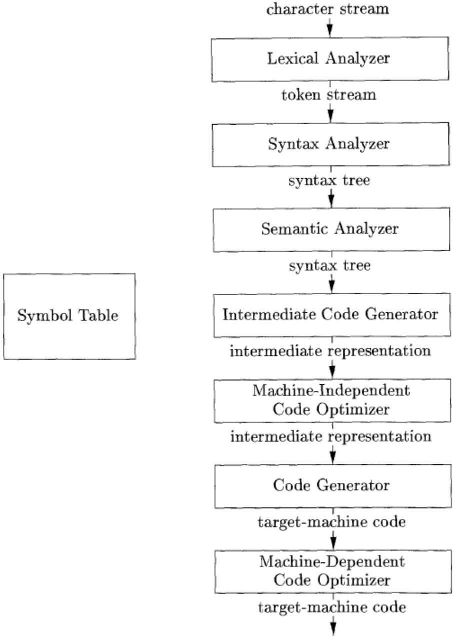
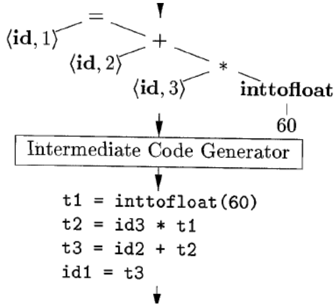
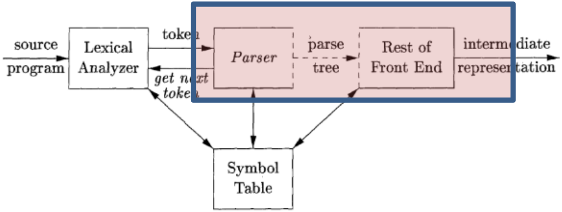
Syntax-Directed Translation

The Phases of a Compiler



Syntax-Directed Translation





- **Semantic analysis** and **translation** actions can be **interlinked** with parsing
- Implemented as a **single module**.

Syntax-Directed Translation

- **Translation** of languages **guided** by **context-free grammars**.

- Attach ***attributes*** to the **grammar symbol**

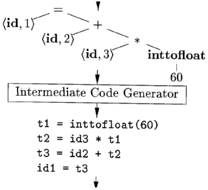
- **Syntax-directed definition** specifies the **values of attributes**
  - By associating **semantic rules** with the **grammar productions**

Syntax-Directed Translation

- *Syntax-directed definition* (SDD) is a **context-free grammar** together with **attributes and rules**
    - Attributes are associated with grammar symbols
    - Rules are associated with productions.

- If ***X*** is a **grammar symbol** and ***a*** is one of its **attributes**,
    - ***X.a*** denotes the value of the attribute *X*.

- Attributes may be
    - numbers, types, table references, or strings,
    - Strings may even be code in the intermediate language.



Attributes

Synthesized attribute:

- *Synthesized attribute* for a **nonterminal *A*** at a parse-tree node *N* is defined by
- **Semantic rule** associated with the **production at *N*.**
- The production must have ***A* as its head**.

- A synthesized attribute at node *N* is defined only in terms of attribute values at the **children of *N* and at *N* itself**.

PRODUCTION	SEMANTIC RULE
$E \rightarrow E_1 + T$	$E.code = E_1.code \parallel T.code \parallel '+'$

Attributes

Inherited attribute:

- Inherited attribute for a **nonterminal B** at a parse-tree node N is defined by

- **Semantic rule** associated with the **production at the parent of N**

- Note that the production must have **B as a symbol in its body**.

- An inherited attribute at node N is defined only in terms of **attribute values at N's parent, N itself, and N's siblings**

$T \rightarrow F\ T'$	$T'.inh = F.val$
$T' \rightarrow *\ F\ T'_1$	$T'_1.inh = T'.inh \times F.val$

Attributes

- **Synthesized attribute** at node N to be **defined** in terms of **inherited attribute** values at node **N itself**.

$$T' \to \epsilon \qquad \Big| \; T'.syn = T'.inh$$

- **Do not allow** an **inherited attribute** at node *N* to be defined in terms of attribute values at the **children of node *N***

- **Terminals** can have **synthesized attributes**, but not inherited attributes.
- **Attributes for terminals** have **lexical values** that are supplied by the **lexical analyzer**

$$F \to \mathbf{digit} \qquad \Big| \; F.val = \mathbf{digit}.\text{lexval}$$

Example of SDD

Each of the Non-terminals has a **single synthesized attribute**, called ***val***

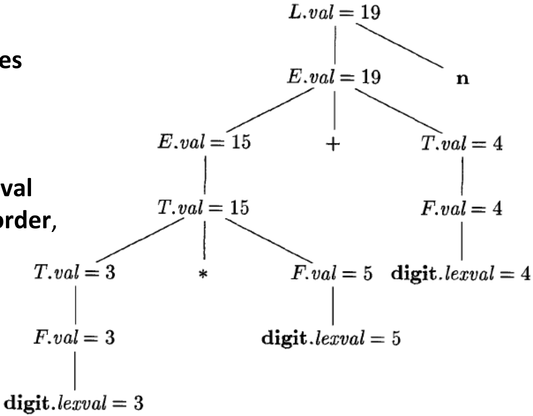
	Production	Semantic Rules
1)	$L \rightarrow E\ \mathbf{n}$	$L.val = E.val$
2)	$E \rightarrow E_1\ +\ T$	$E.val = E_1.val + T.val$
3)	$E \rightarrow T$	$E.val = T.val$
4)	$T \rightarrow T_1\ *\ F$	$T.val = T_1.val \times F.val$
5)	$T \rightarrow F$	$T.val = F.val$
6)	$F \rightarrow (\ E\ )$	$F.val = E.val$
7)	$F \rightarrow \mathbf{digit}$	$F.val = \mathbf{digit}.\text{lexval}$

Annotated parse tree.

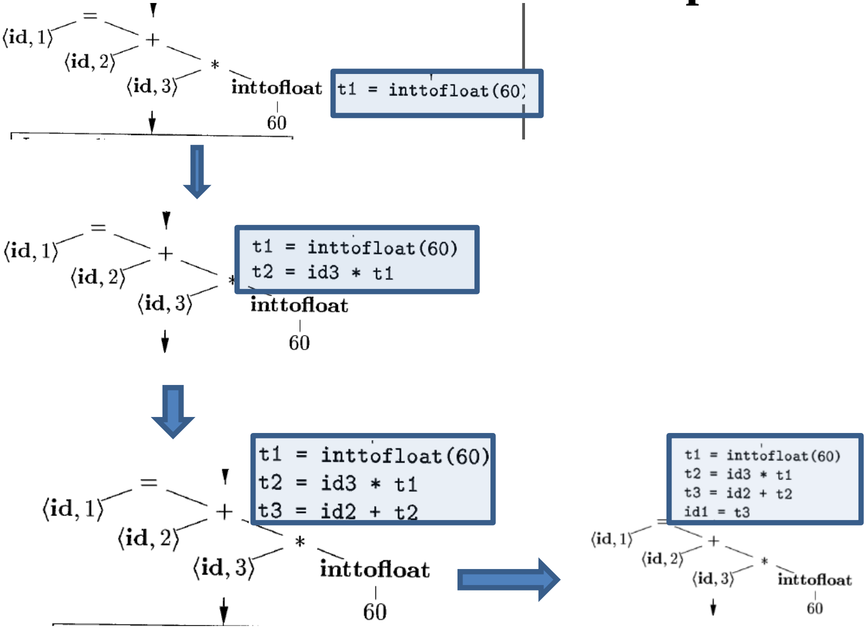
A **parse tree**, showing the **value(s) of its attribute(s)** is called an ***annotated*** *parse tree.*

Input string: **3 * 5 + 4 n**

- We show the resulting **values associated** with **each node**.
- Each of the nodes for the nonterminals has **attribute val** computed in a **bottom-up order**,



Annotation and Evaluation of parse tree

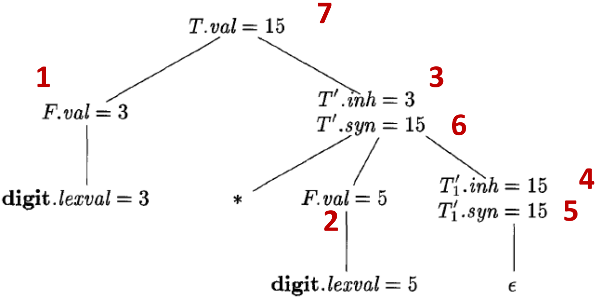


Annotated parse tree.

	Production	Semantic Rules
1)	$T \rightarrow F\,T'$	$T'.inh = F.val$ $T.val = T'.syn$
2)	$T' \rightarrow *\,F\,T'_1$	$T'_1.inh = T'.inh \times F.val$ $T'.syn = T'_1.syn$
3)	$T' \rightarrow \epsilon$	$T'.syn = T'.inh$
4)	$F \rightarrow \textbf{digit}$	$F.val = \textbf{digit}.lexval$

val and **syn**: Synthesized
inh: Inherited

Annotated parse tree for 3 * 5



Evaluation Orders of SDD

- "**Dependency graphs**" are a useful tool for determining an **evaluation order** for the **attribute** instances in a given parse tree.
  - Depicts the **flow of information** among the attribute instances in a particular parse tree
  - **Directed edges**

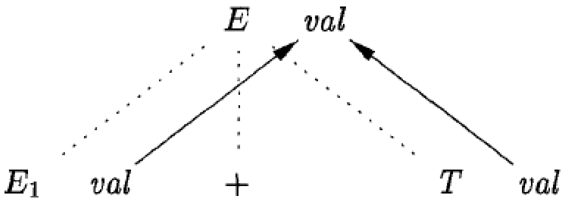
- For a **node A in parse tree** -> **node A in dependency graph**

A has a **synthesized** attribute **b**

Production	**Semantic Rule**
A->…X..	A.b=f(.., X.c, ..)

- **Edge** from X.c to A.b
  - Edge from **child attribute** to **parent attribute**

PRODUCTION	SEMANTIC RULE
$E \to E_1 + T$	$E.val = E_1.val + T.val$



Evaluation Orders of SDD

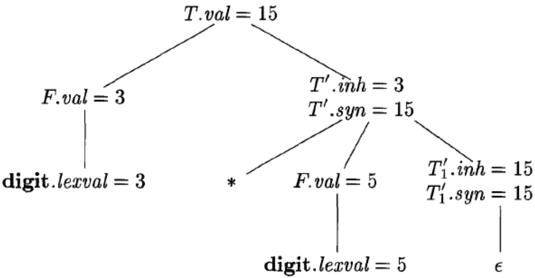
- "Dependency graphs" are a useful tool for determining an **evaluation order** for the attribute instances in a given parse tree.
    - Depicts the flow of information among the attribute instances in a particular parse tree
    - Directed edges
- For a **node A in parse tree** -> **node A in dependency graph**

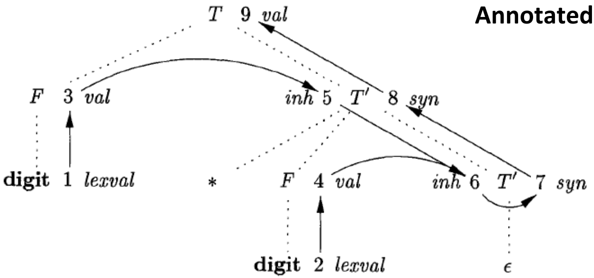
B has an inherited attribute **c**

Production	**Semantic Rule**
A->...B..X..	B.c=f(.., X.a, ..)

- **Edge** from **X.a to B.c**
    - Edge from **attribute a of X** (parent or sibling of B) to **attribute c of B** (body of the production)

	Production	Semantic Rules
1)	$T \to F\,T'$	$T'.inh = F.val$ $T.val = T'.syn$
2)	$T' \to *\,F\,T'_1$	$T'_1.inh = T'.inh \times F.val$ $T'.syn = T'_1.syn$
3)	$T' \to \epsilon$	$T'.syn = T'.inh$
4)	$F \to \mathbf{digit}$	$F.val = \mathbf{digit}.lexval$





Annotated parse tree for 3 * 5

Ordering the Evaluation of Attributes

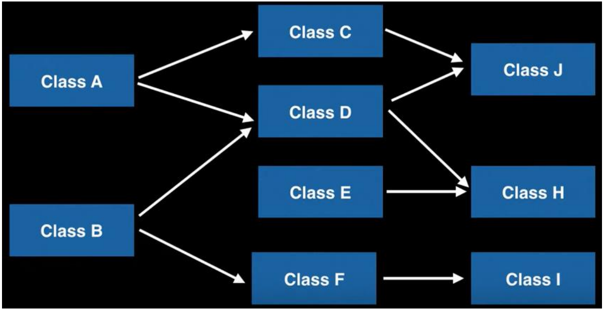
- The **dependency graph** characterizes the possible **evaluation orders**
  - In which we can **evaluate the attributes** at the various nodes of a parse tree.

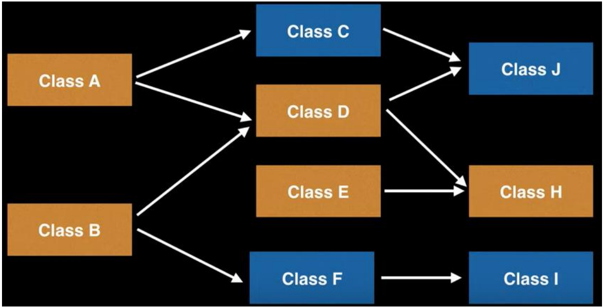
- If the dependency graph has an **edge from node M to node N**,
  - Attribute corresponding to **M must be evaluated before** the attribute of N.

- If there is an edge of the dependency graph from **Ni to Nj, such that i < j**
  - the only allowable orders of evaluation are those sequences of nodes **N1, N2,... ,Nk**

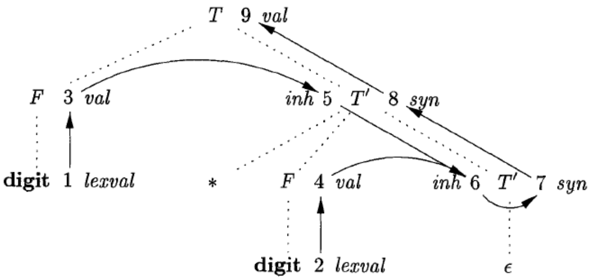
- Embeds a directed graph into a linear order, and is called a **topological sort** of the graph

Topological Sort





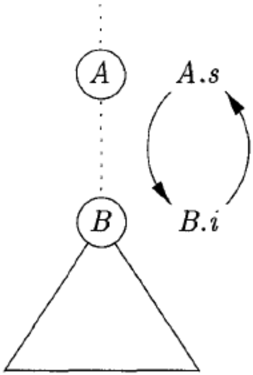
Topological Sort- Ordering the Evaluation



- One **topological sort** is the order in which the nodes have already been numbered: 1,2,... ,9.
- There are other topological sorts as well, such as 1,3,5,2,4,6,7,8,9.

Ordering the Evaluation – Cycles

PRODUCTION	SEMANTIC RULES
$A \to B$	$A.s = B.i;$ $B.i = A.s + 1$



These rules are circular; it is impossible to evaluate either *A.s* or *B.i*

Classes of SDD

(a) S-Attributed Definitions
(b) L-Attributed Definitions

Guarantee an evaluation order

S-Attributed SDD

An SDD is *S-attributed* if **every attribute is synthesized**.

	Production	Semantic Rules
1)	$L \to E\ \mathbf{n}$	$L.val = E.val$
2)	$E \to E_1 + T$	$E.val = E_1.val + T.val$
3)	$E \to T$	$E.val = T.val$
4)	$T \to T_1 * F$	$T.val = T_1.val \times F.val$
5)	$T \to F$	$T.val = F.val$
6)	$F \to (E)$	$F.val = E.val$
7)	$F \to \mathbf{digit}$	$F.val = \mathbf{digit}.\mathrm{lexval}$

S-Attributed SDD

An SDD is *S-attributed* if **every attribute is synthesized**.

When an SDD is S-attributed, we can evaluate its attributes in any bottom-up order of the nodes of the parse tree. It is often especially simple to evaluate the attributes by performing a postorder traversal of the parse tree and evaluating the attributes at a node N when the traversal leaves N for the last time.

postorder(N) {
    for (each child C of N, from the left) *postorder*(C);
    evaluate the attributes associated with node N;
}

L-Attributed SDD

- The idea behind L-attributed SDD class is that,
  - **Between the attributes** associated with a **production body**, **dependency-graph edges can go from left to right**,
  - But not from right to left (hence "L-attributed")

1. Synthesized, or

2. Inherited, but with the rules limited as follows. Suppose that there is a production $A \rightarrow X_1 X_2 \cdots X_n$, and that there is an inherited attribute $X_i.a$ computed by a rule associated with this production. Then the rule may use only:

   (a) Inherited attributes associated with the head A.

   (b) Either inherited or synthesized attributes associated with the occurrences of symbols $X_1, X_2, \ldots, X_{i-1}$ located to the left of X_i.

   (c) Inherited or synthesized attributes associated with this occurrence of X_i itself, but only in such a way that there are no cycles in a dependency graph formed by the attributes of this X_i.

L-Attributed SDD

	Production	Semantic Rules
1)	$T \to F\ T'$	$T'.inh = F.val$ $T.val = T'.syn$
2)	$T' \to *\ F\ T'_1$	$T'_1.inh = T'.inh \times F.val$ $T'.syn = T'_1.syn$
3)	$T' \to \epsilon$	$T'.syn = T'.inh$
4)	$F \to \textbf{digit}$	$F.val = \textbf{digit}.lexval$

Production	Semantic Rules
$A \to B\ C$	$A.s = B.b;$ $B.i = f(C.c, A.s)$