with yield $x a_1 a_2 \cdots a_m y = w$. By induction, we therefore claim that the result is true for all sentential forms.

In a similar vein, we can show that every partial derivation tree represents some sentential form. We will leave this as an exercise.

Since a derivation tree is also a partial derivation tree whose leaves are terminals, it follows that every sentence in $L(G)$ is the yield of some derivation tree of $G$ and that the yield of every derivation tree is in $L(G)$.

Derivation trees show which productions are used in obtaining a sentence, but do not give the order of their application. Derivation trees are able to represent any derivation, reflecting the fact that this order is irrelevant, an observation that allows us to close a gap in the preceding discussion. By definition, any $w \in L(G)$ has a derivation, but we have not claimed that it also had a leftmost or rightmost derivation. However, once we have a derivation tree, we can always get a leftmost derivation by thinking of the tree as having been built in such a way that the leftmost variable in the tree was always expanded first. Filling in a few details, we are led to the not surprising result that any $w \in L(G)$ has a leftmost and a rightmost derivation (for details, see Exercise 25 at the end of this section).

**Exercises**

1. Complete the arguments in Example 5.2, showing that the language given is generated by the grammar.
2. Draw the derivation tree corresponding to the derivation in Example 5.1.
3. Give a derivation tree for $w = abbbaabbaa$ for the grammar in Example 5.2. Use the derivation tree to find a leftmost derivation.
4. Show that the grammar in Example 5.4 does in fact generate the language described in Equation 5.1.
5. Is the language in Example 5.2 regular?
6. Complete the proof in Theorem 5.1 by showing that the yield of every partial derivation tree with root $S$ is a sentential form of $G$.
7. Find context-free grammars for the following languages (with $n \geq 0, m \geq 0$).
   
   (a) $L = \{a^n b^m : n \leq m + 3\}$.
   (b) $L = \{a^n b^m : n \neq m - 1\}$.
   (c) $L = \{a^n b^m : n \neq 2m\}$.
   (d) $L = \{a^n b^m : 2n \leq m \leq 3n\}$.
   (e) $L = \{w \in \{a, b\}^* : n_a(w) \neq n_b(w)\}$.
   (f) $L = \{w \in \{a, b\}^* : n_a(v) \geq n_b(v) , \text{where } v \text{ is any prefix of } w\}$.
Chapter 5  CONTEXT-FREE LANGUAGES

(g) \( L = \{ w \in \{a,b\}^* : n_a \cdot (w) = 2n_b \cdot (w) + 1 \} \).

8. Find context-free grammars for the following languages (with \( n \geq 0, m \geq 0, k \geq 0 \)).

(a) \( L = \{ a^n b^m c^k : n = m \text{ or } m \leq k \} \).
(b) \( L = \{ a^n b^m c^k : n = m \text{ or } m \neq k \} \).
(c) \( L = \{ a^n b^m c^k : k = n + m \} \).
(d) \( L = \{ a^n b^m c^k : n + 2m = k \} \).
(e) \( L = \{ a^n b^m c^k : k = |n - m| \} \).
(f) \( L = \{ w \in \{a,b,c\}^* : n_a \cdot (w) + n_b \cdot (w) \neq n_c \cdot (w) \} \).
(g) \( L = \{ a^n b^m c^k, k \neq n + m \} \).
(h) \( L = \{ a^n b^m c^k : k \geq 3 \} \).

9. Show that \( L = \{ w \in \{a,b,c\}^* : |w| = 3n_a(w) \} \) is a context-free language.

10. Find a context-free grammar for \( \text{head}(L) \), where \( L \) is the language in Exercise 7(a) above. For the definition of \( \text{head} \) see Exercise 18, Section 4.1.

11. Find a context-free grammar for \( \Sigma = \{a,b\} \) for the language \( L = \{ a^n w w^R b^n : w \in \Sigma^*, n \geq 1 \} \).

\*12. Given a context-free grammar \( G \) for a language \( L \), show how one can create from \( G \) a grammar \( \tilde{G} \) so that \( L(\tilde{G}) = \text{head}(L) \).

13. Let \( L = \{ a^n b^n : n \geq 0 \} \).

(a) Show that \( L^2 \) is context-free.
(b) Show that \( L^k \) is context-free for any given \( k \geq 1 \).
(c) Show that \( L \) and \( L^* \) are context-free.

14. Let \( L_1 \) be the language in Exercise 8(a) and \( L_2 \) the language in Exercise 8(d). Show that \( L_1 \cup L_2 \) is a context-free language.

15. Show that the following language is context-free.
\[
L = \{ u v w v^R : u, v, w \in \{a,b\}^+, |v| = |w| = 2 \}.
\]

\*16. Show that the complement of the language in Example 5.1 is context-free.

17. Show that the complement of the language in Exercise 8(c) is context-free.

18. Show that the language \( L = \{ w_1 c w_2 : w_1, w_2 \in \{a,b\}^+, w_1 \neq w_2^R \} \), with \( \Sigma = \{a,b,c\} \), is context-free.

19. Show a derivation tree for the string \( aabbb \) with the grammar
\[
S \rightarrow AB|\lambda,
A \rightarrow aB,
B \rightarrow Sb.
\]

Give a verbal description of the language generated by this grammar.
4. Show that every s-grammar is unambiguous.

5. Let \( G = (V, T, S, P) \) be an s-grammar. Give an expression for the maximum size of \( P \) in terms of \( |V| \) and \( |T| \).

6. Show that the following grammar is ambiguous.
   \[
   S \rightarrow AB | aaB, \\
   A \rightarrow a | Aa, \\
   B \rightarrow b.
   \]


8. Give the derivation tree for \(((a + b) \ast c)) + a + b\), using the grammar in Example 5.12.

9. Show that a regular language cannot be inherently ambiguous.

10. Give an unambiguous grammar that generates the set of all regular expressions on \( \Sigma = \{a, b\} \).

11. Is it possible for a regular grammar to be ambiguous?

12. Show that the language \( L = \{ww^R : w \in \{a, b\}^*\} \) is not inherently ambiguous.

13. Show that the following grammar is ambiguous.
   \[
   S \rightarrow aSbS | bSaS | \lambda.
   \]

14. Show that the grammar in Example 5.4 is ambiguous, but that the language denoted by it is not.

15. Show that the grammar in Example 1.13 is ambiguous.

16. Show that the grammar in Example 5.5 is unambiguous.

17. Use the exhaustive search parsing method to parse the string \( aabbbaabb \) with the grammar in Example 5.5. In general, how many rounds will be needed to parse any string \( w \) in this language?

18. Is the string \( aabbababb \) in the language generated by the grammar \( S \rightarrow aSS | b \)?

19. Show that the grammar in Example 1.14 is unambiguous.

20. Prove the following result. Let \( G = (V, T, S, P) \) be a context-free grammar in which every \( A \in V \) occurs on the left side of at most one production. Then \( G \) is unambiguous.

21. Find a grammar equivalent to that in Example 5.5 that satisfies the conditions of Theorem 5.2.
The previous examples are fairly easy ones, so rigorous arguments may seem superfluous. But often it is not so easy to find a grammar for a language described in an informal way or to give an intuitive characterization of the language defined by a grammar. To show that a given language is indeed generated by a certain grammar \( G \), we must be able to show (a) that every \( w \in L \) can be derived from \( S \) using \( G \) and (b) that every string so derived is in \( L \).

Take \( \Sigma = \{a, b\} \), and let \( n_a(w) \) and \( n_b(w) \) denote the number of a's and b's in the string \( w \), respectively. Then the grammar \( G \) with productions

\[
\begin{align*}
S & \rightarrow SS, \\
S & \rightarrow \lambda, \\
S & \rightarrow aSb, \\
S & \rightarrow bSa
\end{align*}
\]

generates the language

\[ L = \{w : n_a(w) = n_b(w)\} . \]

This claim is not so obvious, and we need to provide convincing arguments.

First, it is clear that every sentential form of \( G \) has an equal number of a's and b's, since the only productions that generate an \( a \), namely \( S \rightarrow aSb \) and \( S \rightarrow bSa \), simultaneously generate a \( b \). Therefore, every element of \( L(G) \) is in \( L \). It is a little harder to see that every string in \( L \) can be derived with \( G \).

Let us begin by looking at the problem in outline, considering the various forms \( w \in L \) can have. Suppose \( w \) starts with a and ends with b. Then it has the form

\[ w = aw_1b, \]

where \( w_1 \) is also in \( L \). We can think of this case as being derived starting with

\[ S \rightarrow aSb \]

if \( S \) does indeed derive any string in \( L \). A similar argument can be made if \( w \) starts with b and ends with a. But this does not take care of all cases, since a string in \( L \) can begin and end with the same symbol. If we write down a string of this type, say \( aabbba \), we see that it can be considered as the concatenation of two shorter strings \( aabb \) and \( ba \), both of which are in \( L \). Is this true in general? To show that this is indeed so, we can use the following argument: Suppose that, starting at the left end of the string, we count +1 for an \( a \) and -1 for a \( b \). If a string \( w \) starts and ends with \( a \), then the count will be +1 after the leftmost symbol and -1 immediately before the rightmost one. Therefore, the count has to go through zero somewhere in the middle of the string, indicating that such a string must have the form

\[ w = w_1w_2, \]