

UNIT 6: BOTTOM-UP PARSING TECHNIQUES

We want to develop a translator for a language of arithmetic expressions to code C, Pascal or Java. The characteristics of the language are:

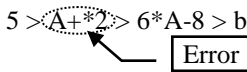
- Operands are variables and integer numeric constants.
- The arithmetic operators are: +, -, *, /
- All operations are integer.
- There is no precedence between the different arithmetic operators. Expressions are evaluated from left to right.
- The operator > loads the result of the arithmetic operation to the left of the operator in the variable to the right of it. To perform this "assignment" operation on the right only an identifier can appear as an expression, otherwise the result of evaluating the expression is lost.
- The iteration statement indicates the expressions that it affects with the operators [], the number of iterations indicating the value of the arithmetic expression enclosed in parentheses that will appear after the symbol].
- When the variables are first used, they are initialized with a value of 0.

Example:

3>A>[2*A-1+b>b](A)

The operations that are performed are: value 3 is assigned to variable A, the operation $2 * A - 1 + b$ is performed and the result, 5, is assigned to variable b, this last operation is repeated 5 times, therefore, after running the entire line, variable b will take the value 15.

An erroneous expression could be:

5 > A + * 2 > 6 * A - 8 > b


It is required

1. Define the grammar G for the translator.
2. Construct the SLR(1) analysis table for the translator.

Solution:

1. The tokens of the proposed grammar in this solution are: $\{[,], (,), >, \text{op}, \text{id}, \text{num}\}$. The token "op" represents any operation (+, -, *, /), can be treated as a grammatical symbol because they have the same precedence. The token "id" represents variables and "num" represents numeric constants (it could also return the lexical analyzer a single token for variables and constants). The production rules of the grammar are:

$$\begin{aligned} S &\rightarrow E > S \mid E \\ E &\rightarrow [S] (E) \mid P \text{ op } E \mid P \\ P &\rightarrow \text{id} \mid \text{num} \end{aligned}$$

The axiom, S, constructs the line of expressions joined with the > operator and the iterations. The non-terminal "E" constructs the arithmetic expressions, set of operands joined by an operator

After factoring the grammar:

$$\begin{aligned} S &\rightarrow E S' \\ S' &\rightarrow > S \mid \lambda \\ E &\rightarrow [S] (E) \mid P E' \\ E' &\rightarrow \text{op } E \mid \lambda \\ P &\rightarrow \text{id} \mid \text{num} \end{aligned}$$

2.

1. $S_0 \rightarrow S$
2. $S \rightarrow E S'$
3. $S' \rightarrow > S$
4. $S' \rightarrow \lambda$
5. $E \rightarrow [S] (E)$
6. $E \rightarrow P E'$
7. $E' \rightarrow \text{op } E$
8. $E' \rightarrow \lambda$
9. $P \rightarrow \text{id}$
10. $P \rightarrow \text{num}$

State 0	Action	Go To
$S_0 \rightarrow \cdot S$		[0,S]=1
$S \rightarrow \cdot E S'$		[0,E]=2
$E \rightarrow \cdot [S](E)$	[0,[]=D3	
$E \rightarrow \cdot P E'$		[0,P]=4
$P \rightarrow \cdot id$	[0,id]=D5	
$P \rightarrow \cdot num$	[0,num]=D6	
State 1	Action	Go To
$S_0 \rightarrow S \cdot$	[1,\$]=ACP	
State 2	Action	Go To
$S \rightarrow E \cdot S'$		[2,S']=7
$S' \rightarrow \cdot > S$	[2,>]=D8	
$S' \rightarrow \lambda$	[2,]=R4	
	[2,\$]=R4	
State 3	Action	Go To
$E \rightarrow [\cdot S](E)$		[3,S]=9
$S \rightarrow \cdot E S'$		[3,E]=2
$E \rightarrow \cdot [S](E)$	[3,[]=D3	
$E \rightarrow \cdot P E'$		[3,P]=4
$P \rightarrow \cdot id$	[3,id]=D5	
$P \rightarrow \cdot num$	[3,num]=D6	
State 4	Action	Go To
$E \rightarrow P \cdot E'$		[4,E']=10
$E' \rightarrow \cdot op E$	[4,op]=D11	
$E' \rightarrow \lambda$	[4,>]=R8	
	[4,)=R8	
	[4,]=R8	
	[4,\$]=R8	
State 5	Action	Go To
$P \rightarrow id \cdot$	[5,op]=R9	
	[5,>]=R9	
	[5,)=R9	
	[5,]=R9	
	[5,\$]=R9	
State 6	Action	Go To
$P \rightarrow num \cdot$	[6,op]=R10	
	[6,>]=R10	
	[6,)=R10	
	[6,]=R10	
	[6,\$]=R10	
State 7	Action	Go To
$S \rightarrow E S' \cdot$	[7,]=R2	
	[7,\$]=R2	

State 8	Action	Go To
$S' \rightarrow > \cdot S$		[8,S]=12
$S \rightarrow \cdot E S'$		[8,E]=2
$E \rightarrow \cdot [S](E)$	[8,[]=D3	
$E \rightarrow \cdot P E'$		[8,P]=4
$P \rightarrow \cdot id$	[8,id]=D5	
$P \rightarrow \cdot num$	[8,num]=D6	
State 9	Action	Go To
$E \rightarrow [S \cdot](E)$	[9,]=D13	
State 10	Action	Go To
$E \rightarrow P E' \cdot$	[10,>]=R6	
	[10,)=R6	
	[10,]=R6	
	[10,\$]=R6	
State 11	Action	Go To
$E' \rightarrow op \cdot E$		[11,E]=14
$E \rightarrow P \cdot E'$		[11,E']=10
$E' \rightarrow \cdot op E$	[11,op]=D11	
$E' \rightarrow \lambda$	[11,>]=R8	
	[11,)=R8	
	[11,]=R8	
	[11,\$]=R8	
State 12	Action	Go To
$S' \rightarrow > S \cdot$	[12,]=R3	
	[12,\$]=R3	
State 13	Action	Go To
$E \rightarrow [S] \cdot (E)$	[13,(]=D15	
State 14	Action	Go To
$E' \rightarrow op E \cdot$	[14,>]=R7	
	[14,)=R7	
	[14,]=R7	
	[14,\$]=R7	
State 15	Action	Go To
$E \rightarrow [S] (\cdot E)$		[15,E]=16
$E \rightarrow \cdot [S](E)$	[15,[]=D3	
$E \rightarrow \cdot P E'$		[15,P]=4
$P \rightarrow \cdot id$	[15,id]=D5	
$P \rightarrow \cdot num$	[15,num]=D6	
State 16	Action	Go To
$E \rightarrow [S] (E \cdot)$	[16,)=D17	
State 17	Action	Go To
$E \rightarrow [S] (E) \cdot$	[17,>]=R5	
	[17,)=R5	
	[17,]=R5	
	[17,\$]=R5	

Table SLR(1)		Shift							Go to						
		\$	>	()	[]	id	num	op	E'	E	P	S	S'
States	0					D3		D5	D6			2	4	1	
	1	ACP													
	2	R4	D8				R4								7
	3					D3		D5	D6			2	4	9	
	4	R8	R8		R8		R8			D11	10				
	5	R9	R9		R9		R9			R9					
	6	R10	R10		R10		R10			R10					
	7	R2					R2								
	8					D3		D5	D6			2	4	12	
	9						D13								
	10	R6	R6		R6		R6								
	11	R8	R8		R8		R8			D11	10	14			
	12	R3					R3								
	13				D15										
	14	R7	R7		R7		R7								
	15					D3		D5	D6			16	4		
	16				D17										
	17	R5	R5		R5		R5								