Context-Free Grammar

CSE 211 (Theory of Computation)

Tanjeem Azwad Zaman

Adjunct Lecturer
Department of Computer Science and Engineering
Bangladesh University of Engineering & Technology

Adapted from slides by
Dr. Muhammad Masroor Ali & Dr. Atif Hasan Rahman

Ambiguous Grammars

- A string w is derived ***ambiguously*** in context-free grammar G if it has two or more different parse trees
  - or equivalently two or more different leftmost derivations
  - two or more different rightmost derivations
- Grammar G is ***ambiguous*** if it generates some string ambiguously

Ambiguous Grammars

$$
\begin{array}{llll}
1. & E & \rightarrow & I \\
2. & E & \rightarrow & E + E \\
3. & E & \rightarrow & E * E \\
4. & E & \rightarrow & (E) \\
\\
5. & I & \rightarrow & a \\
6. & I & \rightarrow & b \\
7. & I & \rightarrow & Ia \\
8. & I & \rightarrow & Ib \\
9. & I & \rightarrow & I0 \\
10. & I & \rightarrow & I1
\end{array}
$$

Figure 5.2: A context-free grammar for simple expressions

Ambiguous Grammars

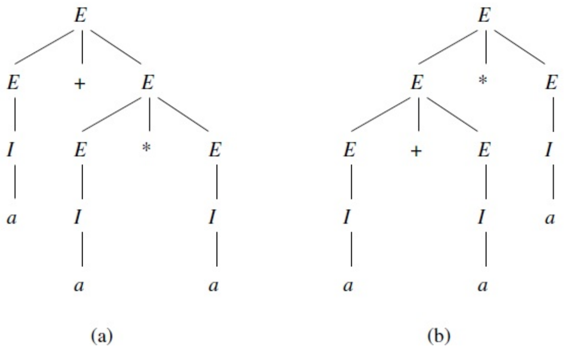

Figure 5.18: Trees with yield $a + a * a$, demonstrating the ambiguity of our expression grammar

Removing Ambiguity

- No known algorithm to remove ambiguity
- No algorithm to even tell that a grammar is ambiguous
- There are well known techniques for eliminating ambiguity

Removing Ambiguity

Two causes of ambiguity in our expression grammar

- The precedence of operators is not respected. We need to group the * before the + operator
- A sequence of identical operators can group either from the left or from the right. Since addition and multiplication are associative, it doesn't matter whether we group from the left or the right, but to eliminate ambiguity, we must pick one

Removing Ambiguity

We introduce several variables

- A factor is an expression that cannot be broken apart by any adjacent operator. Factors can be
  - Identifiers. It is not possible to separate the letters of an identifier by attaching an operator
  - Any parenthesized expression, no matter what appears inside the parentheses
- A term is an expression that cannot be broken by the + operator. In our example, where + and * are the only operators, a term is a product of one or more factors
- An expression will refer to any possible expression, including those that can be broken by either an adjacent + or an adjacent *

Removing Ambiguity

$$
\begin{array}{lcl}
I & \rightarrow & a \mid b \mid Ia \mid Ib \mid I0 \mid I1 \\
F & \rightarrow & I \mid (E) \\
T & \rightarrow & F \mid T * F \\
E & \rightarrow & T \mid E + T
\end{array}
$$

Figure 5.19: An unambiguous expression grammar

Removing Ambiguity

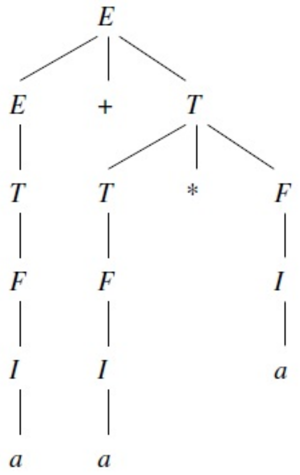

Figure 5.20: The sole parse tree for $a + a * a$

Inherent Ambiguity

- A context free language L is said to be ***inherently ambiguous*** if all its grammars are ambiguous
- If even one grammar for L is unambiguous, then L is an unambiguous language
- There are inherently ambiguous languages (we will not prove the statement)
- An example of an inherently ambiguous language

$$L = \{a^n b^n c^m d^m | n \geq 1, m \geq 1\} \cup \{a^n b^m c^m d^n | n \geq 1, m \geq 1\}$$

Inherent Ambiguity

- CFG for the lanuage
  $L = \{a^n b^n c^m d^m | n \geq 1, m \geq 1\} \cup \{a^n b^m c^m d^n | n \geq 1, m \geq 1\}$

Inherent Ambiguity

- CFG for the lanuage
  $L = \{a^n b^n c^m d^m | n \geq 1, m \geq 1\} \cup \{a^n b^m c^m d^n | n \geq 1, m \geq 1\}$

- $S \rightarrow S_1 | S_2$

Inherent Ambiguity

- CFG for the lanuage
  $L = \{a^n b^n c^m d^m | n \geq 1, m \geq 1\} \cup \{a^n b^m c^m d^n | n \geq 1, m \geq 1\}$

- $S \rightarrow S_1 | S_2$
- $S_1 \rightarrow AB$

Inherent Ambiguity

- CFG for the lanuage
  $L = \{a^n b^n c^m d^m | n \geq 1, m \geq 1\} \cup \{a^n b^m c^m d^n | n \geq 1, m \geq 1\}$

- $S \rightarrow S_1 | S_2$
- $S_1 \rightarrow AB$
- $A \rightarrow aAb | ab$
- $B \rightarrow cBd | cd$

Inherent Ambiguity

- CFG for the lanuage
  $L = \{a^n b^n c^m d^m | n \geq 1, m \geq 1\} \cup \{a^n b^m c^m d^n | n \geq 1, m \geq 1\}$

- $S \rightarrow S_1 | S_2$
- $S_1 \rightarrow AB$
- $A \rightarrow aAb | ab$
- $B \rightarrow cBd | cd$
- $S_2 \rightarrow C$
- $C \rightarrow aCd | aDd$

Inherent Ambiguity

- CFG for the lanuage
  $L = \{a^n b^n c^m d^m | n \geq 1, m \geq 1\} \cup \{a^n b^m c^m d^n | n \geq 1, m \geq 1\}$

- $S \rightarrow S_1 | S_2$
- $S_1 \rightarrow AB$
- $A \rightarrow aAb | ab$
- $B \rightarrow cBd | cd$
- $S_2 \rightarrow C$
- $C \rightarrow aCd | aDd$
- $D \rightarrow bDc | bc$

Inherent Ambiguity

$$
\begin{array}{lcl}
S & \rightarrow & AB \mid C \\
A & \rightarrow & aAb \mid ab \\
B & \rightarrow & cBd \mid cd \\
C & \rightarrow & aCd \mid aDd \\
D & \rightarrow & bDc \mid bc
\end{array}
$$

Figure 5.22: A grammar for an inherently ambiguous language

Inherent Ambiguity

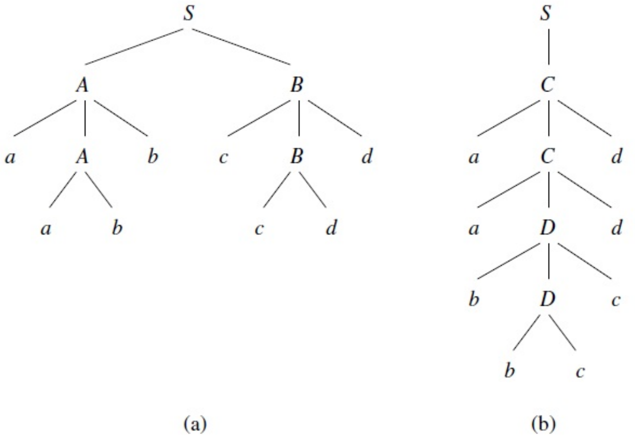

Figure 5.23: Two parse trees for *aabbccdd*