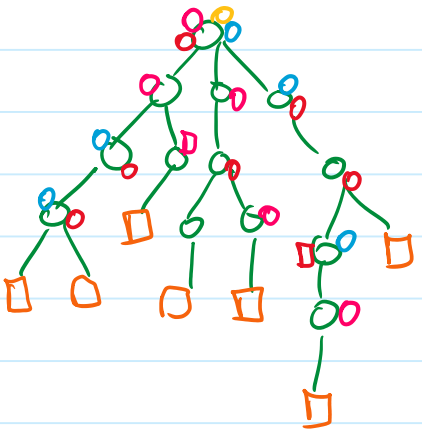
16 Operational Semantics

Wednesday, November 13, 2024 12:09 PM

- Main idea:
  - Let's attach semantic information to our parse tree
  - extra information on the intermediate nodes



- ATTRIBUTE GRAMMAR:
  - expand parse tree with semantic info:
    - D. Knuth and D. Wagner

Idea:

- each node will have "attributes"
- for each grammar symbol X
  a set of attributes Att(X)
- for each grammar rule R
  a collection of "attribute" rules
  that assign values to attributes of symbols in R

E.G. #1

Grammar symbols A B C

attributes x y

Grammar Rule:

$A \to BC$

Attribute Rules:

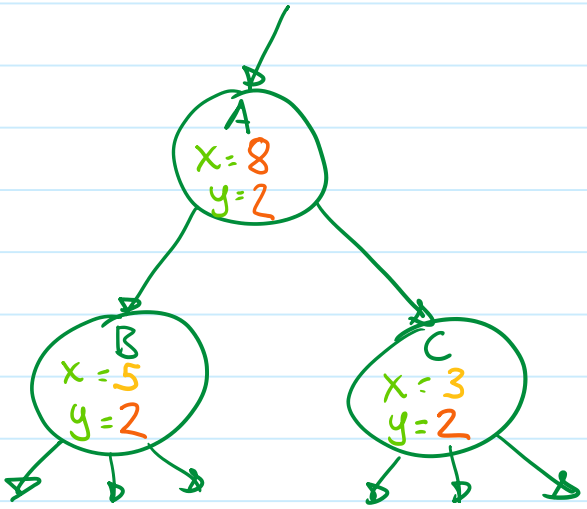
$A.y \leftarrow 2$

$A.x \leftarrow B.x + C.x$

$B.y \leftarrow A.y$

$C.y \leftarrow B.y$

/



$B.y \leftarrow A.y$
$C.y \leftarrow B.y$

Synthesized:
$A.x$ $A.y$

Inherited:
$B.y$ $C.y$

- Types of Attributes:
  - "Synthesized" Attributes
    - value depends on the attribute values of a node's children
    - info flows from the bottom to the top.
  - "Inherited" Attributes
    - value depends on the attribute values of a node's parents or syblings.
    - info flows from top-to-bottom or sideways.

- E.G. #2

attributes type exptype.

$A \rightarrow \underline{var} :- E$ $\quad E.exptype \leftarrow var.type$

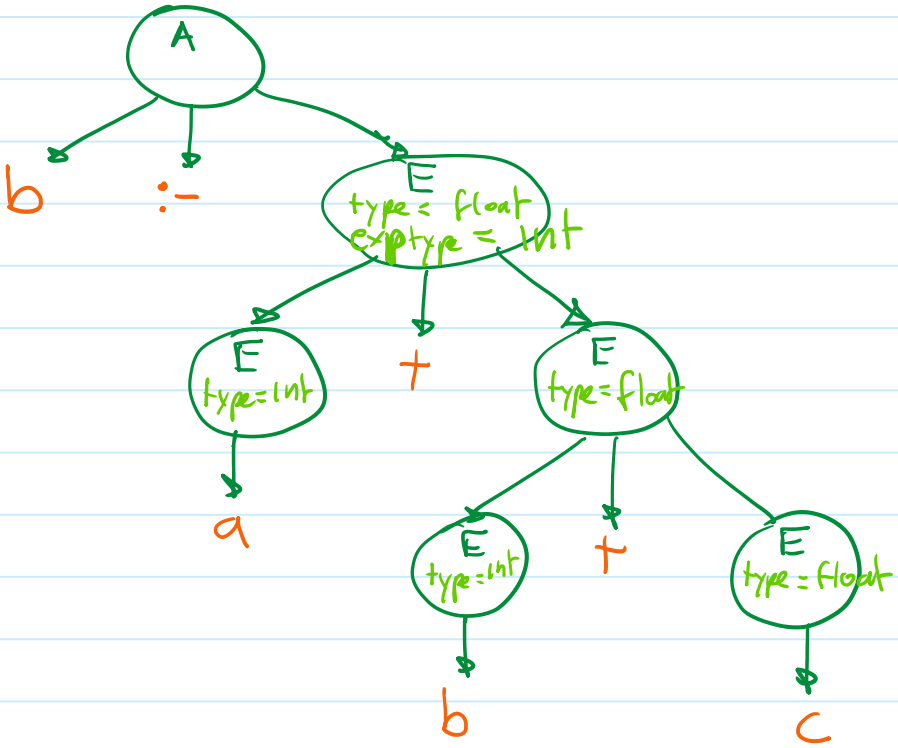
$E_0 \rightarrow E_1 + E_2$ $\quad E_0.type = \begin{cases} int & \text{if } E_1.type = int \text{ AND } E_2.type = int \\ float & \text{otherwise} \end{cases}$

$E \rightarrow \underline{var}$ $\quad E.type \leftarrow var.type$

Symbol table:

a	int
b	int
c	float

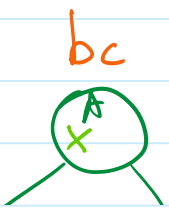
b :- a + b + c

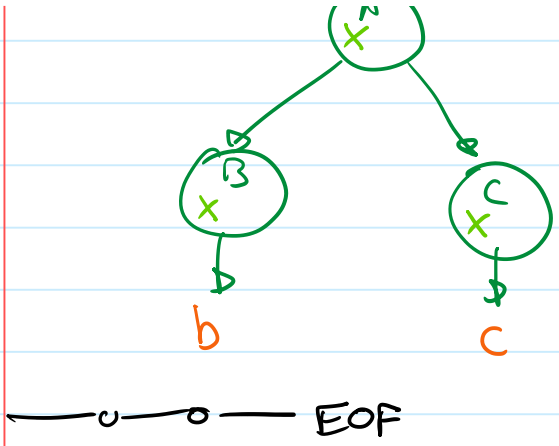


- COMPUTING ATTRIBUTE VALUES:
  - Synthesized Attributes
    Traverse bottom up
  - Inherited
    Traverse top-to-bottom
  - Both kinds
    Multiple traversals bottom-up & top-down

E.G: Degenerate attribute grammar

$A \rightarrow B\ C$	$A.x \leftarrow C.x$
$B \rightarrow b$	$B.x \leftarrow A.x$
$C \rightarrow c$	$C.x \leftarrow B.x$





EOF