Theory of Computation

Dr Samayveer Singh

Context-Free Languages

Context-Free Languages

- A context-free grammar (CFG) consisting of a finite set of grammar rules is a quadruple (V_N, ∑, P, S) where

  - V_N is a set of non-terminal symbols.
  - ∑ is a set of terminals where $V_N \cap \sum$ = NULL.
  - P is a set of rules, P: $V_N \rightarrow (V_N \cup \sum)^*$, i.e., the left-hand side of the production rule P does have any right context or left context.
  - S is the start symbol.

- Example
  - The grammar ({A}, {a, b, c}, P, A), P : A → aA, A → abc.
  - The grammar ({S, a, b}, {a, b}, P, S), P: S → aSa, S → bSb, S → ε
  - The grammar ({S, F}, {0, 1}, P, S), P: S → 00S | 11F, F → 00F | ε

Context-Free Grammar

- A grammar is context-free if every production is of the form

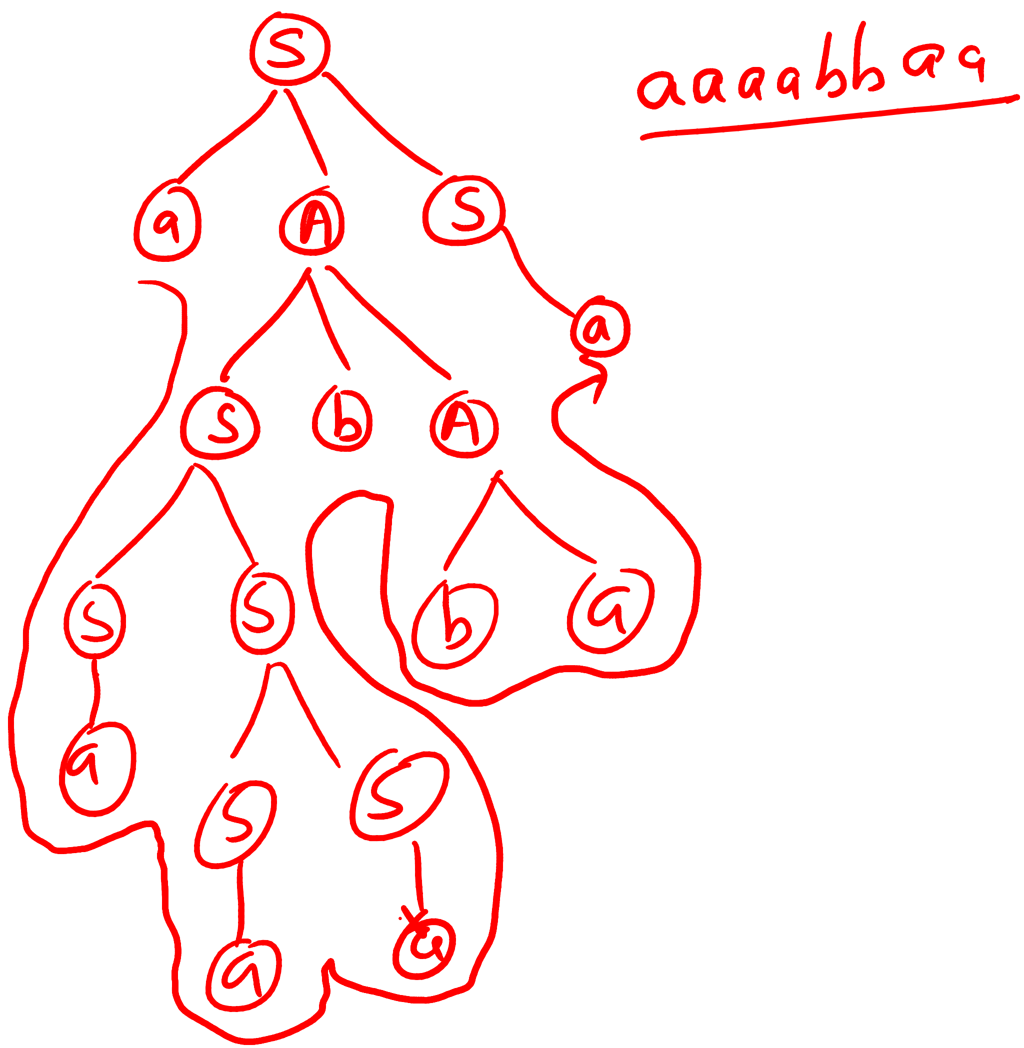
$$A \rightarrow \alpha$$

where $A \mathcal{E} V_N$ *and* $\alpha \mathcal{E} (V_N \; U \sum)^*$.

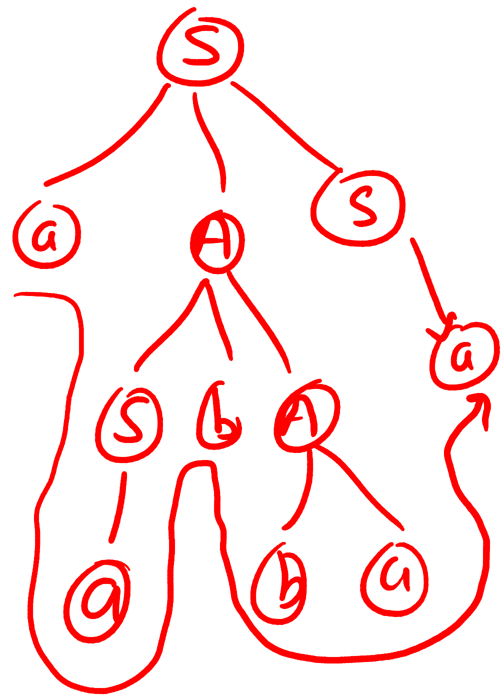
Derivation Trees

- The derivations in a CFG can be represented using trees. Such trees representing derivations are called derivation trees.
- A derivation tree (also called a parse tree) for a CFG G = *(VN,* $\sum$, *P, S) is a tree satisfying the following conditions:*
  - Every vertex has a label which is a variable or terminal or $\wedge$.
  - The root has label S.
  - The label of an internal vertex is a variable.
  - If the vertices $n_1, n_2, ..n_k$ *written with labels* $X_1, X_2, ..., X_k$ *are* the sons of vertex ***n*** with label *A, then* $A \rightarrow X_1, X_2, ..., X_k$ *is a* production in *P*.
  - A vertex ***n*** is a leaf if its label is $a \varepsilon \sum$ *or* $\wedge$; ***n*** *is the only son of* its father if its label is $\wedge$.

For eg: let G = *({S, A}, {a, b}, P, S). where P consists of S →aAS | a | SS, A →SbA | ba. Draw the derivation tree for the G.*



Consider G whose productions are S → *aAS* | *a*, *A* → *SbA* | *SS* | *ba*. *Show that* S ==> *aabbaa and construct a derivation tree whose yield is aabbaa.*



S → aAS
→ aSbAS
→ aabAS
→ aabbaa

Types of derivation trees

- There are two types of derivation namely leftmost and rightmost derivation tree.
  - Leftmost derivation: A derivation is called a *leftmost derivation* if we apply a production only to the leftmost variable at every step.
  - Rightmost derivation: A derivation is a *rightmost derivation* if we apply production to the rightmost variable at every step.

Let G be the grammar S →*0B* |*1A*, *A* →*0* | *0S* |*1AA*, *B* →*1*| *1S* | *0BB*. *For* the string 00110101, ~~find (a)~~ the leftmost derivation, (b) the rightmost derivation, and (c) the derivation tree.

(a)

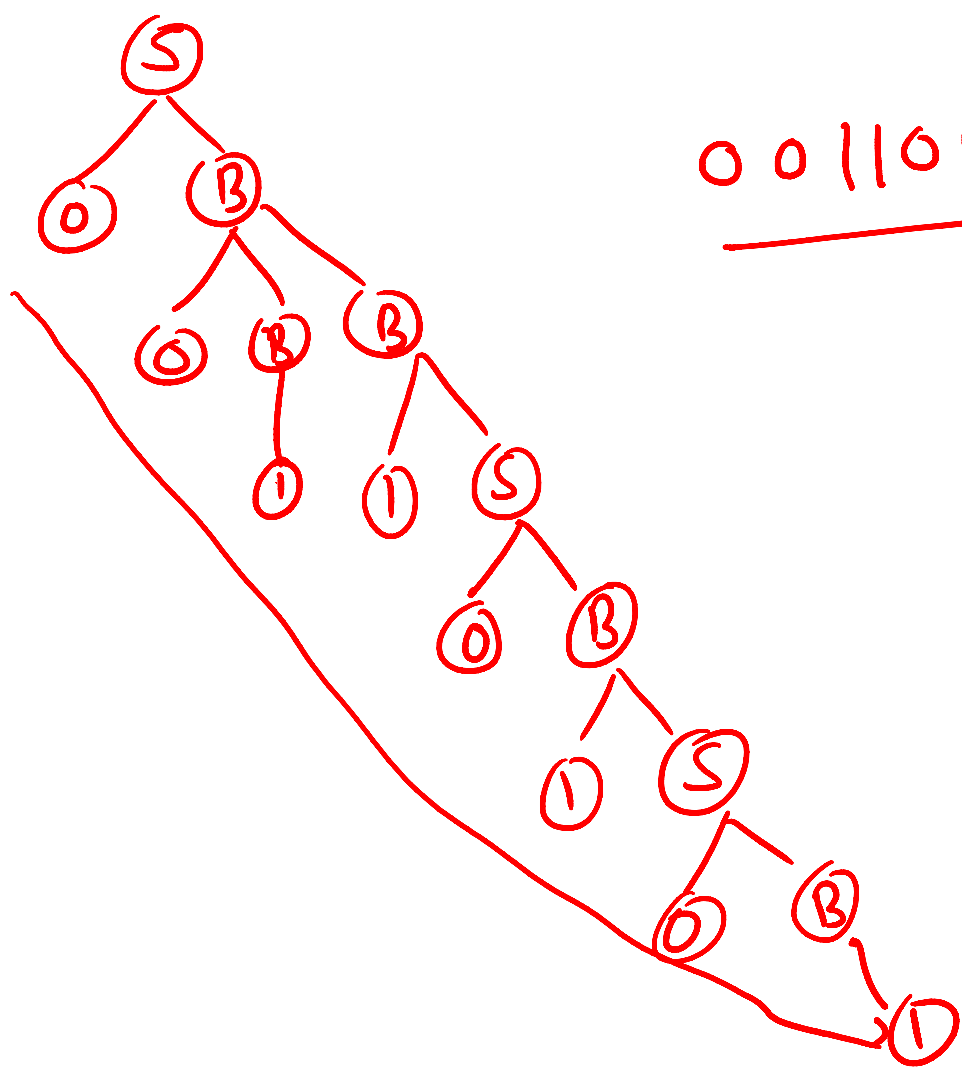
S → 0B
→ 00BB
→ 001B
→ 0011S
→ 00110B
→ 001101S
→ 0011010B
→ 00110101

(b)

S → 0B
→ 00BB → 00B1
→ 00B1S → 001S1
→ 00B10B → 0011A1
→ 00B101S → 00110S1
→ 00B1010B → 001101A1
→ 00B10101 → 00110101
→ 00110101

Let G be the grammar S →*OB* |*1A*, *A* → *0* | *0S* |*1AA*, *B* →*1*| *1S* | *OBB*. *For* the string 00110101, find (a) the leftmost derivation, (b) the rightmost derivation, and (c) the derivation tree.

(c)



00110101

Ambiguity in Context-free Grammars

- A terminal string *W Ɛ L(G) is ambiguous if there exist two* or more derivation trees for *w (or there exist two or more leftmost derivations* of *w).*

Consider, for example, G = *({S}, {a, b, +, *}, P. S), where P consists* of S $\rightarrow$ S + S | S * S | *a* | *b. Can we draw two derivation trees for string a + a * b.*

(a)

$S \rightarrow S+S$

$\rightarrow a+S$

$\rightarrow a+S*S$

$\rightarrow a+a*S$

$\rightarrow a+a*b$

(b)

$S \rightarrow S*S$

$\rightarrow S+S*S$

$\rightarrow a+S*S$

$\rightarrow a+a*S$

$\rightarrow a+a*b$

If G is the grammar S →*SbS* | *a, show that G is ambiguous.*