CS3311
Oct 15, 2012
Monday (1)

Last time: CFG examples

$S \to ABC$
$A \to aA \mid \lambda$
$B \to bB \mid \lambda$
$C \to cC \mid c$

$S \Rightarrow \underline{A}\underline{B}\underline{C} \Rightarrow aABC \Rightarrow aaABC$
$\Rightarrow aaBC \Rightarrow aaC \Rightarrow aacC \Rightarrow aacc$

leftmost derivation : always ~~do~~ replace the leftmost variable in a string.

rightmost derivation : always replace the rightmost variable in a string.

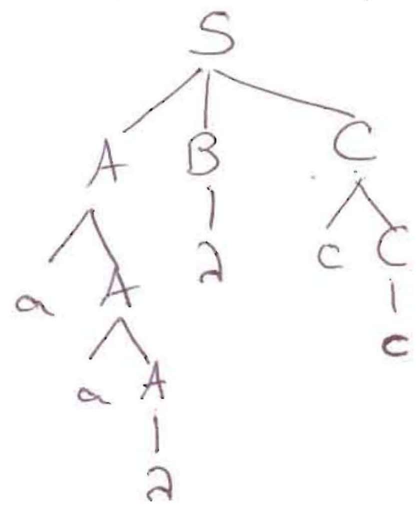
$S \Rightarrow ABC \Rightarrow ABcC \Rightarrow ABcc$
$\Rightarrow Acc \Rightarrow aAcc \Rightarrow aaAcc$
$\Rightarrow aacc$

Main idea: the order the variables are replaced does not matter.

A derivation tree (DT) of $S \xRightarrow{*} w$ is an ordered tree which can be built iteratively as follows:-

1. initialize DT to root S
2. if $A \to x_1 x_2 \cdots x_n$ with $x_i \in (V \cup \Sigma)$ is the rule applied to the string uAv, then add $x_1 \cdots x_n$ as children of A.
3. if $A \to \lambda$ is the rule, then add λ as the only child of A.

$$S \Rightarrow ABC \Rightarrow aABC \Rightarrow aaABC \Rightarrow aaBC \Rightarrow aaC \Rightarrow aacC \Rightarrow aacc$$



$aa\lambda\lambda cc$

$aacc$

Definition 3.5.1

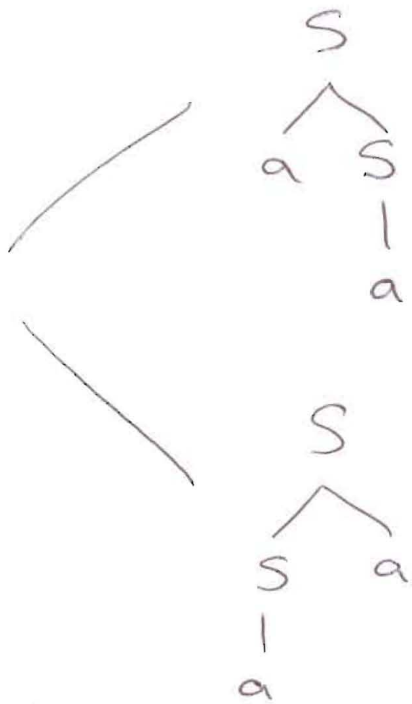
A context-free grammar is ambiguous if there is a string $w \in L(G)$ that can be described using two distinct leftmost derivations.

A grammar that is not ambiguous is called unambiguous.

$$S \to aS \mid Sa \mid a$$

$$S \Rightarrow aS \Rightarrow aa$$

$$S \Rightarrow Sa \Rightarrow aa$$



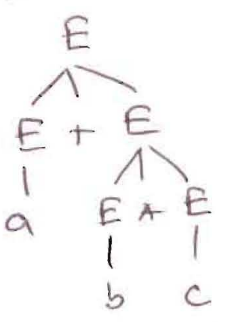
$$E \to E+E \mid E*E \mid a \mid b \mid c$$

$a+b*c$

(I) $E \Rightarrow E+E \Rightarrow a+E \Rightarrow a+E*E \Rightarrow a+b*E \Rightarrow a+b*c$

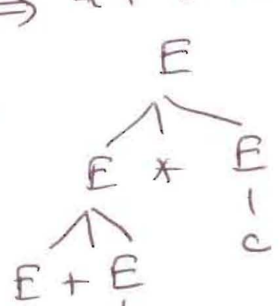
(II) $E \Rightarrow E*E \Rightarrow E+E*E \Rightarrow a+E*E \Rightarrow a+b*E \Rightarrow a+b*c$

(I)



$a+(b*c)$

(II)



$(a+b)*c$

Some context-free ~~grammars~~ languages cannot be generated using unambiguous grammars. Such languages are called inherently ambiguous.

$L = \{ a^n b^m c^k \mid n = m \ \boxed{\text{or}}\ m = k$ where $n, m, k \geq 0 \}$

$$\underbrace{\begin{aligned} S &\Rightarrow \cancel{A}^{\,n}\ a S b C \mid \lambda \\ C &\rightarrow cC \mid \lambda \end{aligned}}_{\text{for } n=m}$$

abc abccc

does not allow only c's

$S \rightarrow a S b \underline{\underline{C}} \mid C$

$C \rightarrow c C \mid \lambda$

only one a or one b

$S \Rightarrow a S b C \Rightarrow a a S b C b C$

$S \rightarrow a S b \mid C$

$C \rightarrow c C \mid \lambda$

$S \Rightarrow a S b \Rightarrow a a S b b \Rightarrow a a C b b$

$S \rightarrow A \mid C$

$A \rightarrow a A b \mid \lambda$

$C \rightarrow c C \mid \lambda$

$S \rightarrow X C \mid A Y$

$X \rightarrow a X b \mid \lambda$

$Y \rightarrow b Y c \mid \lambda$

$A \rightarrow a A \mid \lambda$

$C \rightarrow c C \mid \lambda$

$S \to AC$

$A \to aAb \mid \lambda$

$C \to cC \mid \lambda$

$\overbrace{\boxed{a^n b^m}}^{A}\overbrace{\boxed{c^k}}^{C}$

$n = m$

$\underbrace{a^i b^i}_{A}\ \underbrace{c^k}_{C}$

$S \to XC$

$X \to aXb \mid \lambda$

$C \to cC \mid \lambda$

$S \to T_1 \mid T_2$

$T_1 \to AB$

$A \to aAb \mid \lambda$

$B \to cB \mid \lambda$

T_1 takes care of $n=m$

$T_2 \to CD$

$C \to aC \mid \lambda$

$D \to bDc \mid \lambda$

T_2 takes care of $m=k$