Sandip foundation's
Sandip Institute of Technology & Research Center, Nashik.
DEPARTMENT OF COMPUTER ENGINEERING
Beyond Curriculum
Subject : Discrete Mathematics (210241)
Year : Second Year **Semester: I**

Modeling Computation

Discrete Structure subject helps in representing languages.

Languages and Grammars

Introduction

Words in the English language can be combined in various ways. The grammar of English tells us whether a combination of words is a valid sentence. For instance, the frog writes neatly is a valid sentence, because it is formed from a noun phrase, the frog, made up of the article the and the noun frog, followed by a verb phrase, writes neatly, made up of the verb writes and the adverb neatly. We do not care that this is a nonsensical statement, because we are concerned only with the syntax, or form, of the sentence, and not its semantics, or meaning. We also note that the combination of words swims quickly mathematics is not a valid sentence because it does not follow the rules of English grammar. The syntax of a natural language, that is, a spoken language, such as English, French, German, or Spanish, is extremely complicated. In fact, it does not seem possible to specify all the rules of syntax for a natural language. Research in the automatic translation of one language to another has led to the concept of a formal language, which, unlike a natural language, is specified by a well-defined set of rules of syntax. Rules of syntax are important not only in linguistics, the study of natural languages, but also in the study of programming languages. We will describe the sentences of a formal language using a grammar. The use of grammars helps when we consider the two classes of problems that arise most frequently in applications to programming languages: (1) How can we determine whether a combination of words is a valid sentence in a formal language? (2) How can we generate the valid sentences of a formal language? Before giving a technical definition of a grammar, we will describe an example of a grammar that generates a subset of English. This subset of English is defined using a list of rules that describe how a valid sentence can be produced. We specify that 1. a sentence is made up of a noun phrase followed by a verb phrase;

2. a noun phrase is made up of an article followed by an adjective followed by a noun, or

3. a noun phrase is made up of an article followed by a noun;

4. a verb phrase is made up of a verb followed by an adverb, or

5. a verb phrase is made up of a verb;

6. an article is a, or

7. an article is the;

8. an adjective is large, or

9. an adjective is hungry;

10. a noun is rabbit, or

11. a noun is mathematician;

12. a verb is eats, or

13. a verb is hops;

14. an adverb is quickly, or

15. an adverb is wildly.

From these rules we can form valid sentences using a series of replacements until no more rules can be used. For instance, we can follow the sequence of replacements:

sentence

noun phrase verb phrase

article adjective noun verb phrase

article adjective noun verb adverb

the adjective noun verb adverb

the large noun verb adverb

the large rabbit verb adverb

the large rabbit hops adverb

the large rabbit hops quickly

to obtain a valid sentence. It is also easy to see that some other valid sentences are: a hungry mathematician eats wildly, a large mathematician hops, the rabbit eats quickly, and so on. Also, we can see that the quickly eats mathematician is not a valid sentence. Phrase-Structure Grammars

Before we give a formal definition of a grammar, we introduce a little terminology.

Languages can be specified in various ways. One way is to list all the words in the language. Another is to give some criteria that a word

must satisfy to be in the language. In this section, we describe another important way to specify a language, namely, through the use of a grammar, such as the set of rules we gave in the introduction to this section. A grammar provides a set of symbols of various types and a set of rules for producing words. More precisely, a grammar has a vocabulary V , which is a set of symbols used to derive members of the language. Some of the elements of the vocabulary cannot be replaced by other symbols. These are called terminals, and the other members of the vocabulary, which can be replaced by other symbols, are called nonterminals. The sets of terminals and nonterminals are usually denoted by T and N , respectively. In the example given in the introduction of the section, the set of terminals is {a, the, rabbit, mathematician, hops, eats, quickly, wildly}, and the set of nonterminals is {sentence, noun phrase, verb phrase, adjective, article, noun, verb, adverb}. There is a special member of the vocabulary called the start symbol, denoted by S, which is the element of the vocabulary that we always begin with. In the example in the introduction, the start symbol is sentence. The rules that specify when we can replace a string from V^* , the set of all strings of elements in the vocabulary, with another string are called the productions of the grammar. We denote by $z_0 \rightarrow z_1$ the production that specifies that z_0 can be replaced by z_1 within a string. The productions in the grammar given in the introduction of this section were listed. The first production, written using this notation, is sentence → noun phrase verb phrase. We summarize this terminology in Definition 2. A phrase-structure grammar G = (V , T , S, P) consists of a vocabulary V , a subset T of V consisting of terminal symbols, a start symbol S from V , and a finite set of productions P . The set V − T is denoted by N. Elements of N are called nonterminal symbols.

Every production in P must contain at least one nonterminal on its left side.Let G = (V , T , S, P), where V = {a, b, A, B, S}, T = {a, b}, S is the start symbol, and P ={S → ABa, A → BB, B → ab, AB → b}. G is an example of a phrase-structure grammar.

EXAMPLE 1

We will be interested in the words that can be generated by the productions of a phrase structure grammar.

DEFINITION 3

Let G = (V , T , S, P) be a phrase-structure grammar. Let $w_0 = lz_0r$ (that is, the concatenation of l, z_0 , and r) and $w_1 = lz_1r$ be strings over V . If $z_0 \rightarrow z_1$ is a production of G, We say that w_1 is directly derivable from w_0 and we write $w_0 \Rightarrow w_1$. If $w_0, w_1, \ldots, w_n$ are strings over V such that $w_0 \Rightarrow w_1, w_1 \Rightarrow w_2, \ldots, w_{n-1} \Rightarrow w_n$, then we say that w_n is derivable from w_0 , and we write $w_0 \Rightarrow w_n$. The sequence of steps used to obtain w_n from w_0 is called a derivation. Types of Phrase-Structure Grammars Phrase-structure grammars can be classified according to the types of productions that are allowed. We will describe the classification scheme introduced by Noam Chomsky. We will see that the different types of languages defined in this scheme correspond to the classes of languages that can be recognized using different models of computing machines. A type 0 grammar has no restrictions on its productions. A type 1 grammar can have

productions of the form w 1 → w 2 , where w 1 = lAr and w 2 = lwr, where A is a nonterminal symbol, l and r are strings of zero or more terminal or nonterminal symbols, and w is a nonempty string of terminal or nonterminal symbols. It can also have the production S → λ as long as S does not appear on the right-hand side of any other production. A type 2 grammar can have productions only of the form w 1 → w 2 , where w 1 is a single symbol that is not a terminal symbol. A type 3 grammar can have productions only of the form w 1 → w 2 with w 1 = A and either w 2 = aB or w 2 = a, where A and B are nonterminal symbols and a is a terminal symbol, or with w 1 = S and w 2 = λ. Type 2 grammars are called context-free grammars because a nonterminal symbol that is the left side of a production can be replaced in a string whenever it occurs, no matter what else is in the string. A language generated by a type 2 grammar is called a context-free language. When there is a production of the form lw 1 r → lw 2 r (but not of the form w 1 → w 2), the grammar is called type 1 or context-sensitive because w 1 can be replaced by w 2 only when it is surrounded by the strings l and r. A language generated by a type 1 grammar is called a context-sensitive language. Type 3 grammars are also called regular grammars. A language generated by a regular grammar is called regular. Section 13.4 deals with the relationship between regular languages and finite-state machines. Of the four types of grammars we have defined, context-sensitive grammars have the most complicated definition. Sometimes, these grammars are defined in a different way. A production of the form w 1 → w 2 is called non contracting if the length of w 1 is less than or equal to the Derivation Trees A derivation in the language generated by a context-free grammar can be represented graphically using an ordered rooted tree, called a derivation, or parse tree. The root of this tree represents the starting symbol. The internal vertices of the tree represent the nonterminal symbols that arise in the derivation. The leaves of the tree represent the terminal symbols that arise. If the production A → w arises in the derivation, where w is a word, the vertex that represents A has as children vertices that represent each symbol in w, in order from left to right.