



COMPILER DESIGN

SUBJECT CODE: 303105349

Vishal Singh , Assistant Professor
Computer Science & Engineering





CHAPTER-4

Syntax-directed definitions





Contents

- Syntax Directed Definitions
- Evaluation Orders of SDD's
- Dependency graph
- S-attributed and L-attributed
- Recursive-descent Parser





Syntax Directed Definitions

- We can associate information with a language construct by attaching attributes to the grammar symbols.
- A syntax directed definition specifies the values of attributes by associating semantic rules with the grammar productions.

Production

$E \rightarrow E_1 + T$

Semantic Rule

$E.\text{code} = E_1.\text{code} = T.\text{code}$





Syntax Directed Definitions

- A SDD is a context free grammar with attributes and rules.
- Attributes are associated with grammar symbols and rules with productions.
- Attributes may be of many kinds: numbers, types, table references, strings, etc.





Syntax Directed Definitions

Synthesized attributes:

A synthesized attribute at node N is defined only in terms of attribute values of children of N and at N it.

Inherited attributes:

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





Synthesized attributes

- A synthesized attribute at node N is defined only in terms of attribute values of children of N and at N it.
- Each of the non-terminals has a single synthesized attribute, called val.
- An SDD that involves only synthesized attributes is called S-attributed.
- Each rule computes an attribute for the non-terminal at the head of a production from attributes taken from the body of the production.





SDD for expression grammar with Synthesized attribute

Production

- 1) $L \rightarrow E\ n$
- 2) $E \rightarrow E_1 + T$
- 3) $E \rightarrow T$
- 4) $T \rightarrow T_1 * F$
- 5) $T \rightarrow F$
- 6) $F \rightarrow (E)$
- 7) $F \rightarrow \text{digit}$

Semantic Rules

- $L.\text{val} = E.\text{val}$
 $E.\text{val} = E_1.\text{val} + T.\text{val}$
 $E.\text{val} = T.\text{val}$
 $T.\text{val} = T_1.\text{val} * F.\text{val}$
 $T.\text{val} = F.\text{val}$
 $F.\text{val} = E.\text{val}$
 $F.\text{val} = \text{digit.lexval}$





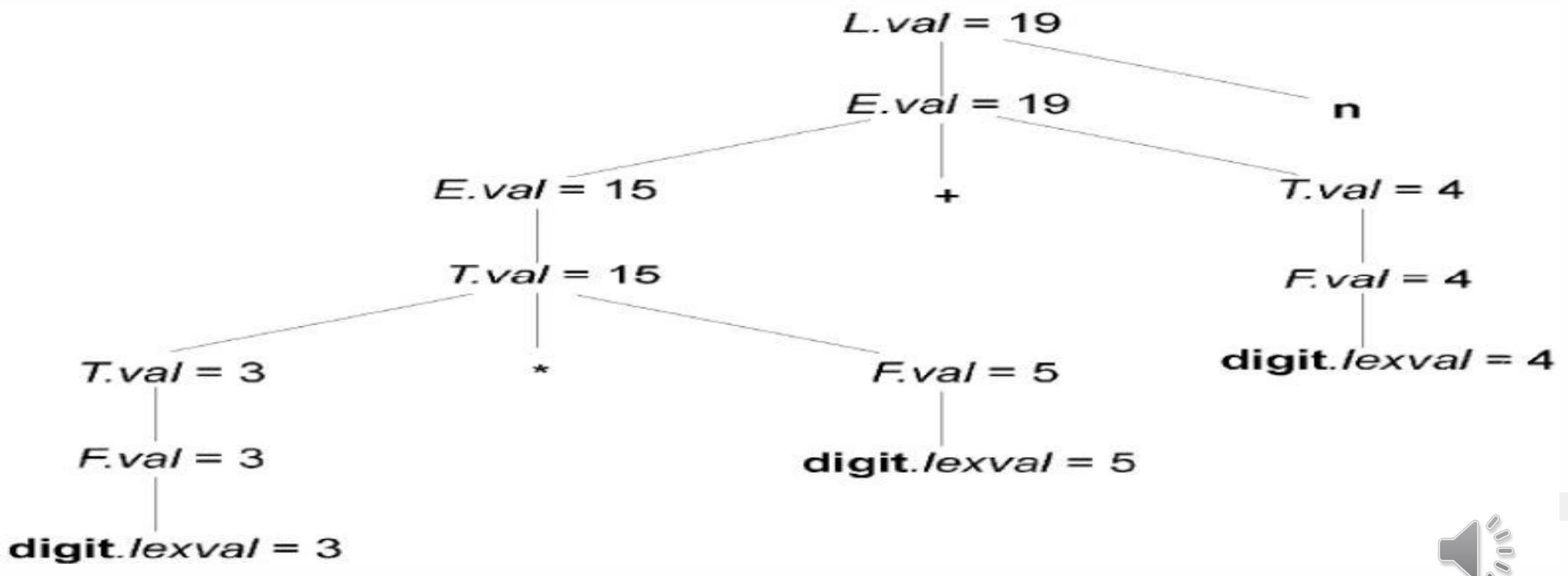
Evaluating an SDD at the Nodes of a Parse Tree

- A parse tree, showing the value(s) of its attribute(s) is called an annotated parse tree.
- With synthesized attributes, evaluate attributes in bottom-up order.





Annotated Parse Tree for $3 * 5 + 4$





Inherited Attributes

- An INHERITED ATTRIBUTE for a non-terminal B at a parse-tree node N is defined by a semantic rule associated with the production at the parent of N. 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





SDD for expression grammar with inherited grammar

PRODUCTION	SEMANTIC RULE
$D \rightarrow TL$	$L.in := T.type$
$T \rightarrow \text{int}$	$T.type := \text{integer}$
$T \rightarrow \text{real}$	$T.type := \text{real}$
$L \rightarrow L_1, id$	$L_1.in := L.in; addtype(id.entry, L.in)$
$L \rightarrow id$	$addtype(id.entry, L.in)$





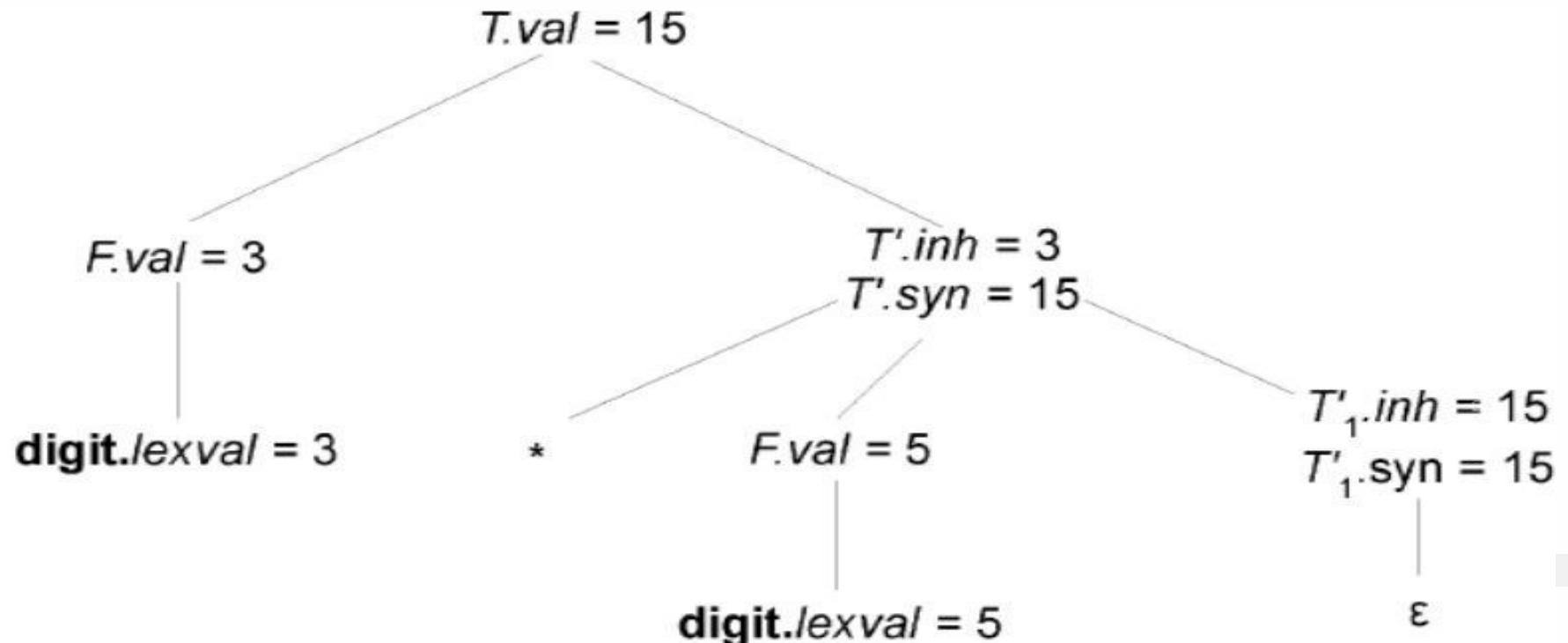
SDD for expression grammar with inherited grammar

Production	Semantic Rules
$T \rightarrow F T'$	$T'.inh = F.val$ $T.val = T'.syn$
$T' \rightarrow *F T'_1$	$T'_1.inh = T'.inh \times F.val$ $T'.syn = T'_1.syn$
$T' \rightarrow \epsilon$	$T'.syn = T'.inh$
$F \rightarrow \text{digit}$	$F.val = \text{digit.lexval}$





Annotated Parse Tree for $3 * 5$





Evaluation Orders for SDD's

- Dependency graphs" tool for determining an evaluation order for the attribute instances in a given parse tree.
- annotated parse tree shows the values of attributes, a dependency graph helps to determine how those values can be computed.





Dependency graphs

- Edges express constraints implied by the semantic rules.
- Each attribute is associated to a node
- If a semantic rule associated with a production p defines the value of synthesized attribute $A.b$ in terms of the value of $X.c$, then graph has an edge from $X.c$ to $A.b$.
- If a semantic rule associated with a production p defines the value of inherited attribute $B.c$ in terms of value of $X.a$, then graph has an edge from $X.a$ to $B.c$.





Topological Sort

- A dependency graph characterizes the possible order in which we can evaluate the attributes at various nodes of a parse tree.
- If there is an edge from node M to N, then attribute corresponding to M first be evaluated before evaluating N.
- Thus the allowable orders of evaluation are N_1, N_2, \dots, N_k such that if there is an edge from N_i to N_j then $i < j$.





Topological Sort

- Such an ordering embeds a directed graph into a linear order, and is called a topological sort of the graph.
- If there is any cycle in the graph, then there are no topological sorts.





S-Attributed

- If every attribute is synthesized.
- S-attributed SDD can be evaluated in bottom up order of the nodes of the parse tree.





L-Attributed

- Synthesized, or Inherited, but with the rules limited as follows. Suppose that there is a production $A \rightarrow X_1 X_2 \dots X_n$, and that there is an inherited attribute $X_i.a$ computed by a rule associated with this production.
- Inherited attributes associated with the head A .
- Either inherited or synthesized attributes associated with the occurrences of symbols X_1, X_2, \dots, X_{i-1} located to the left of X_i .
- 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 of attributes of





Recursive Descent Parser

- Top-down parsing strategy, for LL(1) grammars.
- One procedure per nonterminal.
- Stack contents embedded in recursive call sequence
- Each procedure “commits” to one production, based on the next input symbol, and the select sets.
- It uses back-tracking. So, it is not useful.

