Theory of Machines and Languages

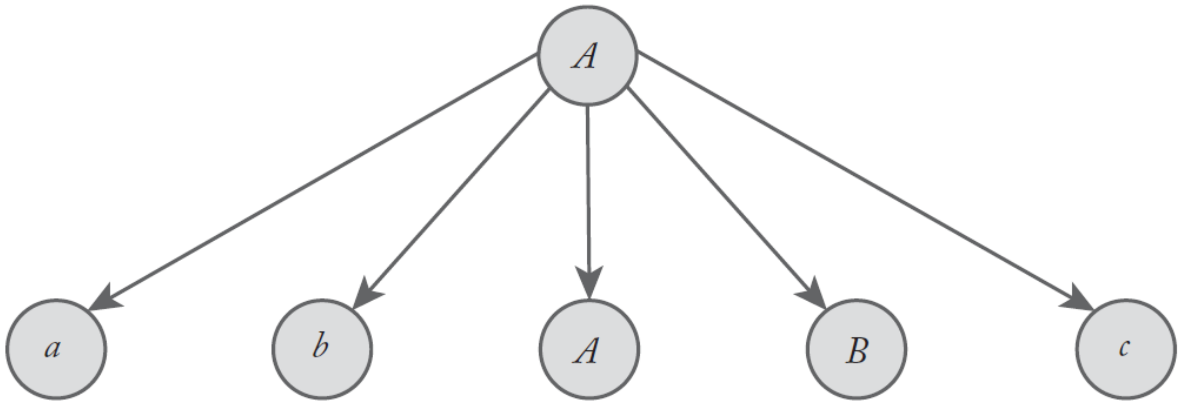
Fatemeh Deldar

1403-1404

Derivation Trees

- A second way of showing derivations, **independent of the order in which productions are used**, is by a derivation or parse tree

- **Example**
  - A part of a derivation tree representing the production $A \rightarrow abABc$



Derivation Trees

- Let $G = (V, T, S, P)$ be a context-free grammar. An ordered tree is a derivation tree for G if and only if it has the following properties.

  1. The root is labeled S.

  2. Every leaf has a label from $T \cup \{\lambda\}$.

  3. Every interior vertex (a vertex that is not a leaf) has a label from V.

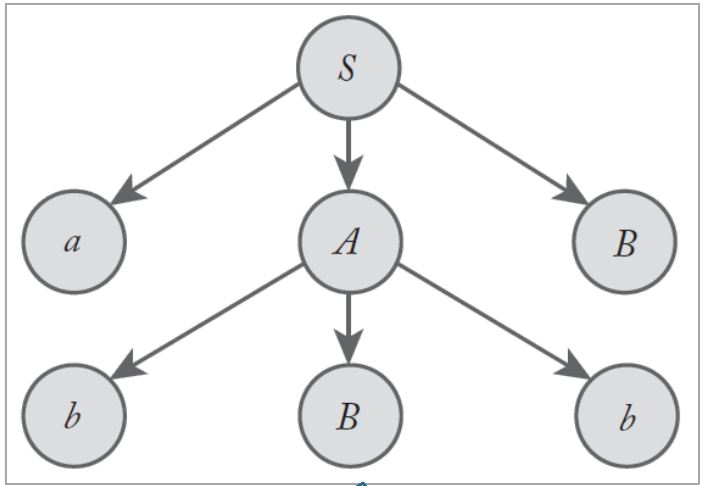
  4. If a vertex has label $A \in V$, and its children are labeled (from left to right) $a_1, a_2, \ldots, a_n$, then P must contain a production of the form

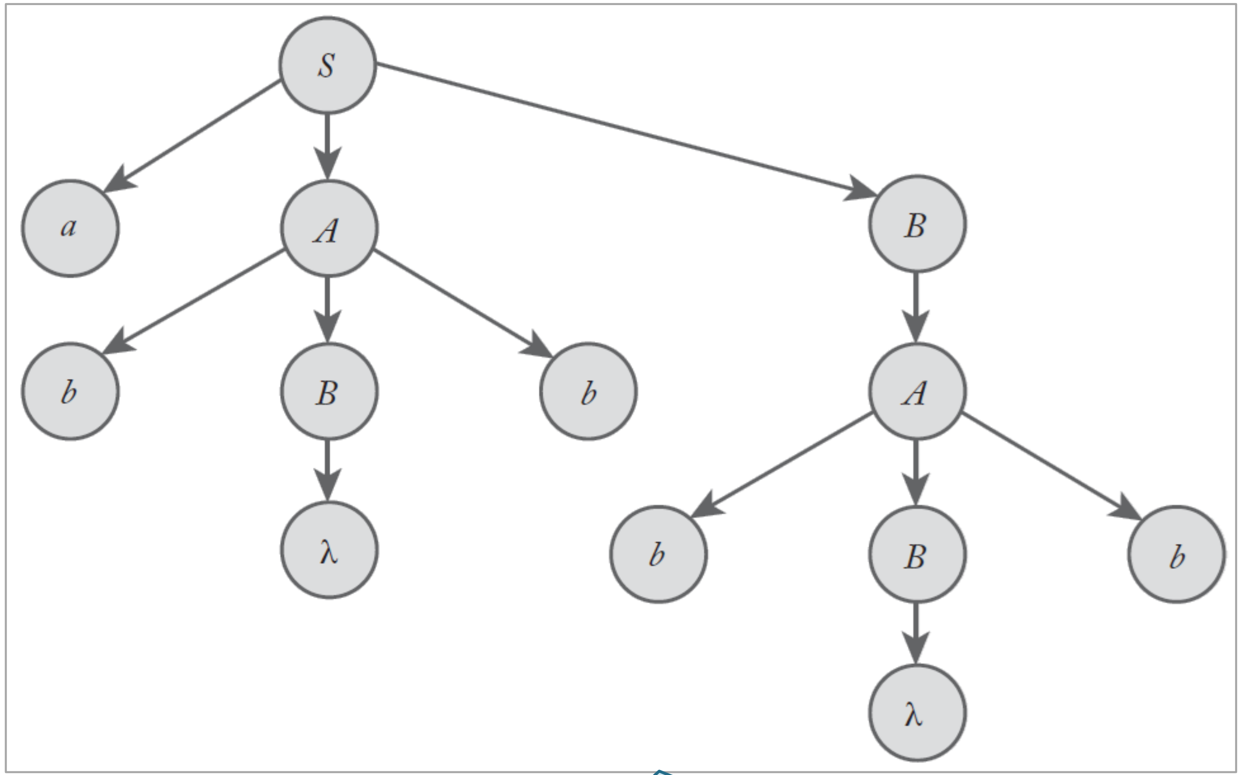
  $$A \to a_1 a_2 \cdots a_n.$$

  5. A leaf labeled λ has no siblings; that is, a vertex with a child labeled λ can have no other children.

Derivation Trees

- **Example:** $S \to aAB,$
  $A \to bBb,$
  $B \to A|\lambda.$





A **partial derivation tree**, which yields the **sentential form** ***abBbB***

A **derivation tree**, which yields the **sentence** ***abbbb***

Parsing and Ambiguity

- **Parsing**
  - **Finding a sequence of productions by which a $w \in L(G)$ is derived**
- **Exhaustive search parsing**
  - **Construct all possible (leftmost) derivations and see whether any of them match w**

Parsing and Ambiguity

- **Example**

Consider the grammar

$$S \to SS\,|aSb|\,bSa|\lambda$$

and the string $w = aabb$.

1. $S \Rightarrow SS,$
2. $S \Rightarrow aSb,$
3. $S \Rightarrow bSa,$
4. $S \Rightarrow \lambda.$

$S \Rightarrow SS \Rightarrow SSS,$
$S \Rightarrow SS \Rightarrow aSbS,$
$S \Rightarrow SS \Rightarrow bSaS,$
$S \Rightarrow SS \Rightarrow S,$

$S \Rightarrow aSb \Rightarrow aSSb,$
$S \Rightarrow aSb \Rightarrow aaSbb,$
$S \Rightarrow aSb \Rightarrow abSab,$
$S \Rightarrow aSb \Rightarrow ab.$

$S \Rightarrow aSb \Rightarrow aaSbb \Rightarrow aabb.$

Round 1 **Round 2** **Round 3**

Parsing and Ambiguity

- **Exhaustive search parsing has serious problems**
  1. **It is not efficient**
  2. **it is possible that it never terminates for strings not in $L(G)$**

To solve the second problem, we restrict the form that the grammar by removing **λ-productions** and **unit-productions**

- **In the exhaustive search parsing**
  - After removing **λ-productions** and **unit-productions,** a derivation cannot involve more than $2|w|$ rounds
  - The total number of sentential forms cannot exceed ($|P|$ is the number of productions)

$$M = |P| + |P|^2 + \cdots + |P|^{2|w|}$$
$$= O(P^{2|w|+1}).$$

Parsing and Ambiguity

- A context-free grammar $G = (V, T, S, P)$ is said to be a **simple grammar** or **s-grammar** if all its productions are of the form

$$A \to ax,$$

where $A \in V$, $a \in T$, $x \in V^*$, and any pair (A, a) occurs at most once in P.

- **Example**

The grammar

$$S \to aS \,|bSS|\, c$$

is an s-grammar. The grammar

$$S \to aS \,|bSS|\, aSS|c$$

is not an s-grammar because the pair (S, a) occurs in the two productions $S \to aS$ and $S \to aSS$.

If G is an s-grammar, then any string w in $L(G)$ can be parsed with an effort proportional to $|w|$.