Context-Free Grammars: Motivation

- Languages in many practical applications are not regular. For example, open and close parentheses.
- Describe arithmetic expressions
- Describe components of programming languages
- Describe natural languages.
Natural Language

<sentence> -&gt; <noun phrase> <verb phrase>
<noun phrase> -&gt; <adjective> <noun phrase>
<noun phrase> -&gt; <noun>
<noun> -&gt; boy|girl
<adjective> -&gt; little
<verb phrase>- &gt; plays
Arithmetic Expressions

<expression> - > <expression> + <expression>
<expression>- > <expression> * <expression>
<expression>- > (<expression>)
<expression>- > <id>
{id> - > <letter><id>
{id> - > <letter>
<letter> - > a|b|c|d
Context-Free Grammar

- Context-Free Grammar $G = (V, T, P, S)$
  - $V$: a finite set of variables (nonterminals)
  - $T$: a finite set of terminals ($V$ and $T$ are disjoint)
  - $P$: a finite set of productions; each production is of the form $A \rightarrow \alpha$, where both $A$ is a variable from $V$ and $\alpha$ is a string from $V + T$.
  - $S$: a special variable called the start symbol.
Example

S -> aB|bA
A -> a|aS|bAA
B -> b|bS|aBB

This context-free grammar describes the language in which all strings consists of equal number of a’s and b’s.
Derivations

• Direct Derivation: \( \Rightarrow_G \)
  
  If \( A \rightarrow \beta \) is a production and \( \alpha \) and \( \gamma \) are any strings in \((V + T)^*\), then \( \alpha A\gamma \Rightarrow_G \alpha\beta\gamma \).

• Derivation: \( \Rightarrow^*_G \)
  
  If \( \alpha_1, \alpha_2, ..., \alpha_m \) are strings in \((V + T)^*\), \( m \geq 1 \), and
  
  \( \alpha_1 \Rightarrow_G \alpha_2, \alpha_2 \Rightarrow_G \alpha_3, ..., \alpha_{m-1} \Rightarrow_G \alpha_m \),
  
  then \( \alpha_1 \Rightarrow^*_G \alpha_m \)
Languages Generated by Grammar

The language generated by Grammar G is:

\[ \{ w | w \text{ in } T^* \text{ and } S \Rightarrow^*_G w \} \]

Namely, a string is in \( L(G) \) if:

1) the string consists solely of terminals

2) the string can be derived from \( S \).
Example

S -> aSbε

Apply S->aSb n times and S-> ε one time, we have
S => aSb => aaSbb => a³Sb³ => … => aⁿSbⁿ => aⁿbⁿ

L(G) = \{aⁿbⁿ | n >= 0\}
Derivation (parse) tree

• A derivation may be displayed as a tree. The vertices of the tree are labeled with terminal or variables and possibly \( \varepsilon \).

• The sons of a vertex labeled \( A \) are labeled \( X_1, \ldots, X_n \), respectively, \( A \rightarrow X_1 \ldots X_n \) must be a production.
Example

S → aSbε
A derivation tree for $a^3b^3$
Leftmost (Rightmost) Derivation

• Leftmost derivation
  If at each step in a derivation a production is applied to the leftmost variable, then the derivation is said to be leftmost.

• Rightmost derivation
  If at each step in a derivation a production is applied to the rightmost variable, then the derivation is said to be rightmost.

• A string $x$ has a unique leftmost derivation and a unique rightmost derivation.
Ambiguity

- A derivation tree enforces a structure on a string.

- A context-free grammar is ambiguous if there exists some string with two different derivation trees.

- A context-free language is inherently ambiguous if every grammar of the language is ambiguous.
Example

S -> S + S | S * S | number
Removing Ambiguity

• It is impossible to remove ambiguity from all grammars.

• Rewrite an ambiguous grammar to add more variables.

• Even if a grammar is ambiguous, but its leftmost (rightmost) derivations are unique.
Inherent Ambiguity

• A context-free language $L$ is said to be inherently ambiguous if all its grammars ambiguous.

• There are context-free languages which are inherently ambiguous.
Regular Grammar

• All regular languages are context-free languages.

• If all the productions have the following forms
  – A->xB
  – A->x
  where A and B are variables and x is a string of terminals, then the language generated by the grammar is regular.