Recap: Ambiguity (1)

A CFG $G = (N, T, P, S)$ is ambiguous if there is at least one word $w \in L(G)$ such that there are

- two different derivation trees, or
- two different left-most derivations, or
- two different right-most derivations

for $w$.

Recap: Ambiguity (2)

Ambiguity can be problematic for a number of reasons, including that the structure of a derivation tree often is used to suggest a meaning for the word.

Example: Arithmetic Expressions

Recap: Ambiguity (3)

$SAE = (N = \{E, I, D\}, T = \{+, *, (, ), 0, 1, \ldots, 9\}, P, E)$ where $P$ is given by:

- $E \rightarrow E + E$
- $E \rightarrow E \ast E$
- $E \rightarrow (E)$
- $E \rightarrow I$
- $I \rightarrow DI | D$
- $D \rightarrow 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9$

Recap: Ambiguity (4)

Consider: $1 + 2 \ast 3$. Two derivation trees:

Disambiguating Grammars

Given an ambiguous grammar $G$, it is often possible to construct an equivalent grammar $G'$ (i.e., $L(G) = L(G')$), such that $G'$ is not ambiguous.

Some languages are inherently ambiguous CFLs, meaning that every CFG generating the language necessarily is ambiguous.

We will consider exploiting

- Operator Precedence
- Associativity

to disambiguate expression grammars as an example.