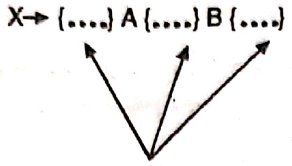
Ans. :

A translation scheme is a context free grammar(CFG) in which attributes are associated with the grammar symbols. And semantic actions enclosed between braces { } are inserted within the right side of productions.

For example



Semantic actions

Translation scheme generates the output by executing the semantic actions in an ordered manner. It uses depth first traversal. Consider a simple translation scheme that converts infix expressions into postfix expressions.

$E \rightarrow TR$

$R \rightarrow$ addop T {print (addop · lexeme) } R | ∈

$T \rightarrow$ num {print (num · Val) }

For example 3 – 5 + 4 as 3 5 – 4 +

Fig. 1-Q. 3(a) shows the annotated parse tree for the input 3 – 5 + 4 with each semantic action attached as the appropriate child of the node corresponding to the left side of their production.

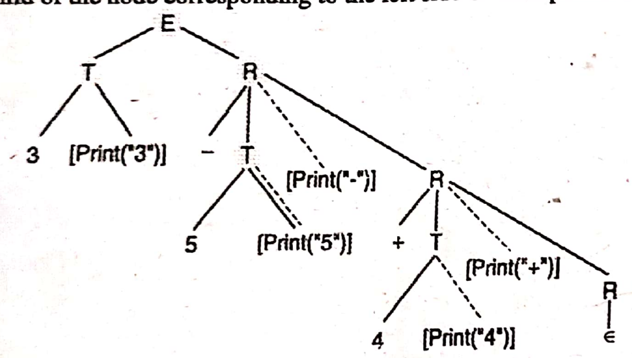


Fig. 1-Q. 3(a) : Annotated Parse tree for 3 – 5 + 4 with action

Q. 3(c) Check given grammar is LL(1) but not SLR(1).
S -> AaAb I BbBa
A -> €
B -> € **(7 Marks)**

Ans. :

FIRST (AaAb) = FIRST (A) – {ε} ∪ FIRST (aAb) = {a}

FIRST (BbBa) = FIRST (B) – {ε} ∪ FIRST (bBa) = {b}

FIRST (AaAb) ∩ FIRST (BbBa) = {a} ∩ {b} = ϕ

Hence the grammar is LL(1).

For parsing table,

FIRST (S) = FIRST (AaAb) ∪ FIRST (BbBa)

= {a} ∪ {b} = {a, b}

FIRST (A) = {ε}

FIRST (B) = {ε}

FOLLOW (S) = {$}

Using $S \rightarrow AaAb$ It get :

FOLLOW (A) = FIRST (aAb) = {a} and

FOLLOW (A) = FIRST (b) = {b}

FOLLOW (A) = {a, b}

$S \rightarrow BbBa$

FOLLOW (B) = FIRST (bBa) = {b}

FOLLOW (B) = FIRST (a) = {a}

FOLLOW (B) = {a, b}

Parsing table for the grammar

	a	b	$
S	$S \rightarrow AaAb$	$S \rightarrow BbBa$	
A	$A \rightarrow \varepsilon$	$A \rightarrow \varepsilon$	
B	$B \rightarrow \varepsilon$	$B \rightarrow \varepsilon$	

No duplicate entries in parsing table Hence the given grammar is LL(1)

Now check for SLR,

Augmented Grammar,

$S' \rightarrow S$

$S \rightarrow AaAb$

$S \rightarrow BbBa$

$A \rightarrow \varepsilon$

$B \rightarrow \varepsilon$

I_0 : $S' \rightarrow \cdot S$

$S \rightarrow \cdot AaAb$

$S \rightarrow \cdot BbBa$

$A \rightarrow \cdot$

$B \rightarrow \cdot$

I_1 : $S' \rightarrow S \cdot$

I_2 : $S \rightarrow A \cdot aAb$

I_3 : $S \rightarrow B \cdot bBa$

I_4 : $S \rightarrow Aa \cdot Ab$

$A \rightarrow \cdot$

I_5: $S \rightarrow Ba \cdot Ba$

$B \rightarrow \cdot$

I_6: $S \rightarrow AaA \cdot b$

I_7: $S \rightarrow BbB \cdot a$

I_8: $S \rightarrow AaAb \cdot$

I_9: $S \rightarrow BbBa \cdot$

Parsing Table for the grammar

	Action			goto		
	a	b	$	S	A	B
I_0	R_3/R_4	R_3/R_4		1	2	3
I_1			accept			
I_2	S_4					
I_3		S_5				
I_4	R_3	R_3				
I_5	R_4	R_4			6	
I_6		S_8				7
I_7	S_9					
I_8			R_1			
I_9			R_2			

Since the action table contains multiple entries, the above grammar is not SLR (1) grammar.

Q. 3(c) **Construct CLR parsing table for the following grammar.**

S→ CC

C→ cC | d **(7 Marks)**

Ans. :

$I_0 : S' \to \cdot S, \$$

$S \to \cdot cC, \$$

$C \to \cdot cC, c/d$

$C \to \cdot d, c/d$

With closure on $S' \to \cdot S$, \$ with item $[A \to \alpha \cdot B\beta, a]$

That is $A \to S'$, $\alpha = \varepsilon$, $B = S$, $\beta = \varepsilon$ and $a = \$$.

Function closure tells us to add $[B \to \cdot y, b]$ for each production $B \to y$ and terminal b in first (βa). In terms of present grammar, $B \to y$ must be $S \to cC$ and β is ε and a is \$

$$\therefore \text{first } (\beta a) = (\varepsilon\$) = \$$$

It continue to compute augmented closure by adding all items.

$$A \to \alpha \cdot B\beta, a$$

$$S \to \cdot cC, \$$$

$$\text{First } (\beta a) = \text{FIRST } (c\$) = \{c, d\}$$

As C is having two productions

$C \to \cdot cC$ and $C \to \cdot d$

$\therefore$ b = c/d

I_1 : goto (I_0, S) :

$S' \to S \cdot, \$$

I_2 : goto (I_0, C)

$S \to C \cdot C, \$$

$C \to \cdot cC, \$$

$C \to \cdot d, \$$

I_3 : goto (I_0, c)

$C \to c \cdot C, c/d$

$C \to \cdot cC, c/d$

$C \to \cdot d, c/d$

I_4 : goto (I_0, d)

$C \to d \cdot, c/d$

I_5 : goto (I_2, C)

$S \to CC \cdot, \$$

I_6 : goto (I_2, c)

$C \to c \cdot C, \$$

$C \to \cdot cC, \$$

$C \to \cdot d, \$$

I_7 : goto(I_2, d)

$C \to d \cdot, \$$

I_8 : goto (I_3, C)

$C \to cC., c/d$

I_9 : goto (I_6, C)

$C \to cC \cdot, \$$

goto $(I_3, c) = I_3$

goto $(I_6, c) = I_6$

Parsing Table for the grammar

State	Action			Goto	
	c	d	\$	S	C
0	S_3	S_4		1	2
1			accept		
2	S_6	S_7			5
3	S_3	S_4			8
4	r_3	r_3			

State	Action			Goto	
	c	d	\$	S	C
5			r_1		
6	S_6	S_7			9
7			r_3		
8	r_2	r_2			
9			r_2		

Q. 3(b)(OR) **Find out FIRST and FOLLOW for the following grammar.**

S→ 1AB | ε A→ 1AC | 0C B→ 0S

C→ 1 **(4 Marks)**

Ans. :

FIRST(S) = {1,ε}

FIRST (A) = {1, 0}

FIRST (B) = {0}

FIRST(C) = {1}

FOLLOW (A) = {FIRST(C) ∪ FIRST (B)} = {1, 0}

FOLLOW (B) = FOLLOW(S) = {$}

FOLLOW (C) = FOLLOW (A) = {1, 0}

The parsing table is constructed from above FIRST and OLLOW functions.

	1	0	$
S	S→1AB		S→ε
A	A→1AC	A→0C	
B		B→0S	
C	C→1		

No duplicate entries in parsing table Hence the given ammar is LL(1)