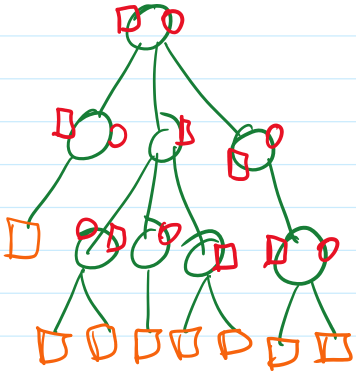
16 Operational Semantics : Attribute Grammars

Wednesday, April 17, 2024 12:32 PM

- Main idea:

  What if we could attach semantic information to our parse tree.

  extra information on the intermidiate nodes.



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

  Idea:
  - attach to each node a collection of attributes.
  - for each grammar symbol X a set of attributes Att(X)
  - for each grammar rule R a collection of rules: attribute rules that assign values to attributes of symbols in R.

E.G. #1

attributes x y

Grammar Rule: . . . . Attribute rules:

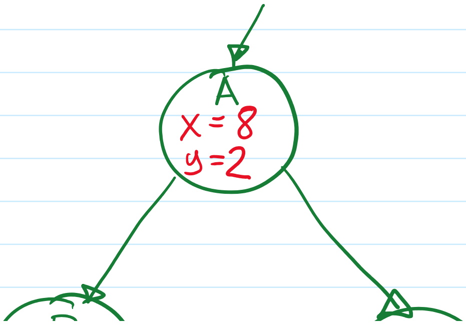
$A \to B\,C$

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

$A.y \Leftarrow 2$

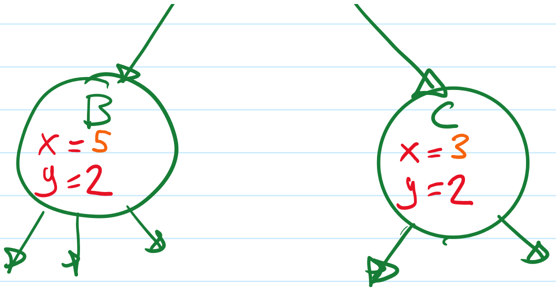
$B.y \leftarrow A.y$

$C.y \leftarrow B.y$



Synthesized:
A.x Ay

Inherited:



Inherited:

B.y
C.y

- Types of attributes:
  - "Synthesized" Attributes:
    - valve depends on the attribute values of a node's children.
    - info flows from bottom to top of the tree.
  - "Inherited" Attributes
    - value depends on the values of attributes of node's siblings or parents
    - info flows $\left\{ \begin{array}{l} \text{- from top to bottom} \\ \text{- Sideways} \end{array} \right\}$ of the tree

E.G. #2   attributes: type exptype.

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

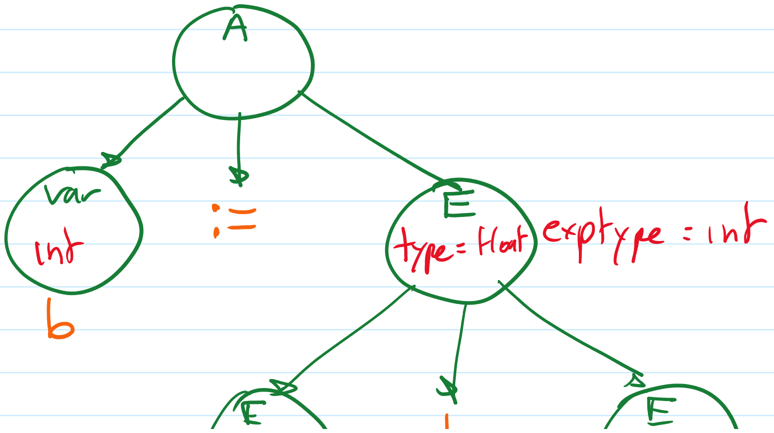
$E_0 \rightarrow E_1 + E_2$ $E_0.type \begin{cases} int & \text{if } E_1.type = INT \text{ AND } E_2.type = INT \\ float & \text{otherwize.} \end{cases}$

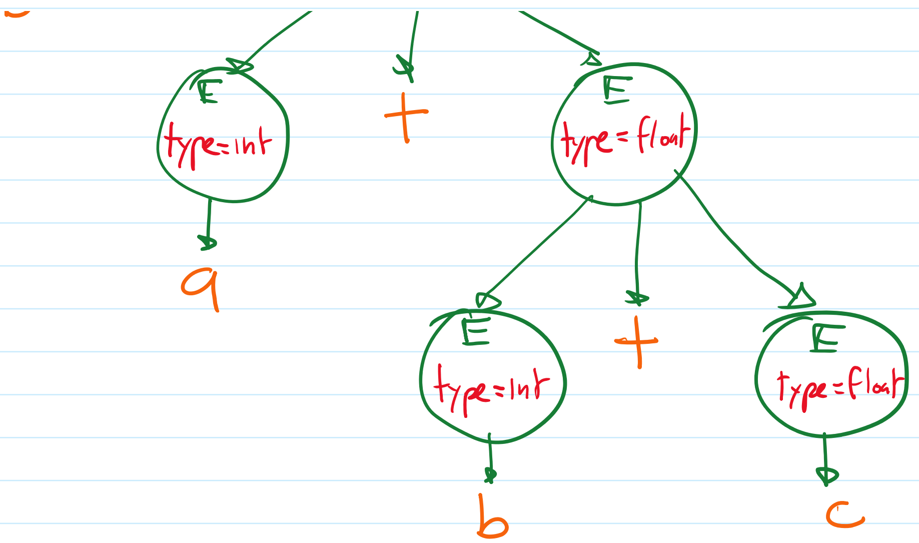
$E \rightarrow \underline{var}$ $E.type \leftarrow var.type.$

b := a + b + c

Symbol table

a	int
b	int
c	float



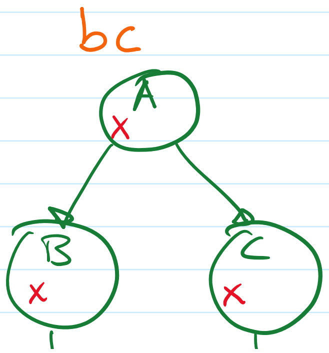


COMPUTING ATTRIBUTE VALUES:

- Synthesized Attributes.
  - traverse the tree bottom up.
- Inherited Attributes
  - traverse the tree top to Bottom
- Both kinds:
  - Multiple travesals bottom-up & top-down

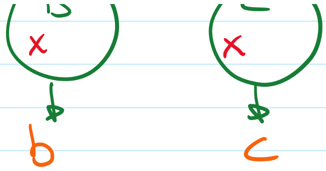
E.G: Degenerate attribute grammar:

	Attributes x
A → BC	A.x ← C.x
B → b	B.x ← A.x
C → c	C.x ← B.x



Synthesized
A.x

Inherited.
B.x C.x



Inherited.

B.x C.x