Ambiguity

A grammar is ambiguous if there exists at least two distinct parse trees for the derivation of the same string.

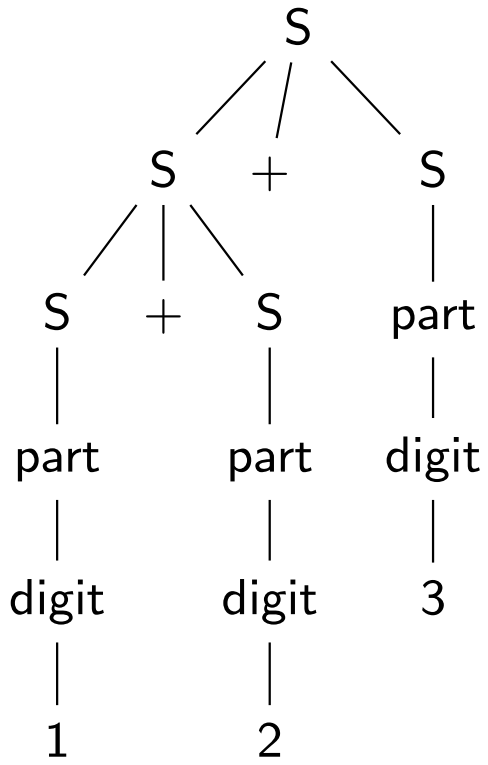
$\texttt{digit} \longrightarrow 0 \mid 1 \mid \ldots \mid 9$

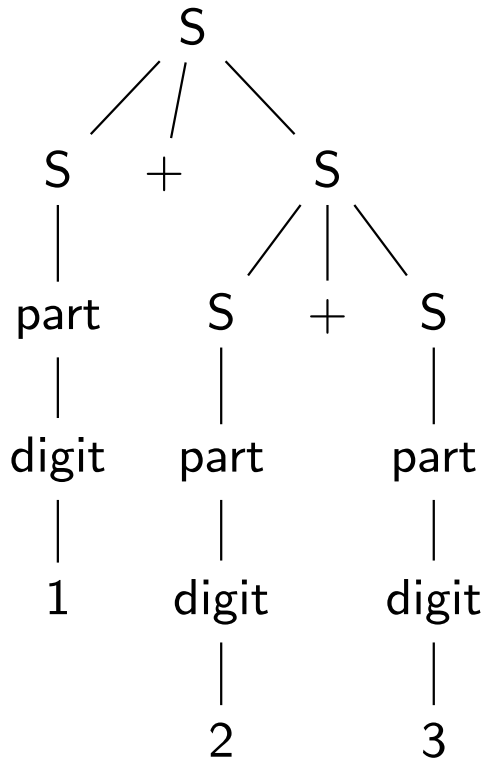
$\texttt{part} \longrightarrow \texttt{digit} \mid \texttt{digitpart}$

$S \longrightarrow \texttt{part} \mid S + S \mid S - S \mid S * S \mid S / S$

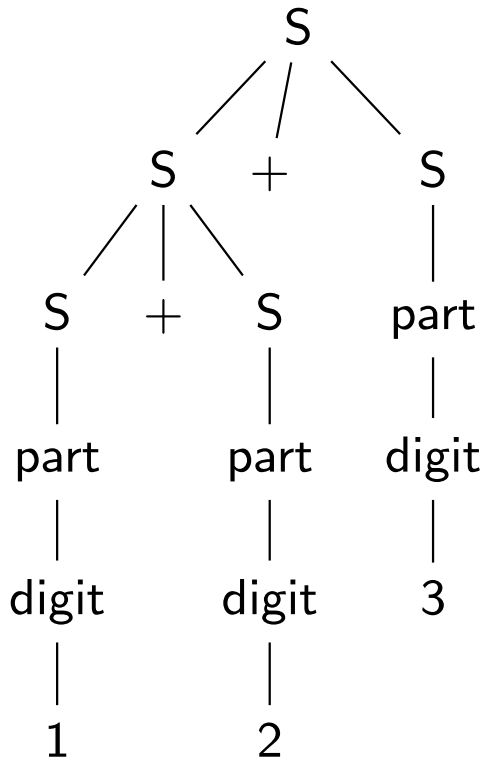
- $1 + 2$
- $1 + 2 + 3$
- $1 + 2$ - 3
- $1 + 2$ * 3

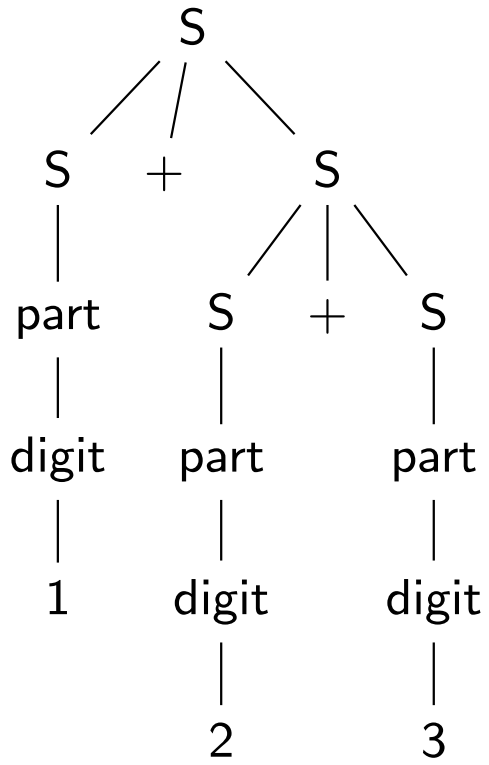
Parse Tree for 1 + 2 + 3





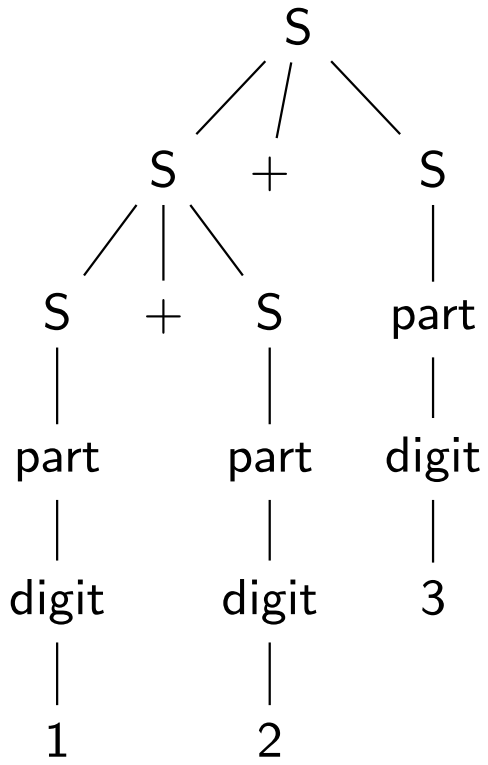
Parse Tree for 1 + 2 + 3



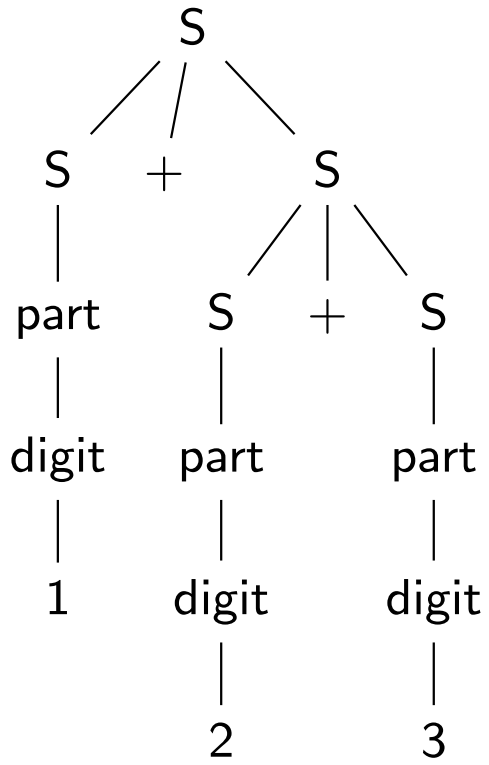


sem(+, sem(+, sem(1), sem(2)), sem(3))

Parse Tree for 1 + 2 + 3

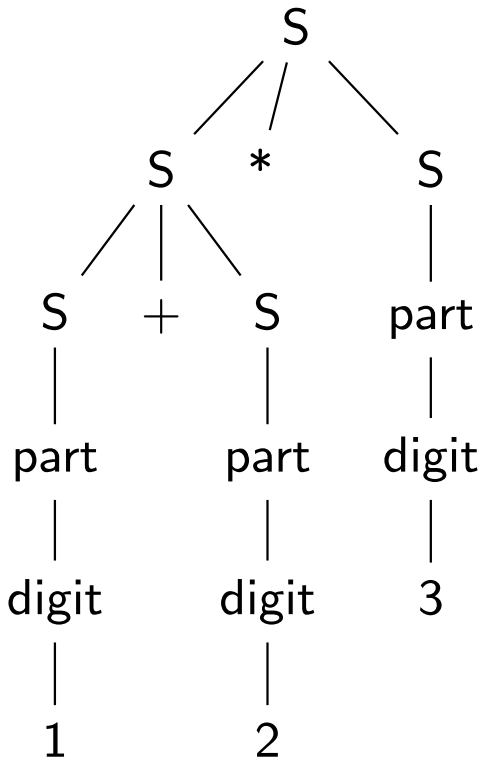


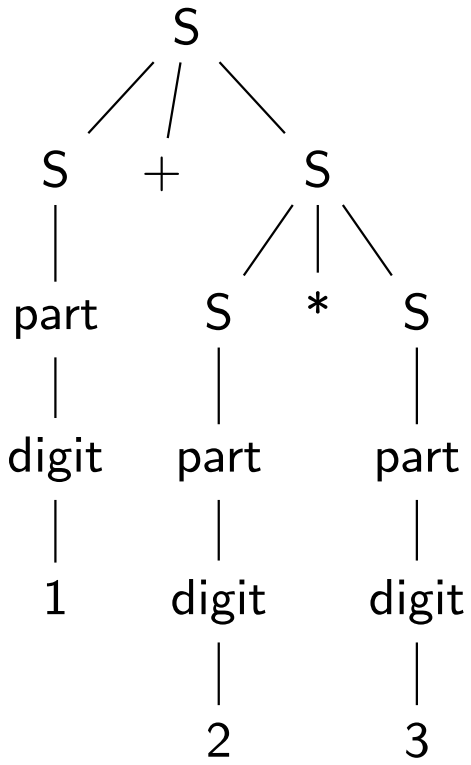
sem(+, sem(+, sem(1), sem(2)), sem(3))



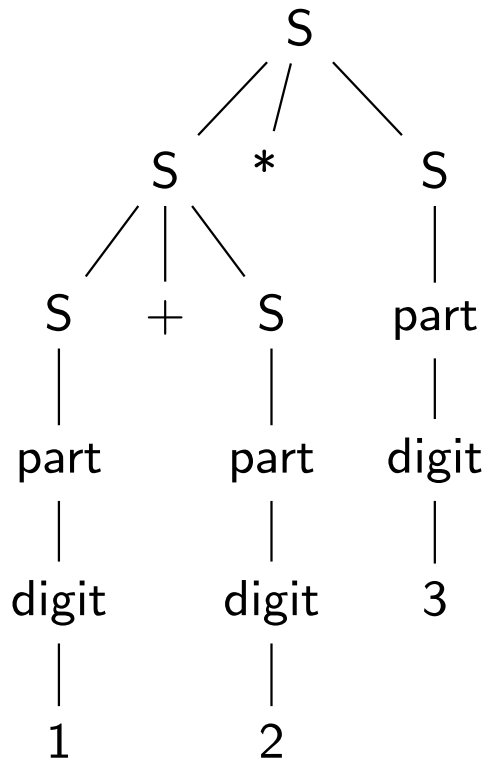
sem(+, sem(1), sem(+, sem(2), sem(3)))

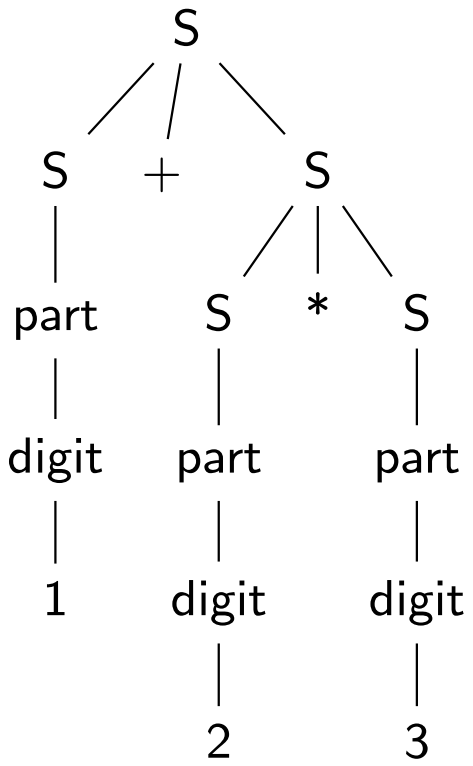
More Critical for 1 + 2 * 3





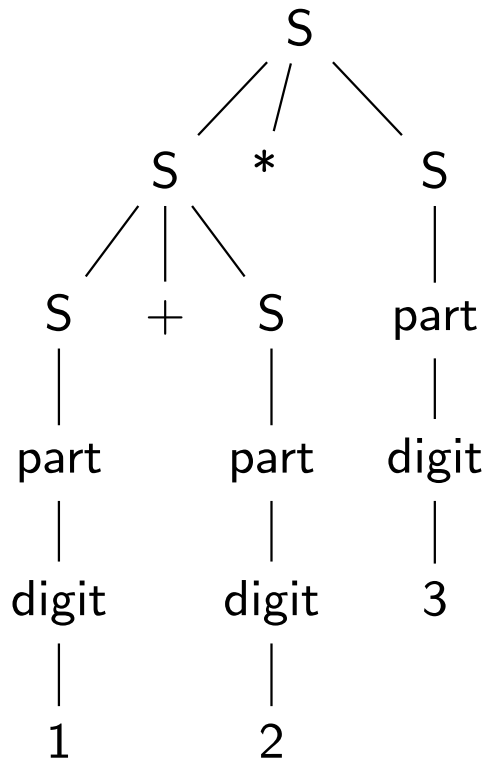
More Critical for 1 + 2 * 3



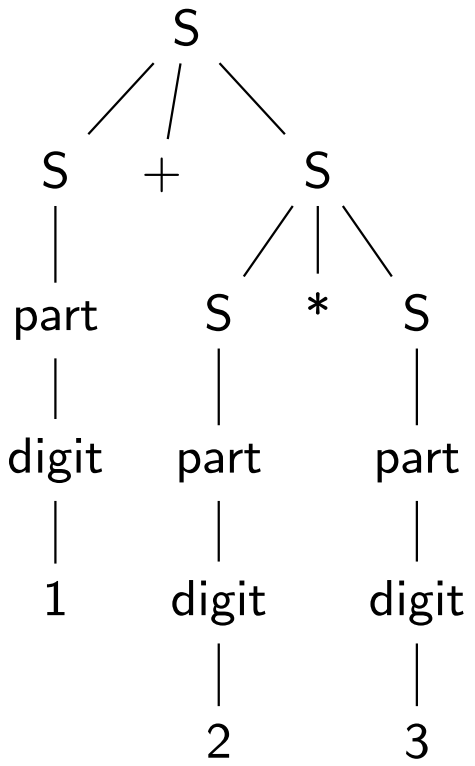


sem(*, sem(+, sem(1), sem(2)), sem(3))

More Critical for 1 + 2 * 3



sem(*, sem(+, sem(1), sem(2)), sem(3))



sem(+, sem(1), sem(*, sem(2), sem(3)))

Ambiguity

Ambiguity in Grammar can result in incorrect application of semantic rules
Ambiguity in Grammar can result in incorrect syntax-directed translation

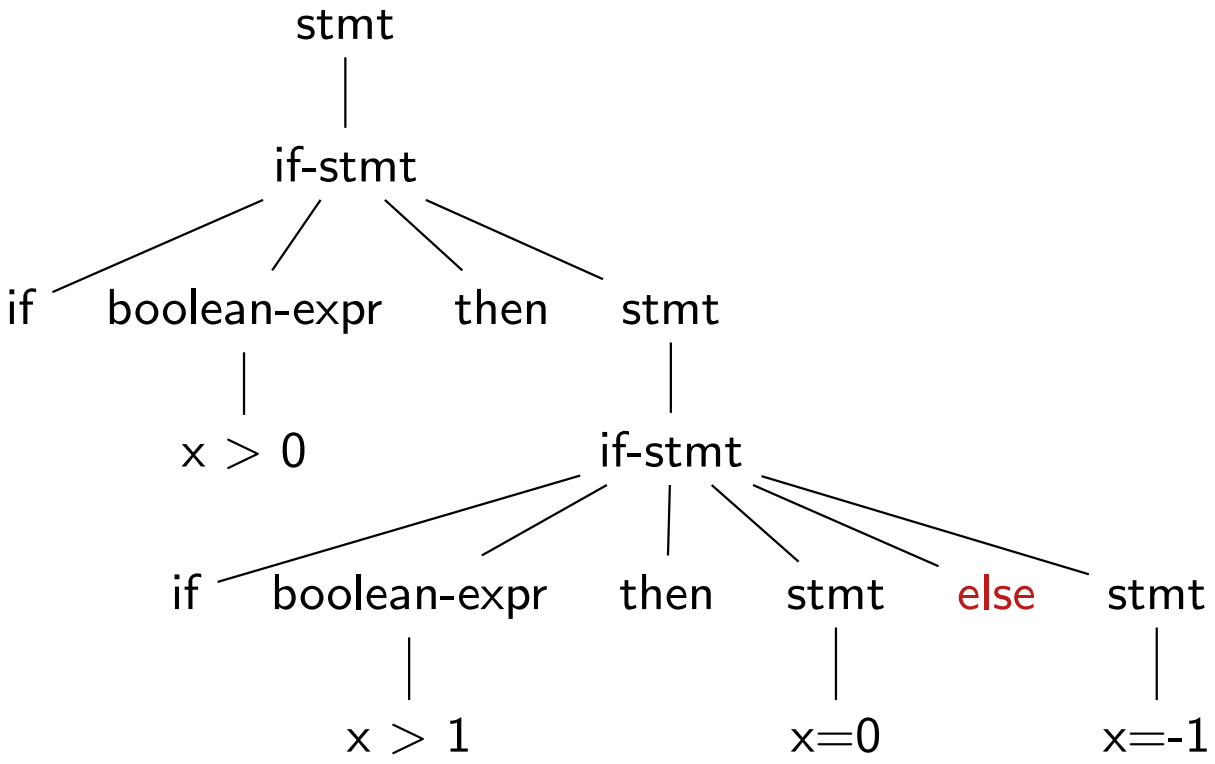
Another example

$$\begin{aligned} \texttt{stmt} &\longrightarrow \texttt{...} \mid \texttt{...} \mid \texttt{if-stmt} \\ \texttt{if-stmt} &\longrightarrow \textbf{if}\ \texttt{boolean-expr}\ \textbf{then}\ \texttt{stmt} \\ &\quad \mid \textbf{if}\ \texttt{boolean-expr}\ \textbf{then}\ \texttt{stmt}\ \textbf{else}\ \texttt{stmt} \end{aligned}$$

if $(x > 0)$ **then**
if $(x > 1)$ **then** $x = 0$
else $x = -1$

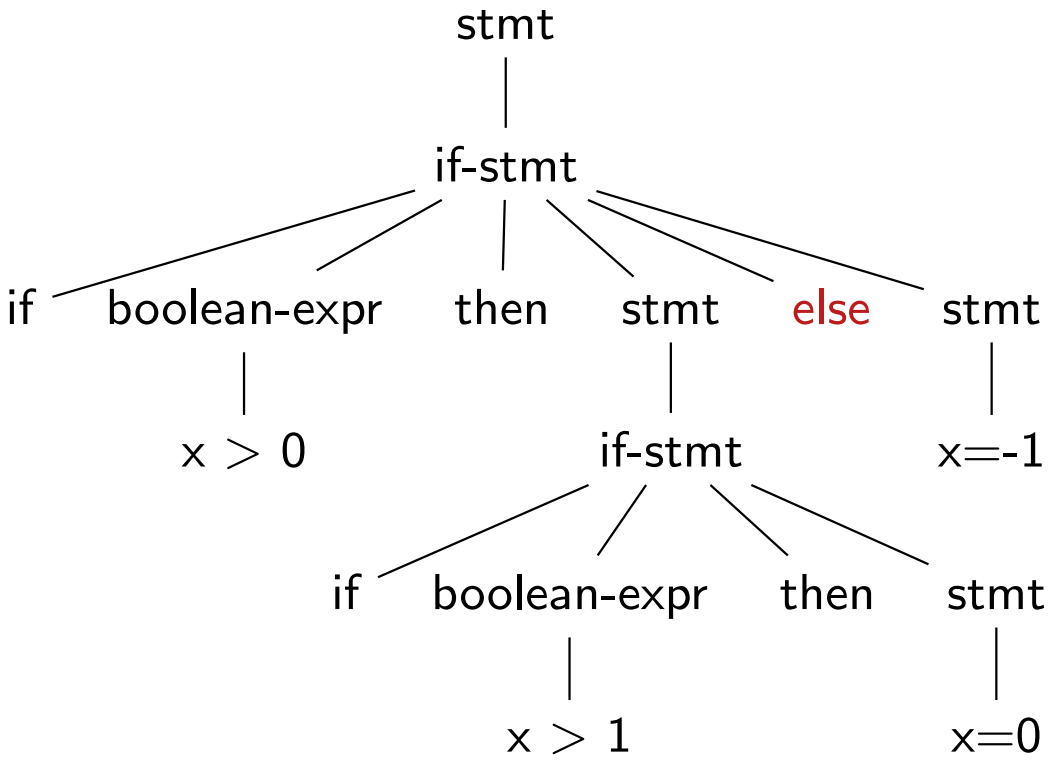
Parse Trees

if $(x > 0)$ **then**
if $(x > 1)$ **then** $x = 0$
else $x = -1$



Parse Trees

if $(x > 0)$ **then**
if $(x > 1)$ **then** $x = 0$
else $x = -1$



Removing Ambiguity

- Add delimiters (e.g., parenthesis; begin and end in if statements)
- Add operator precedence and associativity

Delimiters

$$\texttt{digit} \longrightarrow 0 \mid 1 \mid \ldots \mid 9$$
$$\texttt{part} \longrightarrow \texttt{digit} \mid \texttt{digitpart}$$
$$S \longrightarrow \texttt{part} \mid (S + S) \mid (S - S) \mid (S * S) \mid (S \,/\, S)$$

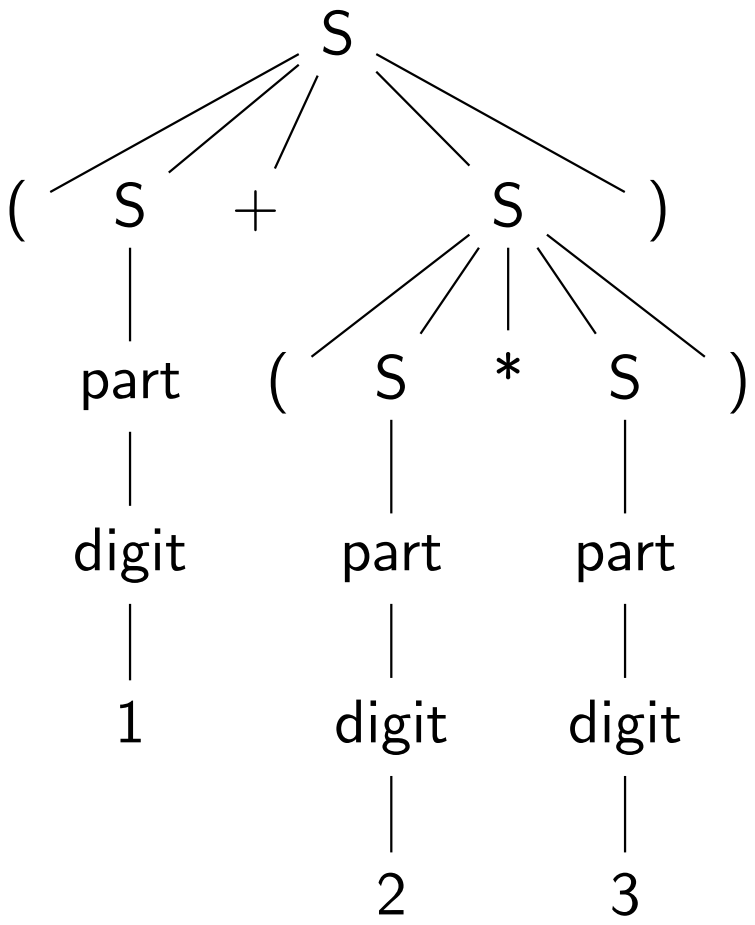
Delimiters

$\texttt{digit} \longrightarrow 0 \mid 1 \mid \ldots \mid 9$

$\texttt{part} \longrightarrow \texttt{digit} \mid \texttt{digitpart}$

$S \longrightarrow \texttt{part} \mid (S + S) \mid (S - S) \mid (S * S) \mid (S / S)$

The parse tree for $(1 + (2 * 3))$



Delimiters

$$
\begin{array}{rcl}
\texttt{stmt} & \longrightarrow & \texttt{...} \mid \texttt{...} \mid \texttt{if-stmt} \\
\texttt{if-stmt} & \longrightarrow & \textbf{if}\ \texttt{boolean-expr}\ \textbf{then begin}\ \texttt{stmt}\ \textbf{end} \\
 & & \mid \textbf{if}\ \texttt{boolean-expr}\ \textbf{then begin}\ \texttt{stmt}\ \textbf{else}\ \texttt{stmt}\ \textbf{end}
\end{array}
$$

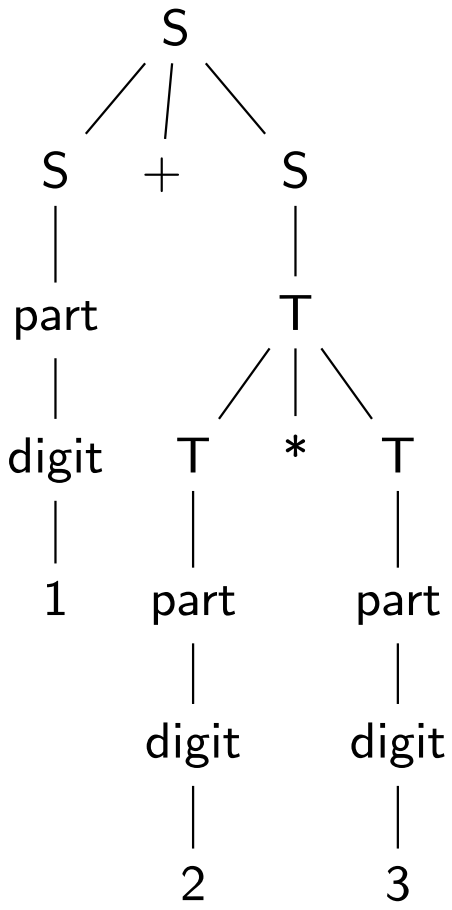
Operator Precedence

If more than one operator is present in the expression, the precedence order decides the order in which the operators should be applied.
Add non-terminals for each precedence level. Push the higher levels towards the bottom of the parse-tree

$$S \longrightarrow S + S \mid S - S \mid T$$
$$T \longrightarrow T * T \mid T / T \mid \texttt{part}$$

Parse Tree for 1 + 2 * 3

$$
\begin{array}{lcl}
S & \longrightarrow & S + S \mid S - S \mid T \\
T & \longrightarrow & T * T \mid T / T \mid \texttt{part}
\end{array}
$$



Associativity

If the same operator appears more than once in the same expression, then associativity rule decides the order in which the operators should be applied.

There are two types of associativity: left and right.

- Left associativity: operators on the left are applied before the operators on the right.
- Right associativity: operators on the right are applied before the operators on the left.

Examples: $x/y/z$ becomes $(x/y)/z$ is $/$ is left associative.

Associativity in Grammar Rules

Assume that $+$ is left-associative. Allow expansion only on the left-hand side.

Associativity in Grammar Rules

Assume that $+$ is left-associative. Allow expansion only on the left-hand side.

$$\begin{array}{lcl} S & \longrightarrow & S + T \mid S - S \mid T \\ T & \longrightarrow & T * T \mid T / T \mid \texttt{part} \end{array}$$

What is the parse tree for $1 + 2 + 3 * 4$

Elimination of Ambiguity

- In general, there is no technique that will always convert an ambiguous grammar to a unambiguous one using additional rules.
- Some grammars are left ambiguous for better representation of concepts
- A language is ambiguous if any grammar that generates the strings in the language is ambiguous.

More on all these in COM S 331 and COM S 440.

Review and Further Reading

- Grammar: is it a syntactic valid string for the language
- CFG: context free grammar
- Derivation: Left-derivation and Right-derivation
- Parse tree: traverse parse tree to evaluate semantics
- Ambiguity
- Precedence, Associativity

Further reading:

- Sebesta: Ch 3

Exercise

- Update the arithmetic expression grammar (in the previous slide) to incorporate left-associativity of $*$ and / operators.
- Does the left-derivation or right-derivation strategies impact the ambiguity of the grammar? If the grammar is unambiguous, do the two derivation strategies result in the same sequence of production rules?
- In postfix notation, the operator appears after the operands. For instance, postfix of $1 + 2$ is $1\ 2\ +$. Write the postfix expression for $1 + 2 + 3 * 4$, where $+$ left associative and $*$ has higher precedence than $+$.
- Any postfix operator is left associative. Write the corresponding infix expression for the following postfix expression (use parenthesis):

$$3\ 2\ *\ 4\ A\ B\ +\ *\ +$$
$$2\ 3\ 4\ +\ *\ 5\ *$$