THEORY OF COMPUTATION AND COMPILERS

Unit - II

CONTEXT FREE GRAMMARS AND PARSING

- Introduction
- Context-Free Grammars - Derivation, Parse trees, Ambiguity
- Types of Parsers
- LL(K) grammars and LL(1) parsing
- Bottom-up Parsing - handle pruning
- **LR Grammar Parsing** 
- LALR parsing
- Parsing ambiguous grammars
- Error Recovery in Parsing
- YACC programming specification

Dr. R. Madana Mohana
Professor, Artificial Intelligence & Data Science | I/c-Head, Artificial Intelligence & Machine Learning
CHAITANYA BHARATHI INSTITUTE OF TECHNOLOGY
Hyderabad - 500 075, Telangana, INDIA
www.cbit.ac.in

Unit-II: Syntax Analysis (or) Parser
Simple LR-Parser (SLR Parser)

Outline:

- Constructing SLR-Parsing Table
- Algorithm for Constructing SLR-Parsing Table
- Example problem

Constructing SLR-Parsing Table

For a given grammar **G**, the **SLR parsing table** can be constructed using the *following steps*:

1. For given grammar **G**, obtain the augmented grammar **G′** .
2. From the augmented grammar **G′** , construct the canonical collection of **LR(0) items** and **LR(0) automaton**.
3. Compute **FIRST** and **FOLLOW** sets for the given grammar **G**.
4. Obtain the parsing table using the algorithm to construct **SLR parsing table**.

Algorithm for Constructing SLR-Parsing Table

Algorithm: **`SLR_PARSING_TABLE (G′)`**

INPUT: An augmented grammar **`G′`**.

OUTPUT: The **`SLR parsing table`** consisting of **`ACTION`** and **`GOTO`**.

Algorithm for Constructing SLR-Parsing Table

METHOD: The procedure is shown below:

1. Obtain **LR(0) items** or **canonical collection of sets of items C** for the **augmented grammar G′**.
2. Let the items be **C = {I_0, I_1, I_2, …, I_n}**. Then **0, 1, 2,.. , n** will be the states of the parser. The parsing **ACTION** for each state **I**, can be obtained as shown below:

Algorithm for Constructing SLR-Parsing Table

METHOD:

2. Let the items be $C = \{I_0, I_1, I_2, \ldots, I_n\}$. Then $0, 1, 2, .., n$ will be the states of the parser. The parsing **ACTION** for each state I, can be obtained as shown below:

   a) If $[A \rightarrow \alpha.a\beta] \epsilon I_i$ and $GOTO\ (I_i, a) = I_j$ then **ACTION** $[i, a] =$ "**shift** j" where 'a' must be a terminal.

Algorithm for Constructing SLR-Parsing Table

METHOD:

b) If **[A → α.]∈I_i**, for each terminal '**a**' in **FOLLOW** (**A**) then **ACTION [i, a] = "reduce A → α"**.

c) If **[S′ → S.]∈I_i**, then **ACTION [i, $] = accept**.

Algorithm for Constructing SLR-Parsing Table

METHOD:

3. For each state **i**, **GOTO** transitions can be obtained as shown below:

   a) if **GOTO (I_i, A)= I_j** then **GOTO [i, A] = j**

      end if

   a) If **I_i** contains **[S′ → S.]** then **'i'** is the initial or start state of the parser.

Algorithm for Constructing SLR-Parsing Table

Note: In **SLR parsing table**, if there are multiple entries in **ACTION**, then the grammar is not **SLR(1)** grammar.

The **parsing table** obtained using this algorithm is **SLR parsing table** and the **parser** which uses this table is called **SLR parser** and the **grammar** is said to be **SLR(1)**.

Constructing SLR-Parsing Table

Example:

Construct **SLR parsing table** for the given grammar **G** as shown below:

1. **E → E + T**
2. **E → T**
3. **T → T * F**
4. **T → F**
5. **F → (E)| id**
6. **F → id**

Constructing SLR-Parsing Table

Example: ***Solution***

Augmented Grammar **G′**:

E′ → E

E → E + T

E → T

T → T * F

T → F

F → (E)| id

F → id

Constructing SLR-Parsing Table

Example: ***Solution***

Step 1: Compute **FIRST** and **FOLLOW** sets for the given grammar **G**.

	E	T	F
FIRST	(, id	(, id	(, id
FOLLOW	+,), $	*, +,), $	*, +,), $

Constructing SLR-Parsing Table

Example: ***Solution***

Step 2: The LR(0) items can be obtained as shown below:

I_0:

E′ → .E

E → .E + T

E → .T

T → .T * F

T → .F

F → .(E)

F → .id

Constructing SLR-Parsing Table

Example: ***Solution***

Step 2: The LR(0) items can be obtained as shown below:

I_1: GOTO (I_0, E)

E′ → E.

E → E. + T

I_2: GOTO (I_0, T)

E → T.

T → T. * F

I_3: GOTO (I_0, F)

T → F.

Constructing SLR-Parsing Table

Example: ***Solution***

Step 2: The LR(0) items can be obtained as shown below:

I_4: GOTO (I_0, ()

F → (.E)

E → .E + T

E → .T

T → .T * F

T → .F

F → .(E)

F → .id

Constructing SLR-Parsing Table

Example: ***Solution***

Step 2: The LR(0) items can be obtained as shown below:

I_5: GOTO (I_0, id)

F → id.

I_6: GOTO (I_1, +)

E → E +. T

T → .T * F

T → .F

F → .(E)

F → .id

Constructing SLR-Parsing Table

Example: ***Solution***

Step 2: The LR(0) items can be obtained as shown below:

I_7: GOTO (I_2, *)

T → T *.F

F → .(E)

F → .id

I_8: GOTO (I_4, E)

F → (E.)

E → E. + T

Constructing SLR-Parsing Table

Example: ***Solution***

Step 2: The LR(0) items can be obtained as shown below:

I_9: GOTO (I_6, T)

E → E + T.

T → T. * F

I_{10}: GOTO (I_7, F)

T → T * F.

I_{11}: GOTO (I_8,))

F → (E).

Constructing SLR-Parsing Table

Example: ***Solution***

Step 2: The LR(0) items can be obtained as shown below:

GOTO (I_4, T) = I_2

E → T.

T → T. * F

GOTO (I_4, F) = I_3

T → F.

Constructing SLR-Parsing Table

Example: ***Solution***

Step 2: The LR(0) items can be obtained as shown below:

GOTO (I_4, ( ) = I_4

F → (.E)

E → .E + T

E → .T

T → .T * F

T → .F

F → .(E)

F → .id

Constructing SLR-Parsing Table

Example: ***Solution***

Step 2: The LR(0) items can be obtained as shown below:

GOTO (I_4, id) = I_5

F → id.

GOTO (I_6, F) = I_3

T → F.

GOTO (I_6, id) = I_5

F → id.

Constructing SLR-Parsing Table

Example: ***Solution***

Step 2: The LR(0) items can be obtained as shown below:

GOTO (I_6, ( ) = I_4

F → (.E)

E → .E + T

E → .T

T → .T * F

T → .F

F → .(E)

F → .id

Constructing SLR-Parsing Table

Example: ***Solution***

Step 2: The LR(0) items can be obtained as shown below:

GOTO (I_7, () = I_4

F → (.E)

E → .E + T

E → .T

T → .T * F

T → .F

F → .(E)

F → .id

Constructing SLR-Parsing Table

Example: ***Solution***

Step 2: The LR(0) items can be obtained as shown below:

GOTO (I_7, id) = I_5

F → id.

GOTO (I_8, +) = I_6

E → E +. T

T → .T * F

T → .F

F → .(E)

F → .id

Constructing SLR-Parsing Table

Example: ***Solution***

Step 2: The LR(0) items can be obtained as shown below:

GOTO (I_9, *) = I_7

T → T *.F

F → .(E)

F → .id

The canonical collection of **LR(0)** items are **C = {I_0, I_1, I_2, I_3, I_4, I_5, I_6, I_7, I_8, I_9, I_{10}, I_{11}}**

Constructing SLR-Parsing Table

Example: ***Solution***

Step 3 (a): Construction of **SLR parsing table**

The **ACTION** entries for terminals can be obtained as shown below: ***Algorithm Rule 2.a***

Transition GOTO (I_i, a) = I_j	ACTION [i, a] = shift j
I_0, (= I_4	[0, (] = s_4
I_0, id = I_5	[0, id] = s_5
I_1, + = I_6	[1, +] = s_6

Constructing SLR-Parsing Table

Example: ***Solution***

Transition GOTO $(I_i, a) = I_j$	ACTION [i, a] = shift j
I_2, * = I_7	[2, *] = s_7
I_4, (= I_4	[4, (] = s_4
I_4, id = I_5	[4, id] = s_5
I_6, (= I_4	[6, (] = s_4
I_7, (= I_4	[7, (] = s_4
I_7, id = I_5	[7, id] = s_5
I_8,) = I_{11}	[8,)] = s_{11}

Constructing SLR-Parsing Table

Example: ***Solution***

Transition GOTO $(I_i, a) = I_j$	ACTION [i, a] = shift j
$I_8, + = I_6$	$[8, +] = s_6$
$I_9, * = I_7$	$[9, *] = s_7$

Constructing SLR-Parsing Table

Example: ***Solution***

Step 3 (b): Construction of **SLR parsing table**

The **ACTION** entries for the items ending with dot (.) are shown below: ***Algorithm Rule 2.b***

$[A \rightarrow \alpha.] \epsilon I_i$	a = FOLLOW (A) then ACTION [i, a] = r $A \rightarrow \alpha$
$[E \rightarrow T.] \epsilon I_2$	[2, +,), $] = r $E \rightarrow T$ (i.e., r_2) FOLLOW (E) = {+,), $}
$[T \rightarrow F.] \epsilon I_3$	[3, *, +,), $] = r_4

Constructing SLR-Parsing Table

Example: ***Solution***

$[A \rightarrow \alpha.]\epsilon I_i$	a = FOLLOW (A) then ACTION [i, a] = r $A \rightarrow \alpha$
$[F \rightarrow id.]\epsilon I_5$	$[5, *, +,), \$] = r_6$
$[E \rightarrow E+T.]\epsilon I_9$	$[9, +,), \$] = r_1$
$[T \rightarrow T*F.]\epsilon I_{10}$	$[10, *, +,), \$] = r_3$
$[F \rightarrow (E).]\epsilon I_{11}$	$[11, *, +,), \$] = r_5$

Constructing SLR-Parsing Table

Example: ***Solution***

Step 3 (c): $[S' \rightarrow S.] \epsilon I_i$ then **ACTION [i, $] = accept**: ***Algorithm Rule 2.c***

$[S' \rightarrow S.] \epsilon I_i$	**ACTION [i, $] = accept**
$[E' \rightarrow E.] \epsilon I_1$	**[1, $] = accept**

Constructing SLR-Parsing Table

Example: ***Solution***

Step 3 (d): The **GOTO** states can be computed using *rule-3* are shown below: ***Algorithm Rule 3***

Transition GOTO $(I_i, A) = I_j$	Table GOTO [i, A] = j
I_0, E = I_1	[0, E] = 1
I_0, T = I_2	[0, T] = 2
I_0, F = I_3	[0, F] = 3
I_4, E = I_8	[4, E] = 8

Constructing SLR-Parsing Table

Example: ***Solution***

Step 3 (d):

Transition GOTO $(I_i, A) = I_j$	Table GOTO [i, A] = j
$I_4, T = I_2$	[4, T] = 2
$I_4, F = I_3$	[4, F] = 3
$I_6, T = I_9$	[6, T] = 9
$I_6, F = I_3$	[6, F] = 3
$I_7, F = I_{10}$	[7, F] = 10

Constructing SLR-Parsing Table

Example: The final SLR parsing table:

	ACTION						GOTO		
	id	+	*	(	)	$	E	T	F
0	S_5			S_4			1	2	3
1		S_6				acc			
2		r_2	S_7		r_2	r_2			
3		r_4	r_4		r_4	r_4			
4	S_5			S_4			8	2	3
5		r_6	r_6		r_6	r_6			
6	S_5			S_4				9	3
7	S_5			S_4					10
8		S_6			S_{11}				
9		r_1	S_7		r_1	r_1			
10		r_3	r_3		r_3	r_3			
11		r_5	r_5		r_5	r_5			

Constructing SLR-Parsing Table

In the above **SLR parsing table**, there are no multiple entries in **ACTION**, then the grammar is **SLR(1)** grammar.

Constructing SLR-Parsing Table

Practice Problem-1:

Consider the following grammar

1. S → L = R | R
2. L → * R | id
3. R → L

a) Obtain LR(0) items

b) Compute FIRST and FOLLOW symbols

c) Obtain SLR parsing table

d) Check whether the given grammar is SLR(1) or not?

Constructing SLR-Parsing Table

Practice Problem-2:

Consider the following grammar

1. **S → AaAb| BbBa**
2. **A → ϵ**
3. **B → ϵ**

Show that the given grammar is LL(1) but not SLR(1)

Constructing SLR-Parsing Table

Practice Problem-3:

Consider the following grammar

A → a| (A)

a) Find LR(0) items

b) Construct SLR parsing table

c) Show the parsing steps for the string "**((a))**"

Constructing SLR-Parsing Table

Practice Problem-4:

Consider the following grammar

S → SS+| SS* | a

a) Obtain canonical collection of LR(0) items

b) Construct SLR parsing table. Is the grammar SLR?

c) Show the actions of your parsing table on the input "**aa*a+**"

Constructing SLR-Parsing Table

Practice Problem-5:

Consider the following grammar

S → CC

C → cC

C → d

a) Obtain canonical collection of LR(0) items

b) Construct SLR parsing table.

c) Show the sequence of moves made by the parser for the string "**cdddd**"

Summary...

Bottom-Up Parsing: SLR-Parser

- Constructing SLR-Parsing Table
- Algorithm for Constructing SLR-Parsing Table
- Example problem

Reading: Aho2, Section 4.6.4

Next Lecture: More Powerful LR Parsers: CLR parser