Heaven's light is our guide"

Rajshahi University of Engineering & Technology
Department of Computer Science & Engineering

Network Security

Course No. : 305

Chapter 11: Modeling Computation

Prepared By : Julia Rahman

11.1 Languages and Grammars

11.1 Languages and Grammars

- The ***grammar*** of English tells us whether a combination of words is a valid sentence.
- ***Syntax*** → form of the sentence, or grammar
- ***Semantics*** → meaning.
- Mathematics is not a valid sentence because it does not follow the rules of English grammar.
- The syntax of a **natural *language*** → spoken language, such as English, French, German, or Spanish, is extremely complicated.
- ***Formal language***, which, unlike a natural language, is specified by a well-defined set of rules of syntax.
- 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?

11.1 Languages and Grammars

- **Rules of a grammar that generates a valid English sentence:**

  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, an or the***;

  - 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***

11.1 Languages and Grammars

- **DEFINITION 1:**
  - ✓ A ***vocabulary*** (or alphabet) V is a finite, nonempty set of elements called symbols.
  - ✓ A ***word*** (or sentence) over V is a string of finite length of elements of V.
  - ✓ The ***empty string*** or null string, denoted by λ, is the string containing no symbols.
  - ✓ The set of all words over V is denoted by V*.
  - ✓ A ***language*** over V is a subset of V* .
  - ✓ {λ} is the set containing exactly one string, namely, the empty string ϕ.
- 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*** - denoted by T .
- The members of the vocabulary, which can be replaced by other symbols, are called ***nonterminals*** - denoted by N.
- The set of terminals is {a, the, rabbit, mathematician, hops, eats, quickly, wildly}.
- The set of nonterminals is {sentence, noun phrase, verb phrase, adjective, article, noun, verb, adverb} .

11.1 Languages and Grammars

- 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.
- **DEFINITION 2:**

  A ***phrase-structure grammar*** G = (V, T, S, P) consists of a vocabulary V, a subset T of V consisting of terminal elements, 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.
- **EXAMPLE 1:** 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.
- **DEFINITION 3:**

  Let G = (V, T, S, P) be a phrase-structure grammar. Let W_0 = lz_0r 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 => W_1 . If $W_0, W_1, \ldots, W_n$ are strings over V such that W_0 => W_1 , W_1 => W_2, . . , W_{n-1} => W_n, then we say that W_n is derivable from W_0, and we write W_0 => W_n. The sequence of steps used to obtain W_n from W_0 is called a derivation.

11.1 Languages and Grammars

- **EXAMPLE 2:** The string Aaba is directly derivable from ABa in the grammar in Example 1 because B → ab is a production in the grammar. The string abababa is derivable from ABa because ABa => Aaba => BBaba => Bababa => abababa, using the productions B → ab, A → BB, B → ab, and B → ab in succession.
- **DEFINITION 4:**

  Let G = (V, T, S, P) be a phrase-structure grammar. The language generated by G (or the language of G), denoted by L(G), is the set of all strings of terminals that are derivable from the starting state S. In other words,

  $$L(G) = \{w \in T^* \mid S \rightarrow w \}.$$
- **EXAMPLE 3:** Let G be the grammar with vocabulary V = {S , A, a, b} , set of terminals T = {a, b}, starting symbol S, and productions P = { S → a A , S → b, A → aa} . What is L(G), the language of this grammar?

  Solution: From the start state S we can derive aA using the production S → aA. We can also use the production S → b to derive b. From aA the production A →aa can be used to derive aaa . No additional words can be derived. Hence, L(G) = {b, aaa}.

11.1 Languages and Grammars

Types of Phrase-Structure Grammars

1) **Type 0 grammar:**
   - ✓ Has no restrictions on its productions.
   - ✓ Most general type
   - ✓ It is obtained by making simple restriction that y can not be the empty string rewrite form xyz → xwz
2) **Type 1 grammar:**
   - ✓ Can have productions of the form $W_1 \to W_2$, where W_1 = IAr and W_1 = Iwr, where A is a nonterminal symbol, I 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.
   - ✓ Called ***context-sensitive*** because W_1 can be replaced by W_2 only when it is surrounded by the strings I and r .
   - ✓ Rewrite rules for context-sensitive grammars have the following form:

     A X B→A Y B

which means that X must be a single nonterminal symbol and Y a nonempty string.

11.1 Languages and Grammars

2) **Type 2 grammar:**
   - ✓ Can have productions only of the form $W_1 \rightarrow W_2$, where W_1 is a single symbol that is not a terminal symbol.
   - ✓ 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.
   - ✓ General form is <symbol> → <symbol 1>……….<symbol k> where k ≥ 1 and where the left hand side is a single nonterminal symbol A →X Y Z where A is a single nonterminal.

3) **Type 3 grammar:**
   - ✓ Can have productions only of the form $W_1 \rightarrow W_2$ with W_1 = A and either W_2 = aB or W_1 = a, where A and B are nonterminal symbols and a is a terminal symbol, or with W_1 = S and W_2 = A .
   - ✓ Called regular grammars.
   - ✓ A language generated by a regular grammar is called regular.

11.1 Languages and Grammars

- A production of the form $W_1 \rightarrow W_2$ is called ***noncontracting*** if the length of W_1 is less than or equal to the length of W_2 .
- According to characterization of context-senstive languages, every production in type 1 grammar, other than the production S → A, if it is present, is noncontracting.
- The lengths of strings in a derivation in a context-free language are nondecreasing unless the production S → A is used. This means that the only way for the empty string to belong to the language generated by a context-sensitive grammar is for the production S → A to be part of the grammar.
- The other way that context-sensitive grammars are defined is by specifying that all productions are noncontracting. A grammar with this property is called noncontracting or monotonic.

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 nontenninal symbols that arise in the derivation.
- ✓ The leaves of the tree represent the tenninal symbols that arise.

11.1 Languages and Grammars

- **EXAMPLE 11:** Construct a derivation tree for the derivation of the hungry rabbit eats quickly, given in the introduction of this section.
  Solution: The derivation tree is shown in Figure 1 .