Recap: Ambiguity (1)

A CFG $G = (N, T, P, S)$ is *ambiguous* is 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:

$$
E \to E + E \\
| \quad E \cdot E \\
| \quad (E) \\
| \quad I \\
I \to DI \mid D \\
D \to 0 \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9
$$
Recap: Ambiguity (4)

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

```
  E   E   E
 / \ / \ / \\
+   *   + \\
I   I   I
/ \ / \ / \\
D   I   D
|   |   |
1   2   3
```

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.