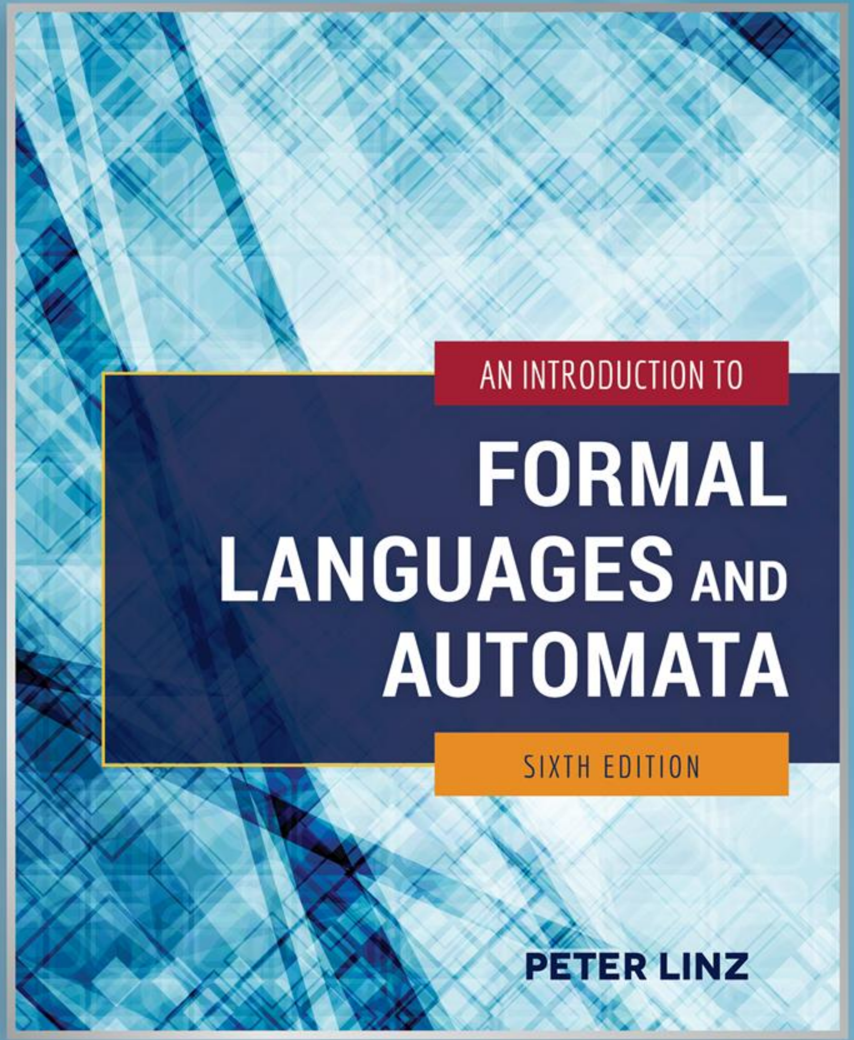
Chapter 11

A HIERARCHY OF FORMAL LANGUAGES AND AUTOMATA



Learning Objectives

At the conclusion of the chapter, the student will be able to:

- Explain the difference between recursive and recursively enumerable languages
- Describe the type of productions in an unrestricted grammar
- Identify the types of languages generated by unrestricted grammars
- Describe the type of productions in a context sensitive grammar
- Give a sequence of derivations to generate a string using the productions in a context sensitive grammar
- Identify the types of languages generated by context-sensitive grammars
- Construct a context-sensitive grammar to generate a particular language
- Describe the structure and components of the Chomsky hierarchy

Recursive and Recursively Enumerable Languages

- A language L is *recursively enumerable* if there exists a Turing machine that accepts it (as we have previously stated, rejected strings cause the machine to either not halt or halt in a nonfinal state)
- A language L is *recursive* if there exists a Turing machine that accepts it and is guaranteed to halt on every valid input string
- In other words, a language is recursive if and only if there exists a membership algorithm for it

Languages That Are Not Recursively Enumerable

- Theorem 11.1 states that, for any nonempty alphabet, there exist languages not recursively enumerable
- One proof involves a technique called diagonalization, which can be used to show that, in a sense, there are fewer Turing Machines than there are languages
- More explicitly, Theorem 11.3 describes the existence of a recursively enumerable language whose complement is not recursively enumerable
- Furthermore, Theorem 11.5 concludes that the family of recursive languages is a proper subset of the family of recursively enumerable languages

Unrestricted Grammars

- An *unrestricted grammar* has essentially no restrictions on the form of its productions:
  - Any variables and terminals on the left side, in any order
  - Any variables and terminals on the right side, in any order
  - The only restriction is that λ is not allowed as the left side of a production
- A sample unrestricted grammar has productions

$$
\begin{aligned}
S &\to S_1B \\
S_1 &\to aS_1b \\
bB &\to bbbB \\
aS_1b &\to aa \\
B &\to \lambda
\end{aligned}
$$

Unrestricted Grammars and Recursively Enumerable Languages

- Theorem 11.6: Any language generated by an unrestricted grammar is recursively enumerable
- Theorem 11.7: For every recursively enumerable language L, there exists an unrestricted grammar G that generates L
- These two theorems establish the result that unrestricted grammars generate exactly the family of recursively enumerable languages, the largest family of languages that can be generated or recognized algorithmically

Context-Sensitive Grammars

- In a context-sensitive grammar, the only restriction is that, for any production, length of the right side is at least as large as the length of the left side
- Example 11.2 introduces a sample unrestricted grammar with productions

$S \rightarrow abc \mid aAbc$

$Ab \rightarrow bA$

$Ac \rightarrow Bbcc$

$bB \rightarrow Bb$

$aB \rightarrow aa \mid aaA$

Characteristics of Context-Sensitive Grammars

- An important characteristic of context-sensitive grammars is that they are **noncontracting**, in the sense that in any derivation, the length of successive sentential forms can never decrease
- These grammars are called context-sensitive because it is possible to specify that variables may only be replaced in certain contexts
- For instance, in the grammar of Example 11.2, variable A can only be replaced if it is followed by either b or c

Context-Sensitive Languages

- A *language L is context-sensitive* if there is a context-sensitive grammar G, such that either L = L(G) or L = L(G) $\cup$ { λ }
- The empty string is included, because by definition, a context-sensitive grammar can never generate a language containing the empty string
- As a result, it can be concluded that the family of context-free languages is a subset of the family of context-sensitive languages
- The language $\{ a^n b^n c^n : n \geq 1 \}$ is context-sensitive, since it is generated by the grammar in Example 11.2

Derivation of Strings Using a Context-Sensitive Grammar

- Using the grammar in Example 11.2, we derive the string aabbcc

$$
\begin{aligned}
S &\Rightarrow aAbc \\
&\Rightarrow abAc \\
&\Rightarrow abBbcc \\
&\Rightarrow aBbbcc \\
&\Rightarrow aabbcc
\end{aligned}
$$

- The variables A and B are effectively used as messengers:
  - an A is created on the left, travels to the right of the first c, where it creates another b and c, as well as variable B
  - the newly created B is sent to the left in order to create the corresponding a

Context-Sensitive Languages and Linear Bounded Automata

- Theorem 11.8 states that, for every context-sensitive language L not including λ, there is a linear bounded automaton that recognizes L
- Theorem 11.9 states that, if a language L is accepted by a linear bounded automaton M, then there is a context-sensitive grammar that generates L
- These two theorems establish the result that context-sensitive grammars generate exactly the family of languages accepted by linear bounded automata, the context-sensitive languages

Relationship Between Recursive and Context-Sensitive Languages

- Theorem 11.10 states that every context-sensitive language is recursive
- Theorem 11.11 maintains that some recursive languages are not context-sensitive
- These two theorems help establish a hierarchical relationship among the various classes of automata and languages:
  - Linear bounded automata are less powerful than Turing machines
  - Linear bounded automata are more powerful than pushdown automata

The Chomsky Hierarchy

- The linguist Noam Chomsky summarized the relationship between language families by classifying them into four language types, type 0 to type 3
- This classification, which became known as the *Chomsky Hierarchy*, is illustrated in Figure 11.3

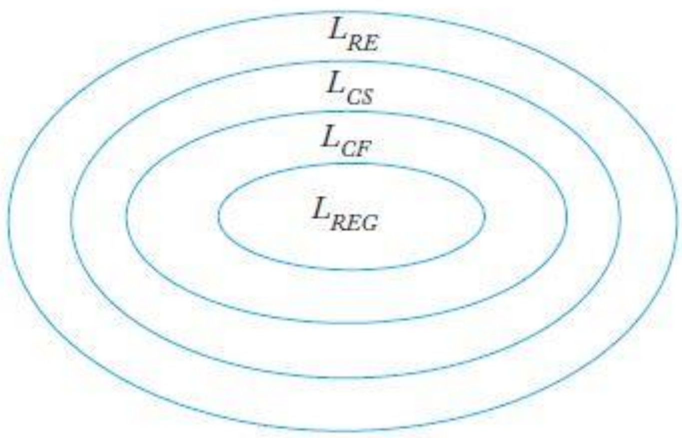


FIGURE 11.3

An Extended Hierarchy

- We have studied additional language families and their relationships to those in the Chomsky Hierarchy
- By including deterministic context-free languages and recursive languages, we obtain the extended hierarchy in Figure 11.4

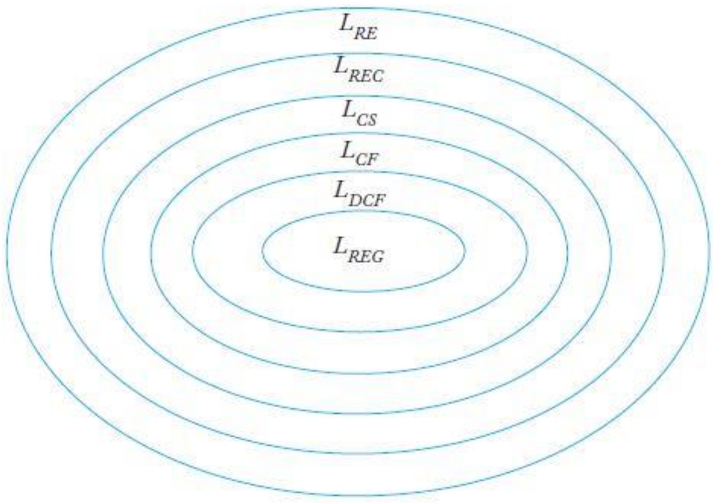


FIGURE 11.4

A Closer Look at the Family of Context-Free Languages

Figure 11.5 illustrates the relationships among various subsets of the family of context-free languages: regular (L_{REG}), linear (L_{LIN}), deterministic context-free (L_{DCF}), and nondeterministic context-free (L_{CF})

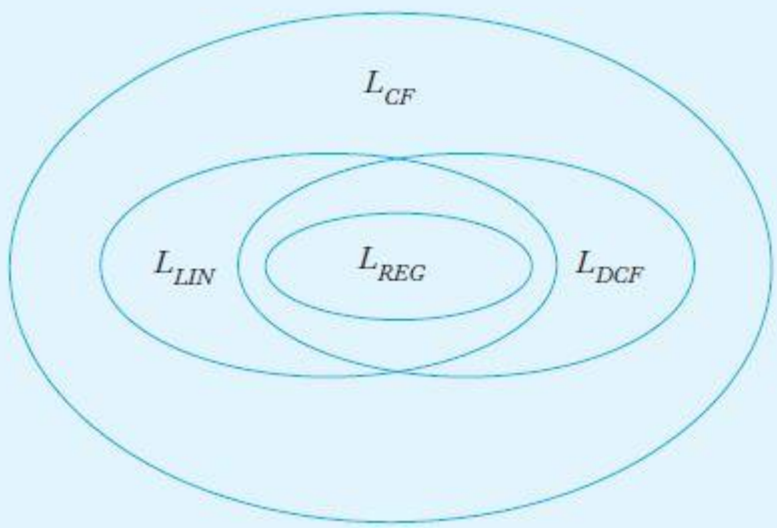


FIGURE 11.5