Theory of Automata

Ambiguous Grammars

Dr. Sabina Akhtar

Revision

Ambiguity in CFGs

- A CFG is said to be *ambiguous* if there exists a string which has more than one left-most derivation

Example:

S ==> AS | ε

A ==> A1 | 0A1 | 01

LM derivation #1:

S => AS

=> 0A1S

=>0A11S

=> 00111S

=> 00111

LM derivation #2:

S => AS

=> A1S

=> 0A11S

=> 00111S

=> 00111

Input string: 00111

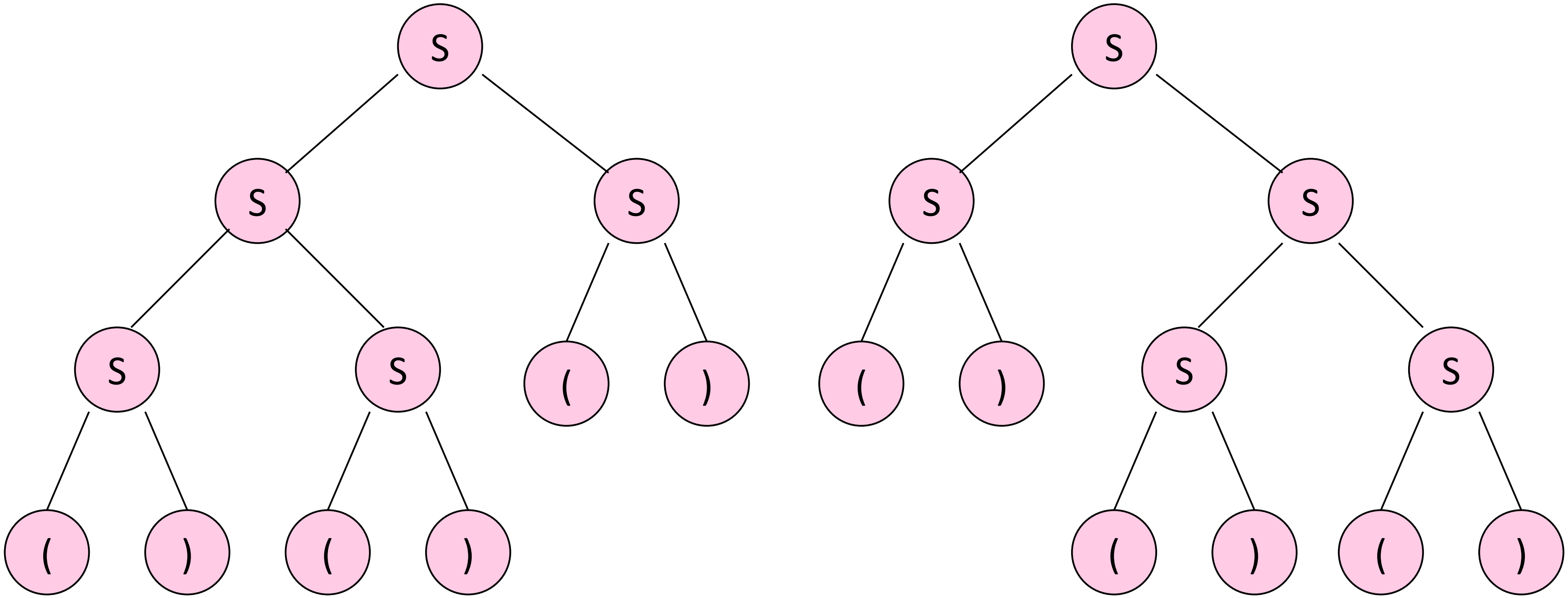
Can be derived in two ways

Ambiguous Grammars

- A CFG is *ambiguous* if there is a string in the language that is the yield of two or more parse trees.
- Example: S -> SS | (S) | ()
- Two parse trees for ()()().

- Draw the parse trees.

Example

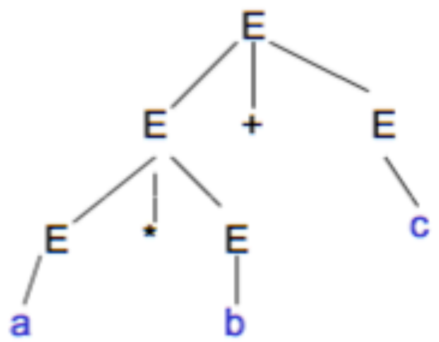


Why does ambiguity matter?

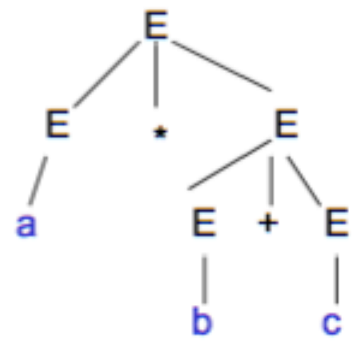
E ==> E + E | E * E | (E) | a | b | c | 0 | 1

*string = a * b + c*

- <u>LM derivation #1:</u>
  - E => E + E => E * E + E ==>* a * b + c



- <u>LM derivation #2</u>
  - E => E * E => a * E => a * E + E ==>* a * b + c



Removing Ambiguity in Expression Evaluations

- It MAY be possible to remove ambiguity for some CFLs
  - E.g.,, in a CFG for expression evaluation by imposing rules & restrictions such as precedence
  - This would imply rewrite of the grammar

Removing Ambiguity in Expression Evaluations

- It MAY be possible to remove ambiguity for some CFLs
  - E.g.,, in a CFG for expression evaluation by imposing rules & restrictions such as precedence
  - This would imply rewrite of the grammar

- Precedence: (), * , +

Modified unambiguous version:

E => E + T | T
T => T * F | F
F => I | (E)
I => a | b | c | 0 | 1

Ambiguous version:

E ==> E + E | E * E | (E) | a | b | c | 0 | 1

How will this avoid ambiguity?

Parse tree for a*b+c and a+b*c

a+b+c

Ambiguity, Left- and Rightmost Derivations

- If there are two different parse trees, they must produce two different leftmost derivations by the construction given in the proof.
- Conversely, two different leftmost derivations produce different parse trees by the other part of the proof.
- Likewise for rightmost derivations.

Ambiguity, etc. – (2)

- Thus, equivalent definitions of "ambiguous grammar" are:
  1. There is a string in the language that has two different leftmost derivations.
  2. There is a string in the language that has two different rightmost derivations.

Ambiguity is a Property of Grammars, not Languages

- For the balanced-parentheses language, here is another CFG, which is unambiguous.

B, the start symbol, derives balanced strings.

B -> (RB | ϵ

R ->) | (RR

R generates strings that have one more right paren than left.

Class Activity: Unambiguous Grammar

B -> (RB | ϵ

R ->) | (RR

- Construct a parse tree for ()()() check if there are more than 1 trees possible.

LL(1) Grammars

- As an aside, a grammar such B -> (RB | ϵ R ->) | (RR, where you can always figure out the production to use in a leftmost derivation by scanning the given string left-to-right and looking only at the next one symbol is called LL(1).
  - "Leftmost derivation, left-to-right scan, one symbol of lookahead."

LL(1) Grammars – (2)

- Most programming languages have LL(1) grammars.
- LL(1) grammars are never ambiguous.

Inherent Ambiguity

- It would be nice if for every ambiguous grammar, there were some way to "fix" the ambiguity, as we did for the balanced-parentheses grammar.
- Unfortunately, certain CFL's are *inherently ambiguous*, meaning that every grammar for the language is ambiguous.

Example: Inherent Ambiguity

- The language $\{0^i1^j2^k \mid i = j \text{ or } j = k\}$ is inherently ambiguous.
- Intuitively, at least some of the strings of the form $0^n1^n2^n$ must be generated by two different parse trees, one based on checking the 0's and 1's, the other based on checking the 1's and 2's.

Class Activity

S -> AB | CD

A -> 0A1 | 01

B -> 2B | 2

C -> 0C | 0

D -> 1D2 | 12

Is the above grammar ambiguous?
If yes prove it by providing 2 parse trees for a word of length greater than 6.
(Start variable is *S*)

Class Activity

2.46 Consider the following CFG G:

$$S \to SS \mid T$$
$$T \to \mathtt{a}T\mathtt{b} \mid \mathtt{ab}$$

Describe $L(G)$ and show that G is ambiguous.

Class Activity

- Design CFG for
  - $\{0^i 1^j 2^k \mid i = j \text{ or } j = k\}$

Exercise

[A]**2.3** Answer each part for the following context-free grammar G.

$$\begin{aligned} R &\to XRX \mid S \\ S &\to \mathtt{a}T\mathtt{b} \mid \mathtt{b}T\mathtt{a} \\ T &\to XTX \mid X \mid \varepsilon \\ X &\to \mathtt{a} \mid \mathtt{b} \end{aligned}$$

a. What are the variables of G?
b. What are the terminals of G?
c. Which is the start variable of G?
d. Give three strings in $L(G)$.
e. Give three strings *not* in $L(G)$.
f. True or False: $T \Rightarrow \mathtt{aba}$.
g. True or False: $T \overset{*}{\Rightarrow} \mathtt{aba}$.
h. True or False: $T \Rightarrow T$.
i. True or False: $T \overset{*}{\Rightarrow} T$.
j. True or False: $XXX \overset{*}{\Rightarrow} \mathtt{aba}$.
k. True or False: $X \overset{*}{\Rightarrow} \mathtt{aba}$.
l. True or False: $T \overset{*}{\Rightarrow} XX$.
m. True or False: $T \overset{*}{\Rightarrow} XXX$.
n. True or False: $S \overset{*}{\Rightarrow} \varepsilon$.
o. Give a description in English of $L(G)$.

Exercise

2.4 Give context-free grammars that generate the following languages. In all parts, the alphabet Σ is $\{0,1\}$.

- [A] **a.** $\{w \mid w \text{ contains at least three 1s}\}$
- **b.** $\{w \mid w \text{ starts and ends with the same symbol}\}$
- **c.** $\{w \mid \text{the length of } w \text{ is odd}\}$
- [A] **d.** $\{w \mid \text{the length of } w \text{ is odd and its middle symbol is a 0}\}$
- **e.** $\{w \mid w = w^{\mathcal{R}}, \text{ that is, } w \text{ is a palindrome}\}$
- **f.** The empty set

References

- Book Chapter
- Lectures from Stanford University
  - http://infolab.stanford.edu/~ullman/ialc/spr10/spr10.html#LECTURE%20NOTES
- Lectures from Washington State University
  - http://www.eecs.wsu.edu/~ananth/CptS317/Lectures/