Lab 7 (Solutions)

1. Create a CFG which produces strings of the form $a^n b^m c^m$. Examples of accepted strings: *abbcc*, *abc*, *aaabbbbcccc*.

The language $L = \{a^n b^m c^m : n, m \geq 0\}$ is the concatentation $L_1 L_2$ of the langauges $L_1 = \{a^n : n \geq 0\}$ and $L_2 = \{b^n c^n : n \geq 0\}$. To construct a context-free grammar generating L, we construct context-free grammars $A \to aA \mid \varepsilon$ generating L_1 and $B \to bBc \mid \varepsilon$ generating L_2, and then form their concatenation using the production $S \to AB$.

$$\begin{aligned} S &\to AB \\ A &\to aA \mid \varepsilon \\ B &\to bBc \mid \varepsilon \end{aligned}$$

2. Make the & operator in the following CFG left-associative and the % operator right-associative

$$S \to a \mid b \mid S \,\&\, S \mid S \,\%\, S \mid \varepsilon$$

We factor the grammar so that $S \to S \,\&\, S$ is left-recursive and $S \to S \,\%\, S$ is right-recursive. Note that there are many solutions to this problem. This solution also gives & lower precedence than %, although your solution does not need to do this.

$$\begin{aligned} S &\to S \,\&\, F \mid F \\ F &\to T \,\%\, F \mid T \\ T &\to a \mid b \mid S \end{aligned}$$

3. Which string is not generated by the following context-free grammar?

$$\begin{aligned} S &\to SS\texttt{*} \mid SS\texttt{+} \mid S\texttt{-} \mid N \\ N &\to \texttt{0} \mid \texttt{1} \end{aligned}$$

(a) `1--0+1+1*`

(b) `1-0*1+-0+-`

(c) `1-0+1-+*1-`

(d) `1-1-*0-+-`

The grammar generates the language of postfix expressions over the alphabet $\Sigma = \{\texttt{0}, \texttt{1}, \texttt{-}, \texttt{+}, \texttt{*}\}$. There are two ways to solve this problem: (a) try to give derivations for each string, and conclude that the string for which you couldn't find a derivation is not generated by the grammar; or (b) find the string which is not a valid postfix expression. For reference, here is a derivation of `1--0+1+1*` in the grammar:

$$\begin{aligned}
S &\Rightarrow SS\texttt{*} && (S \to SS\texttt{*}) \\
&\Rightarrow SS\texttt{+}S\texttt{*} && (S \to SS\texttt{+}) \\
&\Rightarrow SS\texttt{+}S\texttt{+}S\texttt{*} && (S \to SS\texttt{+}) \\
&\Rightarrow S\texttt{-}S\texttt{+}S\texttt{+}S\texttt{*} && (S \to S\texttt{-}) \\
&\Rightarrow S\texttt{--}S\texttt{+}S\texttt{+}S\texttt{*} && (S \to S\texttt{-}) \\
&\Rightarrow N\texttt{--}S\texttt{+}S\texttt{+}S\texttt{*} && (S \to N) \\
&\Rightarrow \texttt{1--}S\texttt{+}S\texttt{+}S\texttt{*} && (N \to \texttt{1}) \\
&\Rightarrow \texttt{1--}N\texttt{+}S\texttt{+}S\texttt{*} && (S \to N) \\
&\Rightarrow \texttt{1--0+}S\texttt{+}S\texttt{*} && (N \to \texttt{0}) \\
&\Rightarrow \texttt{1--0+}N\texttt{+}S\texttt{*} && (S \to N)
\end{aligned}$$

$$\begin{aligned} &\Rightarrow \texttt{1--0+1+}S\texttt{*} && (N \to \texttt{1}) \\ &\Rightarrow \texttt{1--0+1+}N\texttt{*} && (S \to N) \\ &\Rightarrow \texttt{1--0+1+1*} && (N \to \texttt{1}) \end{aligned}$$

4. Which derivation is not a derivation in the following context-free grammar?

$$\begin{aligned} S &\to aS \mid bA \mid \varepsilon \\ A &\to aA \mid bS \mid bB \\ B &\to aB \mid \varepsilon \end{aligned}$$

(a) $S \Rightarrow aS \Rightarrow abA \Rightarrow abbB \Rightarrow abbaB \Rightarrow abba$

(b) $S \Rightarrow bA \Rightarrow baA \Rightarrow babS \Rightarrow babaS \Rightarrow babaaS \Rightarrow babaa$

(c) $S \Rightarrow bA \Rightarrow baA \Rightarrow babS \Rightarrow babbA \Rightarrow babbbB \Rightarrow babbb$

(d) $S \Rightarrow aS \Rightarrow abA \Rightarrow abbA \Rightarrow abbabB \Rightarrow abbab$

This grammar generates the language of strings over the alphabet $\Sigma = \{a, b\}$ having an even number of b's. Like the previous problem, this problem can be solved by trying to give derivations of each string, or reasoning about the language.

To see that our description of the language is correct, we reason as follows: (a) the nonterminal B generates strings matching a^*; (b) the nonterminal A generates sentences of the form a^*bS and a^*ba^*; and (c) the nonterminal S generates sentences of the form a^*bA and ε. Putting this together, we see that $S \Rightarrow^* a^*ba^*bS \mid a^*ba^*ba^* \mid \varepsilon$, which is equivalent to $(a^*ba^*b)^*a^*$. Thus, the grammar generates all strings over Σ having an even number of b's. The only string with an odd number of b's is $abbab$.

Alternatively, note that the nonterminal A can only be produced by generating a single b. The only way to eliminate A from a sentential form is to rewrite using a production that generates another b. This maintains an even number of b's in strings generated by the grammar.

5. Which statement is not true of the following context-free grammar?

$$S \to SS \mid aSb \mid bSa \mid \varepsilon$$

(a) $S \Rightarrow^* aabaabbb$

(b) $S \Rightarrow^* bababbaa$

(c) $S \Rightarrow^* baabbbaa$

(d) $S \Rightarrow^* abbababb$

This grammar generates the language of strings over the alphabet $\Sigma = \{a, b\}$ having the same number of a's and b's. Like the previous problems, this problem can be solved by trying to give derivations of each string, or reasoning about the language. For reference, here is a derivation of $baabbbaa$ in the grammar:

$$\begin{aligned} S &\Rightarrow bSa && (S \to bSa) \\ &\Rightarrow bSSa && (S \to SS) \\ &\Rightarrow baSbSa && (S \to aSb) \\ &\Rightarrow baaSbbSa && (S \to aSb) \\ &\Rightarrow baabbSa && (S \to \varepsilon) \\ &\Rightarrow baabbbSaa && (S \to bSa) \\ &\Rightarrow baabbbaa && (S \to \varepsilon) \end{aligned}$$

Let $|\gamma|_a$ denote the number of a's occurring in a sentential form γ. To see that every string generated by the grammar has the same number of a's and b's, note that each production preserves that invariant:

$$
\begin{array}{lr}
|\varepsilon|_a = |\varepsilon|_b & (S \to \varepsilon) \\
|\alpha|_a = |\alpha|_b \;\to\; |a\alpha b|_a = |a\alpha b|_b & (S \to aSb) \\
|\alpha|_a = |\alpha|_b \;\to\; |b\alpha a|_b = |b\alpha a|_a & (S \to bSa) \\
|\alpha|_a = |\alpha|_b \;\wedge\; |\beta|_a = |\beta|_b \;\to\; |\alpha\beta|_a = |\alpha\beta|_b & (S \to SS)
\end{array}
$$

It then follows by induction on derivations that $S \Rightarrow^* u$ implies that u has the same number of a's and b's. The only string with a different number of a's and b's is $abbababb$, so that must be the answer.

We did not show that the grammar generates every string with the same number of a's and b's. That proof is slightly more difficult, although it makes for a good exercise. However, since the question implies that there is a unique solution, and $abbababb$ is not generated by the grammar, it is be the best choice.

6. Which statement is true of the following context-free grammar?

$$
\begin{array}{lcl}
S & \to & Ab \mid aaB \\
A & \to & a \mid Aa \\
B & \to & bS \mid b
\end{array}
$$

(a) bab has two leftmost derivations
(b) aab has two leftmost derivations
(c) $aaab$ has two leftmost derivations
(d) The grammar is unambiguous

The string aab has two leftmost derivations: $S \Rightarrow aaB \Rightarrow aab$ and $S \Rightarrow Ab \Rightarrow Aab \Rightarrow aab$. Thus, the grammar is ambiguous. You should convince yourself that bab is not generated by the grammar, and that $aaab$ has a unique leftmost derivation.

7. The language generated by the following context-free grammar is inherently ambiguous, in the sense that every context-free grammar generating it is ambiguous. What language is this?

$$
\begin{array}{lcl}
S & \to & XC \mid AY \\
X & \to & aXb \mid aA \mid bB \\
Y & \to & bYc \mid bB \mid cC \\
A & \to & aA \mid \varepsilon \\
B & \to & bB \mid \varepsilon \\
C & \to & cC \mid \varepsilon
\end{array}
$$

(a) $L_1 = \{a^i b^j c^i : i \neq j\}$
(b) $L_2 = \{a^i b^j c^k : i \neq k\}$
(c) $L_3 = \{a^i b^j c^k : i \neq j \;\vee\; j \neq k\}$
(d) $L_4 = \{a^i b^j c^k : i \neq j \;\wedge\; j \neq k\}$

Let $\mathcal{L}(\alpha)$ denote the language of a sentential form α. In other words, $\mathcal{L}(\alpha)$ is the set of strings u such that $\alpha \Rightarrow^* u$. We determine the language L of the grammar in the following way. First, note that

$$
\begin{array}{lcl}
\mathcal{L}(A) & = & \{a^n : n \geq 0\} \\
\mathcal{L}(B) & = & \{b^n : n \geq 0\} \\
\mathcal{L}(C) & = & \{c^n : n \geq 0\}
\end{array}
$$

To determine $\mathcal{L}(X)$, note that there are two distinct ways to terminate derivations beginning with X: either apply $X \to aA$, giving a nonzero number of a's; or $X \to bB$, giving a nonzero number of b's. Thus,

$$X \Rightarrow^n a^n X b^n \Rightarrow^1 a^n aAb^n \Rightarrow^m a^n aa^m b^n = a^{n+m+1}b^n \qquad (n, m \geq 0)$$
$$X \Rightarrow^n a^n X b^n \Rightarrow^1 a^n bBb^n \Rightarrow^m a^n bb^m b^n = a^n b^{n+m+1} \qquad (n, m \geq 0)$$

It follows that every string generated by X has a different number of a's and b's. A similar argument shows that every string generated by Y has a different number of b's and c's. Therefore,

$$\begin{aligned} \mathcal{L}(X) &= \{a^n b^m : n \neq m\} \\ \mathcal{L}(Y) &= \{b^n c^m : n \neq m\} \end{aligned}$$

So the language L of the grammar is given by

$$\begin{aligned} L &= \mathcal{L}(S) \\ &= \mathcal{L}(XC) \cup \mathcal{L}(AY) \\ &= \mathcal{L}(X)\mathcal{L}(C) \cup \mathcal{L}(A)\mathcal{L}(Y) \\ &= \{a^i b^j : i \neq j\}\{c^k : k \geq 0\} \cup \{a^i : i \geq 0\}\{b^j c^k : j \neq k\} \\ &= \{a^i b^j c^k : i \neq j\} \cup \{a^i b^j c^k : j \neq k\} \\ &= \{a^i b^j c^k : i \neq j \vee j \neq k\} \end{aligned}$$

8. The following context-free grammar attempts to resolve the ambiguity in the grammar for conditional expressions by introducing nonterminals for conditionals with and without else branches. Which statement is true of this grammar?

$$\begin{aligned} S &\rightarrow S1 \mid S2 \\ S_1 &\rightarrow \texttt{if } B\ S1 \texttt{ else } S1 \mid \texttt{skip} \\ S_2 &\rightarrow \texttt{if } B\ S \mid \texttt{if } B\ S_1 \texttt{ else } S_2 \\ B &\rightarrow \texttt{true} \mid \texttt{false} \end{aligned}$$

(a) There is no derivation of `if true skip else if true skip` in the grammar

(b) The grammar is unambiguous

(c) There are two rightmost derivations of `if true if true skip else skip`

(d) The `else` is associated with the outer `if` in `if true if true skip else skip`

There is a single rightmost derivation of `if true if true skip else skip`, since the grammar is unambiguous. For reference, here is a derivation of `if true skip else if true skip` in the grammar:

$$\begin{aligned} S &\Rightarrow S_2 && (S \rightarrow S_2) \\ &\Rightarrow \texttt{if } B\ S_1 \texttt{ else } S_2 && (S \rightarrow \texttt{if } B\ S_1 \texttt{ else } S_2) \\ &\Rightarrow \texttt{if true } S_1 \texttt{ else } S_2 && (B \rightarrow \texttt{true}) \\ &\Rightarrow \texttt{if true skip else } S_2 && (S_1 \rightarrow \texttt{skip}) \\ &\Rightarrow \texttt{if true skip else if } B\ S && (S_2 \rightarrow \texttt{if } B\ S) \\ &\Rightarrow \texttt{if true skip else if true } S && (B \rightarrow \texttt{true}) \\ &\Rightarrow \texttt{if true skip else if true } S_1 && (S \rightarrow S_1) \\ &\Rightarrow \texttt{if true skip else if true skip} && (S_1 \rightarrow \texttt{skip}) \end{aligned}$$

To see that the `else` is associated with the inner `if` in `if true if true skip else skip`, we give a leftmost derivation and consider the resulting parse tree.

$$
\begin{array}{lll}
S & \Rightarrow S_2 & (S \to S_2) \\
& \Rightarrow \texttt{if}\ B\ S & (S_2 \to \texttt{if}\ B\ S) \\
& \Rightarrow \texttt{if true}\ S & (B \to \texttt{true}) \\
& \Rightarrow \texttt{if true}\ S_1 & (S \to S_1) \\
& \Rightarrow \texttt{if true if}\ B\ S_1\ \texttt{else}\ S_1 & (S_1 \to \texttt{if}\ B\ S_1\ \texttt{else}\ S_1) \\
& \Rightarrow \texttt{if true if true}\ S_1\ \texttt{else}\ S_1 & (S \to \texttt{true}) \\
& \Rightarrow \texttt{if true if true skip else}\ S_1 & (S_1 \to \texttt{skip}) \\
& \Rightarrow \texttt{if true if true skip else skip} & (S_1 \to \texttt{skip})
\end{array}
$$

Since `if true skip else skip` is derived from `if true` S_1 by applying $S_1 \to$ `if` B S_1 `else` S_1, the `else` is associated with the inner `if`. Note that this is the only way that the `else` can be associated with an `if`, since the grammar is unambiguous.

9. The following context-free grammar generates the language of regular expressions over the alphabet $\{a, b\}$. The union of two regular expressions u and v is written $u + v$ rather than $u \mid v$. Which of the following statements are true of this grammar?

$$
\begin{array}{lcl}
A & \to & A + B \mid B \\
B & \to & CB \mid C \\
C & \to & C^* \mid D \\
D & \to & a \mid b \mid (A)
\end{array}
$$

(a) There are two leftmost derivations of $a^{**}b^{**}$

(b) The Kleene star has lower precedence than concatenation

(c) Concatenation is right-associative

(d) Union has lower precedence than concatenation and the Kleene star

Both (c) and (d) are correct (the original prompt used the singular form of the copula, which was incorrect). The grammar is unambiguous, so $a^{**}b^{**}$ has a single leftmost (and rightmost) derivation.

Union has lower precedence than concatenation, which has lower precedence than the Kleene star. In other words, the Kleene star binds more tightly than concatenation, which binds more tightly than union.

Union is left-associative, while concatenation is right-associative. If you have trouble determining any of this information from the grammar, please review the slides on context-free grammars and parsing.