Slides to Accompany *Programming Languages and Methodologies*

R. J. Schalkoff

Chapter 2, Part 3: Formal Grammars

The Derivation (or Parse) Tree

A derivation or parse tree, T, has the following characteristics:

1. The root of T is the starting symbol $S \in V_N$.
2. Leaf nodes of T are terminals $\in V_T$.
3. Interior[a] nodes are nonterminals $\in V_N$.
4. The children of any non-leaf node[b] represent the right hand side (RHS) of some production in P, where the parent node represents the corresponding LHS of the production.

Derivation trees are important when cataloging the production of x, since (except in cases of grammatical ambiguity) they show the structure of x *independently of the possible sequences*.

[a] neither root nor leaf

[b] recall leaf nodes have no children.

Derivation Tree Example

Recall the productions in an earlier grammar:

$$S \rightarrow AB$$

$$S \rightarrow C$$

$$A \rightarrow C$$

$$A \rightarrow a$$

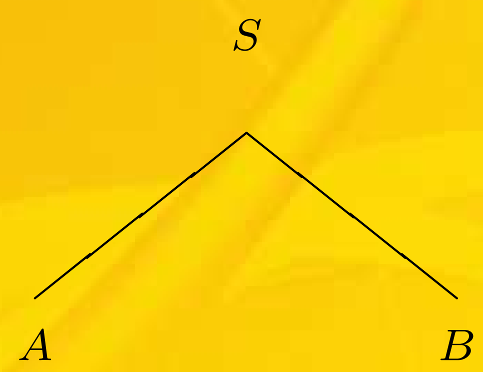
$$B \rightarrow b$$

$$B \rightarrow c$$

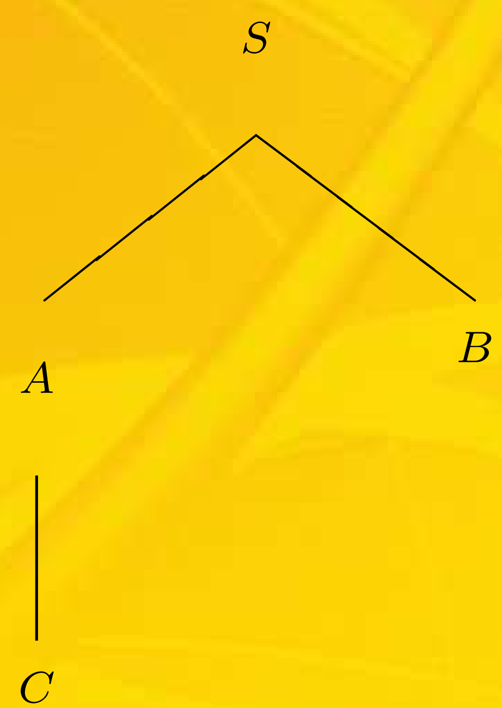
$$C \rightarrow d$$

First, we show the derivation of string dc, uisng the replacement sequence:

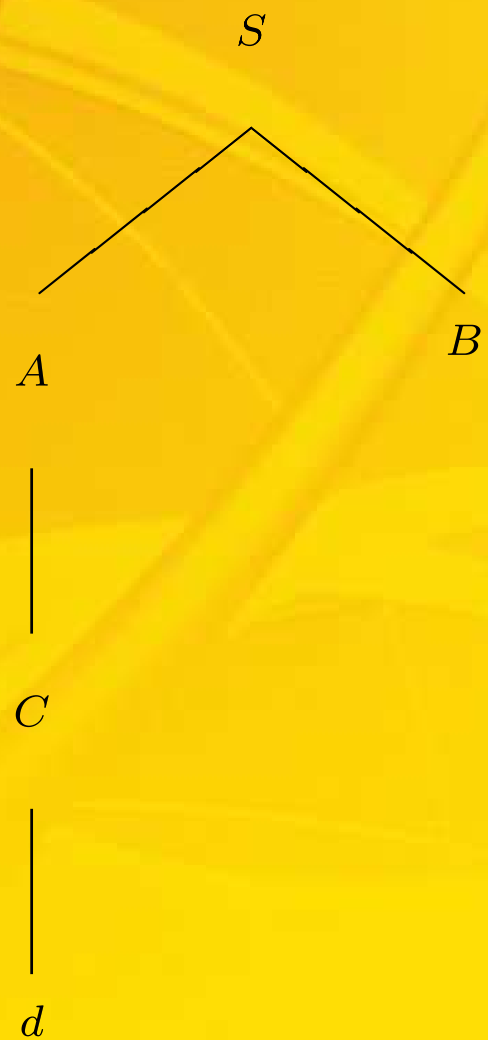
$$\boxed{S \Rightarrow AB \Rightarrow CB \Rightarrow dB \Rightarrow dc}$$



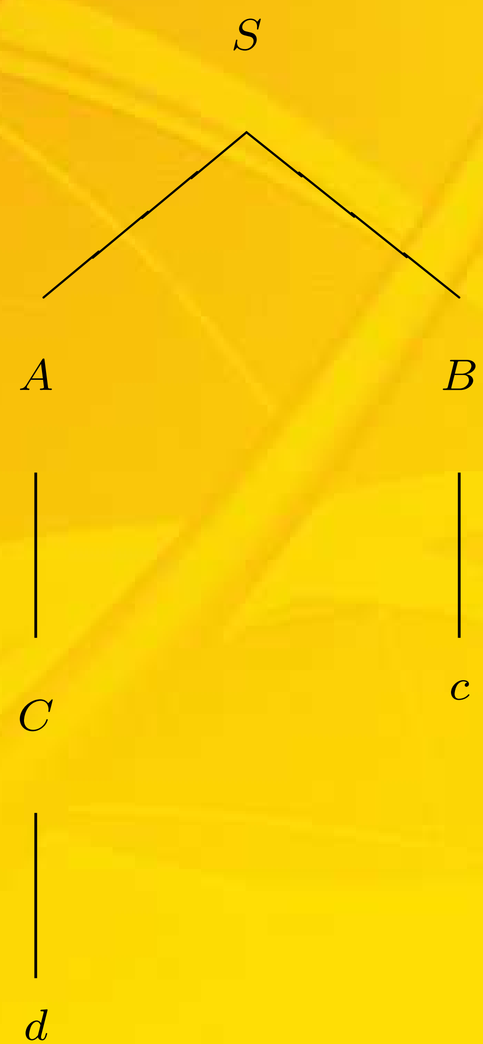
$$S \Rightarrow AB$$



$$AB \Rightarrow CB$$



$$CB \Rightarrow dB$$



$$dB \Rightarrow dc$$

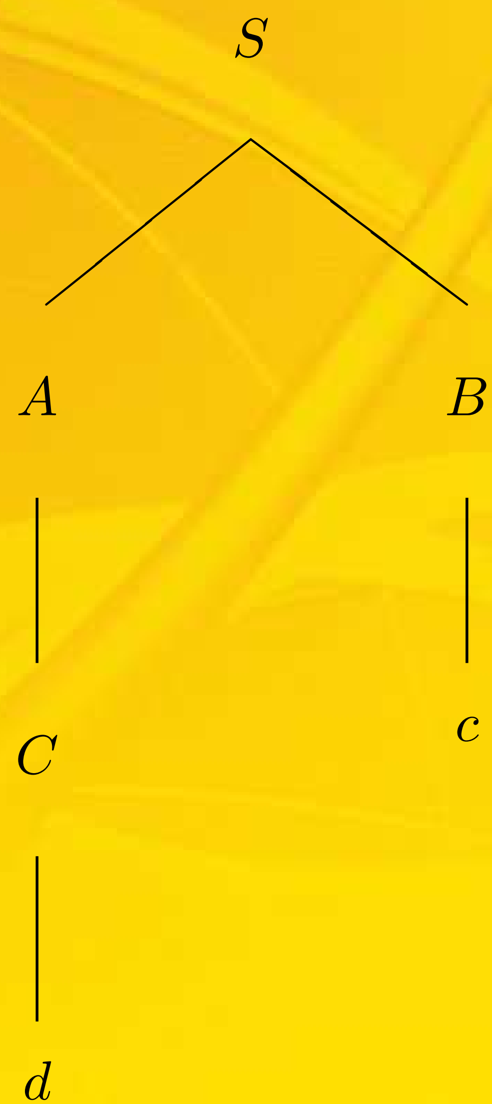
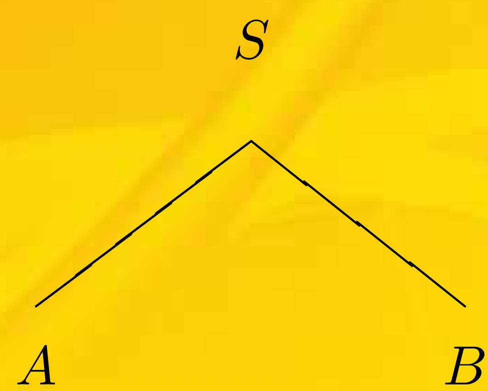


Figure 1: Derivation Tree Using the Production Sequence $S \Rightarrow AB \Rightarrow CB \Rightarrow dB \Rightarrow dc$

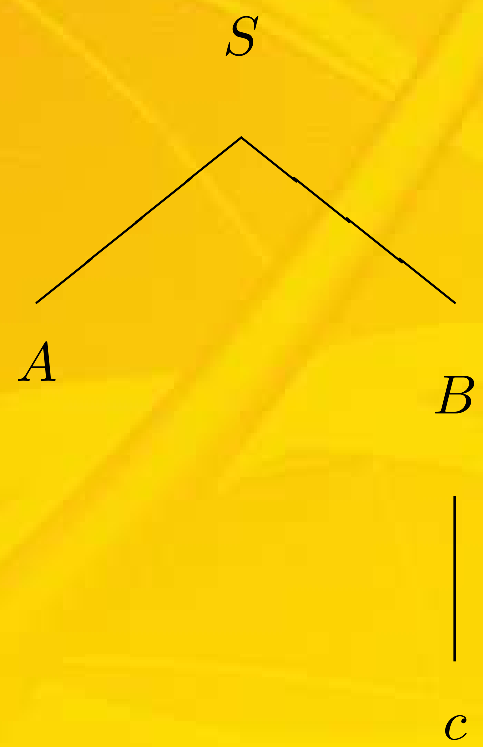
Now consider an alternative derivation *sequence* for the same string:

$$\boxed{S \Rightarrow AB \Rightarrow Ac \Rightarrow Cc \Rightarrow dc}$$

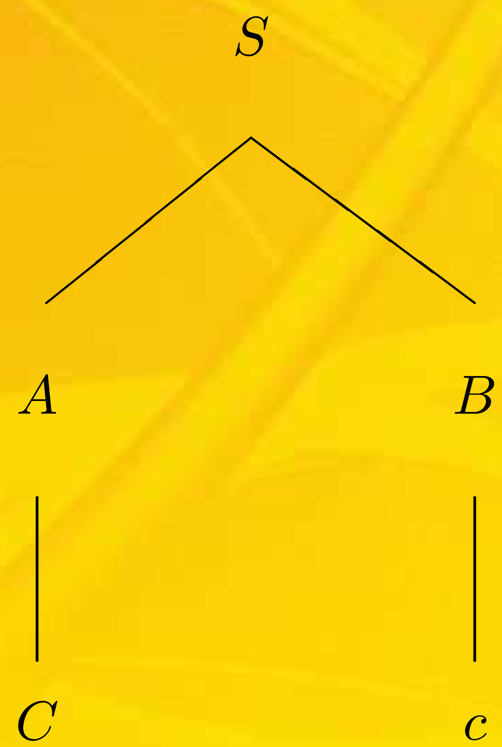
Development of the derivation tree with this sequence proceeds as follows:



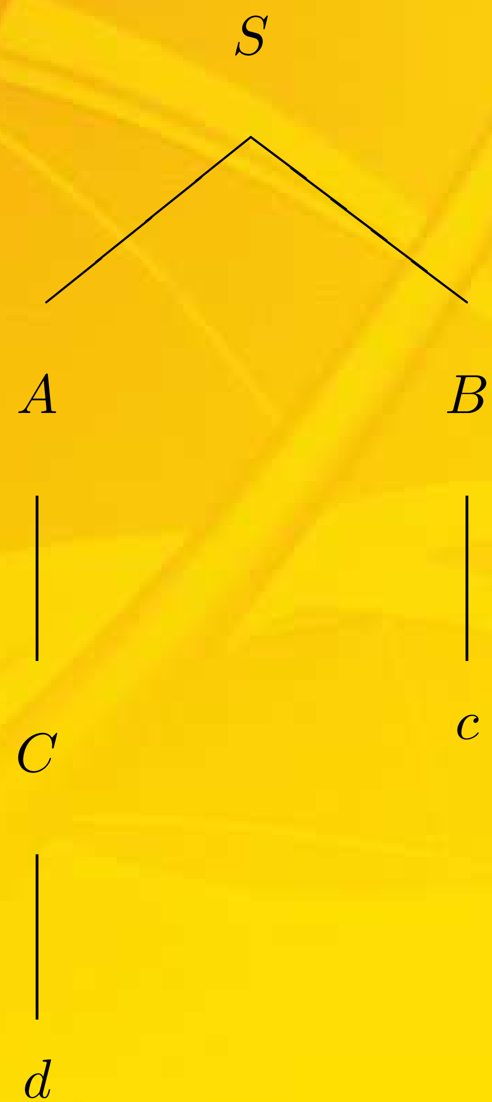
$$S \Rightarrow AB$$



$$AB \Rightarrow Ac$$



$$Ac \Rightarrow Cc$$



$$Cc \Rightarrow dc$$

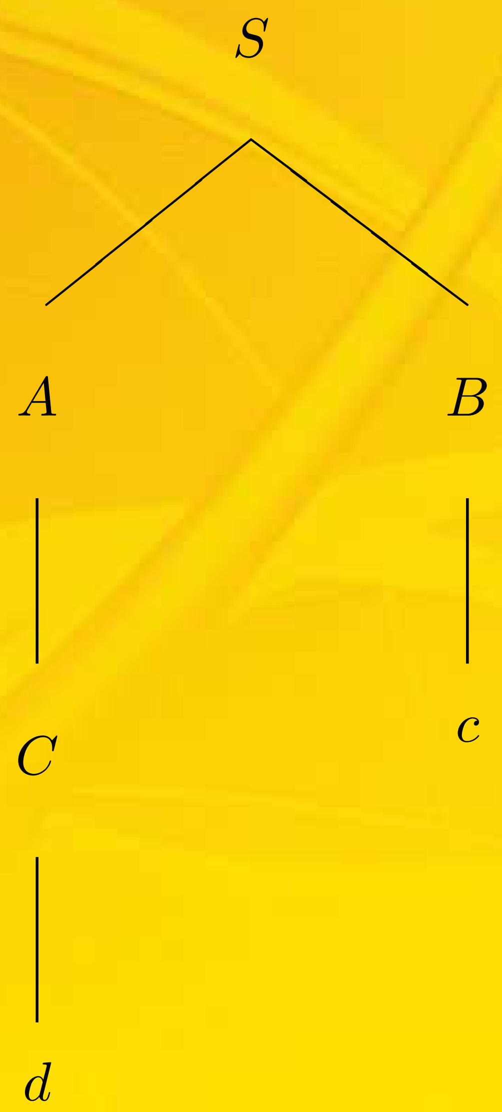


Figure 2: Derivation Tree Using the Production Sequence $S \Rightarrow AB \Rightarrow Ac \Rightarrow Cc \Rightarrow dc$

Grammatical/Syntactic Ambiguity

A grammar is ambiguous if any string in $L(G)$[a] *has two or more distinct derivation trees.*

---

[a] the language generated by grammar G

Classic Examples of Ambiguity

Classic examples of potential ambiguity in programming languages include:

- Strings employing binary mathematical operators. For example, what is the underlying structure of:

  $$a = b + c * a$$

  Of course, specification of operator precedence together with the syntax specification may eliminate this problem.

- The use of nested "if-then-(optional) else" constructs. This is a major source of potential ambiguity, hence the syntax specification is usually augmented with a description of the corresponding interpretation algorithm; and

- Confusion between function invocation and array use. For example, how would (could) you interpret: $a(2)$? Is this function a called with argument 2 or the second (or third) element of array a?