\rightarrow\; S + S \mid S - S \mid S \times S \mid S/S \mid (S) \mid -S \mid 0 \mid 1 \mid \cdots \mid 9$$

The CFG G generates the language $L(G)$ of some types of simple arithmetic expressions.

(a) Consider the strings $---5$ and $2+--4$. Give derivations showing that each string belongs to $L(G)$.

Answer: The CFG G derives the string $---5$ as

$$S \Rightarrow -S \Rightarrow --S \Rightarrow ---S \Rightarrow ---5$$

so $---5 \in L(G)$. The CFG G derives the string $2+--4$ as

$$S \Rightarrow S+S \Rightarrow S+-S \Rightarrow S+--S \Rightarrow 2+--S \Rightarrow 2+--4$$

so $2+--4 \in L(G)$.

(b) Suppose that we want to disallow such strings. Give another CFG that achieves this. More specifically, strings such as $2-3$, $2+-3$ and $2--3$ are allowed, but not $2+--3$ nor $2---3$.

Answer: To prevent generating strings such as in part (a), we can use the CFG $G' = (V', \Sigma, R', S)$, where $V' = \{ S, N \}$ is the set of variables with S as the starting variable, alphabet $\Sigma = \{ +, -, \times, /, (,), 0, 1, 2, \ldots, 9 \}$, and rules R' as

$$\begin{aligned} S &\rightarrow S + S \mid S - S \mid S \times S \mid S/S \mid (S) \mid N \mid -N \\ N &\rightarrow 0 \mid 1 \mid \cdots \mid 9 \end{aligned}$$