CSE 396 Lecture Thu 3/15 Spr 2018

Top Hat # 8744

8744

Def^n: Given a CFG $G = (V, \Sigma, R, S)$ a **parse tree** for a string $x \in L(G)$ has:

- Root labeled S

Later: Can consider any variable A at the root.

- Leaves labeled with terminals or ε.
- Interior nodes have labels $A \in V$.

If its children are $u_1, \dots, u_m$, then $A \to u_1 \cdots u_m$ is a rule in R.

Example: $\Sigma = \{+, -, *, /, (,)\} \cup \{$digits and alphanumeric IDs$\}$.

$E \to \langle const \rangle \mid \langle var \rangle \mid E+E \mid E-E \mid E*E \mid E/E \mid (E)$

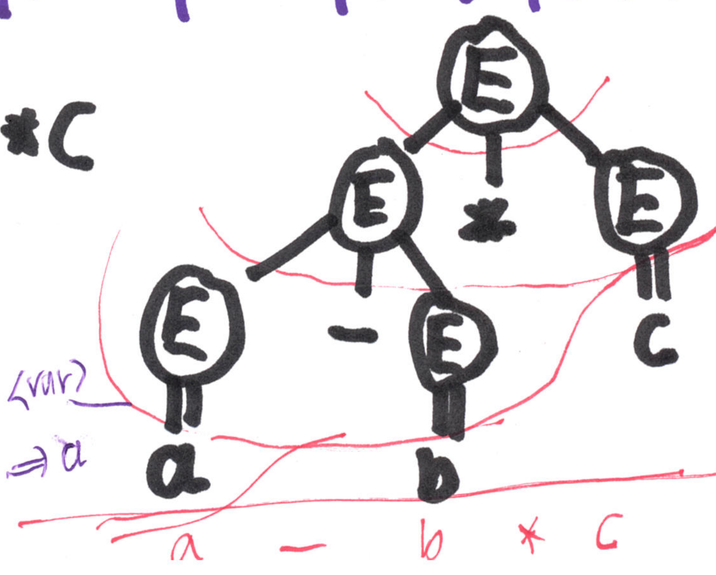
example: $a - b*c$

$E \Rightarrow E*E \Rightarrow E-E*E$

$\Rightarrow^* a - E*E$

$\Rightarrow^* a - b*E$

$\Rightarrow^* a - b*c.$



via $E \Rightarrow \langle var \rangle$

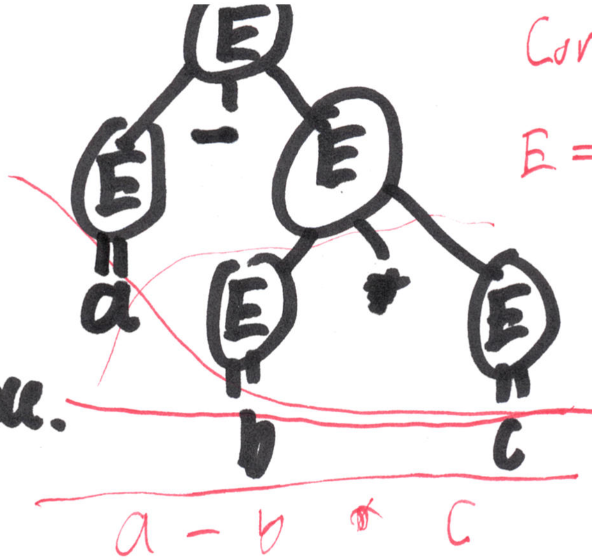
$\langle var \rangle \Rightarrow a$

Legal Parse Tree. A derivation using the tree at left

final yield

"Nicer" parse:

Follows rules of precedence.



Corresponding leftmost derivation: (2)

$E \Rightarrow \underline{E} - E \Rightarrow a - \underline{E} \Rightarrow a - \underline{E} * E \Rightarrow a - b * c$

Defn: A derivation is leftmost if each step expands the leftmost variable.

Fact: Parse trees are in 1-1 corresp with leftmost derivations.

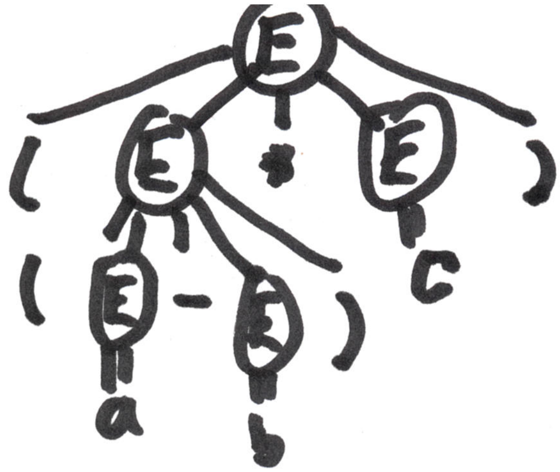
Non LM deriv: $E \Rightarrow E - \underline{E} \Rightarrow E - E * E \Rightarrow a - E * E$ etc.

- **Any** derivation $\vec{d}$ gives a unique parse tree T.
- Any T gives a unique LM deriv [left-to-right preorder]

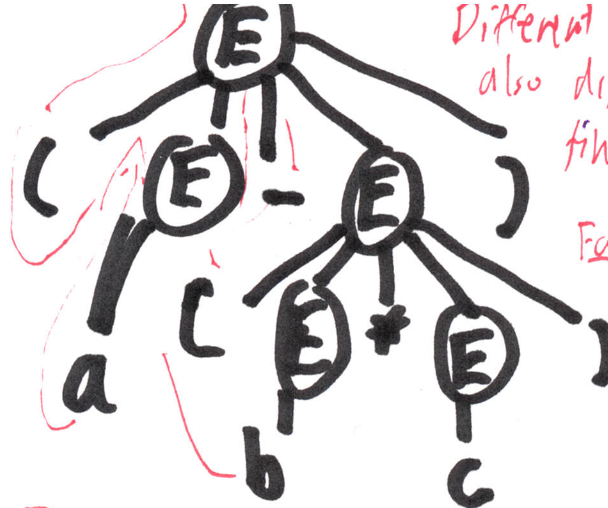
Defn: A string $x \in L(G)$ is unambiguous in G if it has a unique parse tree, equivalently if it has a unique leftmost derivation in G.

Defn: G is unambiguous if every $x \in L(G)$ is unambiguous.
G is ambiguous if some $x \in L(G)$ is ambiguous.

G': E → <const> | <var> | (E+E) | (E-E) | (E*E) | (E/E) | (E
is unambiguous.



$((a-b)*c) \neq$



$(a-(b*c))$

Different trees, but also different at final strings.

Fact: G' is unambiguous.

¿Can we design a CFG G'' s.t. L(G'') includes formulas such as $a-b*c$ the way we naturally write them, but is unambiguous?

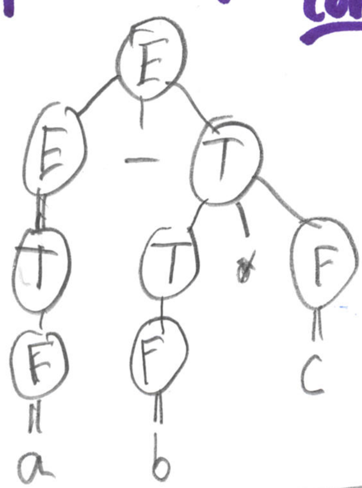
A: Yes, but we need more variables, ie, more "Syntactic categories".

Expression E, Term T, Factor F $V = \{E, T, F\}$

G''

$$E \rightarrow T \mid E+T \mid E-T$$
$$T \rightarrow F \mid T*F \mid T/F$$
$$F \rightarrow \underline{\text{const}} \mid \underline{\text{variable}} \mid (E).$$

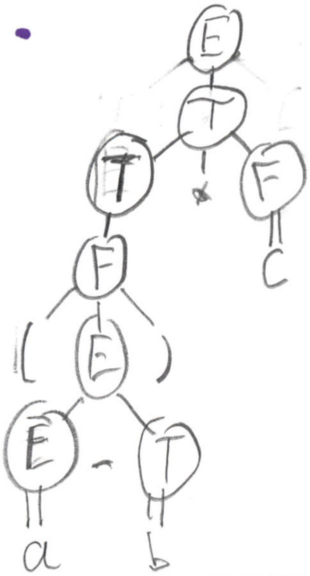
Try to derive $a-b*c$ in "both ways".



$a - b * c$

The yields "$a-b*c$" and "$(a-b)*c$" are different strings. Hence this "ETF grammar" avoids the ambiguity.

Fact (not proved in text either) This G'' is unambiguous.



$(a-b)*c$

Q: What is $30/_{2\cdot 3}$?

$\underbrace{30/2}_{T}*\underbrace{3}_{F}$

Try

$E \Rightarrow T \Rightarrow T/F \Rightarrow F/F \Rightarrow 30/F$
$\Rightarrow 30/(E) \Rightarrow 30/(T) \Rightarrow 30/(T*F)$
$\Rightarrow 30/(F*F) \Rightarrow 30/(2*F) \Rightarrow 30/(2*3).$

This yielded a different string = how we should have grouped it if we want 5. to be the answer.

$E \Rightarrow T \Rightarrow \underline{T}*F \Rightarrow T/F*F$
$\Rightarrow F/F*F \Rightarrow 30/F*F \Rightarrow 30/2*F \Rightarrow 30/2*3$

What about a**b**c? $= a^{b^c}$ Group as $(a^b)^c$ or $\underline{a^{(b^c)}}$?
Because $(a^b)^c = a^{(bc)}$

However, this means "**" operator must have right precedence, hence any CFG giving it must use a syntactic category "~~[illegible]~~ between T and F for it. So complicated that many PLangs skip giving a ** operator.

Note also: Function composition associates to the right: $f\circ g\circ h = f\circ(g\circ h)$.

An often-tolerated ambiguity: the "Dangling If-Else"

```
if (a>0)
    if (b>0)
        stmt1
else
    stmt2.
```

G is ambiguous but often used in the above form.

$G: I \rightarrow \$I \mid \$IdI \mid \varepsilon$

$x = \$\$d \in L(G)$. Which \$ killed the d?