CS3005D Compiler Design

Winter 2024

Lecture #21

Syntax Directed Translation

Saleena N

CSED NIT Calicut

March 2024

Syntax Directed Translation

Translation guided by syntax

- Associate *attributes* with grammar symbols (each symbol represents a language construct)
- Attach *semantic rules* to grammar productions
- Rules are executed when the production is used during syntax analysis
- Semantic rules to generate Intermediate code, evaluate expression, type checking ...

Example: expression evaluation

PRODUCTION	SEMANTIC RULE
$E \rightarrow E_1 + E_2$	$E.val = E_1.val + E_2.val$

- Specifies the evaluation of an expression based on the values of its subexpressions
- *E.val* denotes the attribute *val* associated with grammar symbol *E*

Syntax Directed Definition (SDD)

Syntax Directed Definition (SDD) - Context Free Grammar together with attributes and rules

- notation for specifying the translation
- attributes associated with each grammar symbol
- rules associated with each production
- attributes computed as per the rules
- terminals have lexical values obtained during lexical analysis

Syntax Directed Definition

Production	Semantic Rules
$E \rightarrow E_1 + E_2$	$E.val = E_1.val + E_2.val$
$E \rightarrow \textbf{num}$	$E.val = \textbf{num}.lexval$

- The attribute *lexval* associated with **num** is assumed to be set during lexical analysis

Annotated Parse Tree

Parse Tree annotated with attribute value at each node

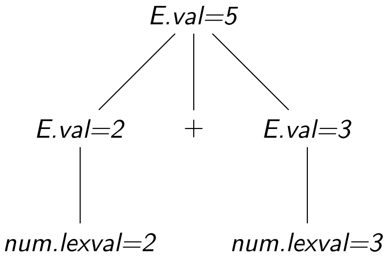

Figure: Annotated Parse Tree for *2+3*

SDD: example

Production	Semantic Rules
$S \rightarrow E$	$print(E.val)$
$E \rightarrow E_1 + E_2$	$E.val = E_1.val + E_2.val$
$E \rightarrow \textbf{num}$	$E.val = \textbf{num}.lexval$

Note the rule *print(E.val)* attached to the first production, for printing the value of the expression

SDD: example

Production	Semantic Rules
$E \rightarrow E_1 + T$	$E.val = E_1.val + T.val$
$E \rightarrow T$	$E.val = T.val$
$T \rightarrow \textbf{num}$	$T.val = \textbf{num}.lexval$

Draw the annotated Parse Tree for 2+3

SDD to construct syntax tree

Production	Semantic Rules
$E \rightarrow E_1 + E_2$	$E.node = CreateNode('+', E_1.node, E_2.node)$
$E \rightarrow \textbf{num}$	$E.node = CreateLeaf(\textbf{num}, \textbf{num}.lexval)$

Draw the Syntax Tree constructed for *1+2*

SDD to construct syntax tree

Production	Semantic Rules
$E \rightarrow E + T$	$E.node = CreateNode('+', E_1.node, T.node)$
$E \rightarrow T$	$E.node = T.node$
$T \rightarrow \textbf{num}$	$T.node = CreateLeaf(\textbf{num}, \textbf{num}.lexval)$

Draw the Syntax Tree constructed for *1+2*

References

References:

- Aho A.V., Lam M.S., Sethi R., and Ullman J.D. Compilers: Principles, Techniques, and Tools (ALSU). Pearson Education, 2007.

Further reading:

- ALSU Chapter 2-sections 2.3, Chapter 5-section 5.1