COEN 175

Lecture 3: Disambiguating Expression Grammars

Enforcing Precedence

- We want multiplication to have higher precedence than addition.
- We need multiplication lower (deeper) in the parse tree than addition.
- Parse trees grow when we apply productions.
- Therefore, we need a longer chain of productions to reach multiplication than to reach addition.
- We need addition to be a list of multiplications, not the other way around.

Disambiguated Grammar

- Here is our disambiguated grammar:

  $E \rightarrow E + T \mid T$
  $T \rightarrow T * F \mid F$
  $F \rightarrow$ **id**

- An expression, *E*, is a list of terms that are added together (from left to right).
- A term, *T*, is a list of factors that are multiplied together (from left to right).
- Draw the parse trees for the following sentences:
  - **id** + **id** $*$ **id**.
  - **Id** + **id** + **id**.

Expressions

- In order to disambiguate an expression grammar, we must know two things:
  - **Precedence** tells us the order of operations.
  - **Associativity** tells us how operators of the same precedence are grouped.
- A related term is **arity**, which is the number of operands an operator expects:
  - If the arity is one, the operator is called **unary**.
  - If the arity is two, the operator is called **binary**.
  - If the arity is three, the operator is called **ternary**.

Enforcing Precedence

- Operators of higher precedence belong lower in the parse tree.
- Therefore, we need a longer chain of productions to reach them.
- Thus, we must place productions for operators of lower precedence before those for operators of higher precedence.
- Operators at the same level of precedence, such as + and –, are grouped together.

Binary Operators

- If the operator is left associative, its grammar rule is left recursive. If the operator is right associative, its grammar rule is right recursive.
- Disambiguate the following grammar:
  - $E \rightarrow E * E \mid E + E \mid E - E \mid E / E \mid$ **id**.
- The correct, disambiguated grammar is:

  $E \rightarrow E + T \mid E - T \mid T$
  $T \rightarrow T * F \mid T / F \mid F$
  $F \rightarrow$ **id**
- Draw the parse tree for the input **id** + **id** $*$ **id** / **id**.

Unary Operators

- Unary operators are either prefix or postfix:
  - Prefix operators are always right associative.
  - Postfix operators are always left associative.
- Disambiguate the following grammar:
  - $E \rightarrow E + E \mid E * E \mid - E \mid \mathbf{id}$.
- The correct, disambiguated grammar is:

  $E \rightarrow E + T \mid T$
  $T \rightarrow T * F \mid F$
  $F \rightarrow - F \mid \mathbf{id}$

- Draw the parse tree for the input **id** + – **id** $*$ **id** + **id**.

Parentheses

- Parentheses are used to override precedence:
  - In effect, they have the highest precedence.
  - Any expression is legal within the parentheses.
- Disambiguate the following grammar:
  - $E \rightarrow E + E \mid E * E \mid (E) \mid \mathbf{id}$.
- The correct, disambiguated grammar is:

  $E \rightarrow E + T \mid T$
  $T \rightarrow T * F \mid F$
  $F \rightarrow (E) \mid \mathbf{id}$
- Draw the parse tree for the input **id** + (**id** + **id**) $*$ **id**.