Chapter 2

Context-free Grammars Part 1

Defining a programming language

Want approach that is

- concise
- precise
- sufficiently powerful
- suitable for syntax-checking algorithm

Context-free grammar

Contains four components

1. set of nonterminals
2. set of terminals
3. set of productions
4. start symbol from set of nonterminals

Example of context-free grammar G1

1. N = { S, B, C}
2. T = { b, c}
3. {

   S -> BC,

   B -> bB,

   B -> λ,

   C -> ccc,

}

4. S

Derivations Using G1

S => BC => Bccc => bBccc => bccc

S => BC => C => ccc

To indicate that bccc can be derived from S in zero or more steps, we write

$$S \overset{*}{=>} bccc$$

L(G1): language defined by G1

$L(G1) = \{x : S \overset{*}{=>} x \text{ and } X \in T^*\}$

That is, it is the set of all terminal strings derivable from S.

Some simple grammars

S -> bS S -> λ	b*
S -> Sb S -> λ	b*
S -> BcD B -> bB B -> λ D -> dD D -> λ	b*c d*

Right linear grammar

Productions of form

A -> nonNullTerminalString   nonterminal

A -> terminalString

S -> bbbS

S -> λ

Regular grammar

Productions of the form

A -> terminal nonterminal

A -> terminal

A -> λ

S -> bS

S -> λ

Convert right linear to regular

S -> bbbS right linear

S -> λ

S -> bB regular

B -> bC

C -> bS

S -> λ

non-Context-free languages

Non-repeating list:

int x, y, z;

TRIPLED = $\{b^i c^i d^i : i \geq 0\}$

Regular grammar

Productions of the form

A -> terminal nonterminal

A -> terminal

A -> λ

S -> bS

S -> λ

Regular grammar

Productions of the form

A -> terminal nonterminal

A -> terminal

A -> λ

S -> bS

S -> λ