CST8152 – Compilers
Article #11
Grammars - Notations

Grammars

Noam Chomsky's Grammar Hierarchy

1. Regular Grammars
2. Context Free Grammars (CFG or BNF)
3. Context Sensitive Grammars
4. General

A Context Free Grammar (CFG) is defined by the following four components:

1. A finite set of terminal symbols (***terminals***) or a final terminal vocabulary $\mathbf{V_t}$. For the lexical grammar the terminals are the alphabet; for the syntactic grammar the terminals are the token set produced by the scanner and defined by the lexical grammar.
2. A finite set of ***nonterminals*** or a nonterminal vocabulary $\mathbf{V_n}$ .Nonterminals are not part of the language. They are intermediate symbols used to define the grammar for the language.
3. A finite set of ***productions*** (rewriting or replacement or substitution or derivation rules) **P**. Productions have the form:

   $A \rightarrow X_1X_2X_3\ldots X_m$
   where $A \in V_n, X_i \in V_n \cup V_t$, $1 \leq i \leq m, m > 0$
   and
   $A \rightarrow \varepsilon$ (empty) $(m = 0)$ is a valid production

4. A **start** (or **goal**) symbol **S**. The start symbol $S \in V_n$ (S belong to V_n) is always the root of the parse tree.

Following the definition above, a CFG is the four-toupe $\mathbf{G = (V_t, V_n, P, S)}$.
L(G) is the language defined or generated by the grammar.

The following notation will be used when discussing grammars.

- a, b, c – a small letter at the beginning of the alphabet will denote a terminal.
  $\{a, b, c, \ldots\} \in V_t$
- A, B, C – a capital letter at the beginning of the alphabet will denote a nonterminal.
  $\{A, B, C, \ldots\} \in V_n$
- X, Y, Z – a capital letter at the end of the alphabet will denote a terminal or a nonterminal.
  $\{\ldots, X, Y, Z\} \in V_t \cup V_n$
- α, β, γ – a small Greek letter at the beginning of the alphabet will denote a string containing a combination of terminals and nonteminals.
  $\{\alpha, \beta, \gamma, \ldots\}$ -> sentential forms of the grammar
- u, v, w – a small letter at the end of the alphabet will denote a string containing only a combination of terminals .
  {… u, v, w } -> sentences of the language defined by the grammar

Example:

Using the notations described above the following grammar defining arithmetic expressions

<expression> -> <expression> + <term> | <expression> - <term> | <term>
<term> -> <term> * <factor> | < term > / < factor > | < factor >
< factor > -> vid | dil | fpl | (<expression>)

will look like

E -> E + T | E – T | T
T -> T * F | T / F | F
F -> i | d | f | (E) or F -> vid | dil | fpl | (E)

or

E -> E α_1| E α_2| β_1
T -> T α_3| T α_4| β_2
F -> i | d | f | (γ