CS & IT ENGINEERING

Compiler Design

Lexical Analysis & Syntax Analysis

Lecture No. 10

By- DEVA Sir

→ LR(0) Table
SLR(1) "
LALR "
CLR "

→ LR Algorithm

→ Operator precedence

LR(0) Table construction :

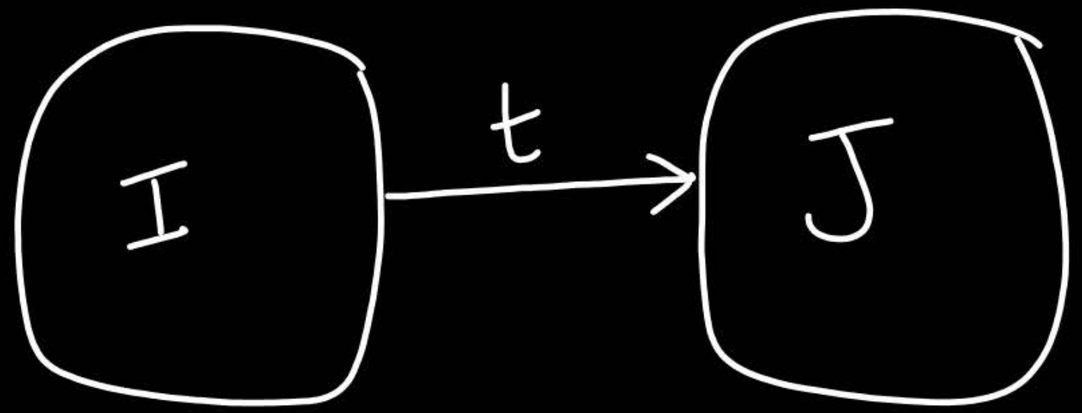
Table Entries
- Shift Entry
- goto(state) Entry
- Reduced Entry
- Acceptance Entry
- Blank Entry

Table size =

No. of rows X No. of columns

No. of states X $\left(\underbrace{|T|+1}_{\text{Action}} + \underbrace{|V|}_{\text{goto}}\right)$

Shift Entry in LR(0) / SLR(1) / LALR(1) / CLR(1) :

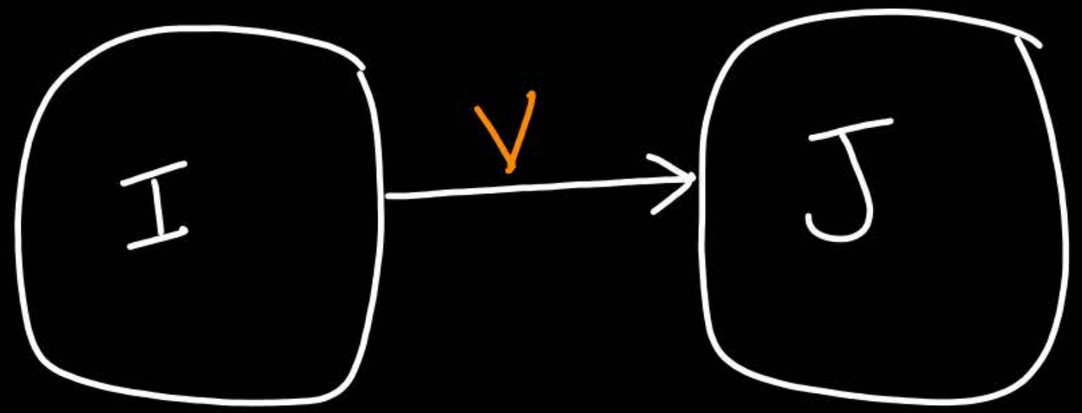


Every terminal transition will make one shift Entry

	t
I	S_J

No. of terminal transitions
=
No. of Shift Entries

State Entry in LR(0) / SLR(1) / LALR(1) / CLR(1) :
(goto)



Every Non-terminal transition will make one goto Entry

	V
I	J

No. of Nonterminal transitions
=
No. of goto Entries

Acceptance Entry for LR(0)/SLR(1)/LALR(1)/CLR(1)

$S' \to S.$

Acceptance Item

I

	$
I	Accept

Reduced Entries for LR(0): [Every Reduced Item in every state, need to apply the below Rule]

I

(ii) $X \to \alpha.$

	← Action →			
I	R_{ii}	R_{ii}	R_{ii}	R_{ii}
	$X \to \alpha$	$X \to \alpha$	$X \to \alpha$	$X \to \alpha$

Reduced Entries for SLR(1)

$Follow(X) = \{t_1, t_2\}$

Computes follow set using whole CFG

		t_1		t_2
I		R_{ii}		R_{ii}

Reduced Entries for CLR & LALR:

I

(ii) $X \to \alpha., t_1$

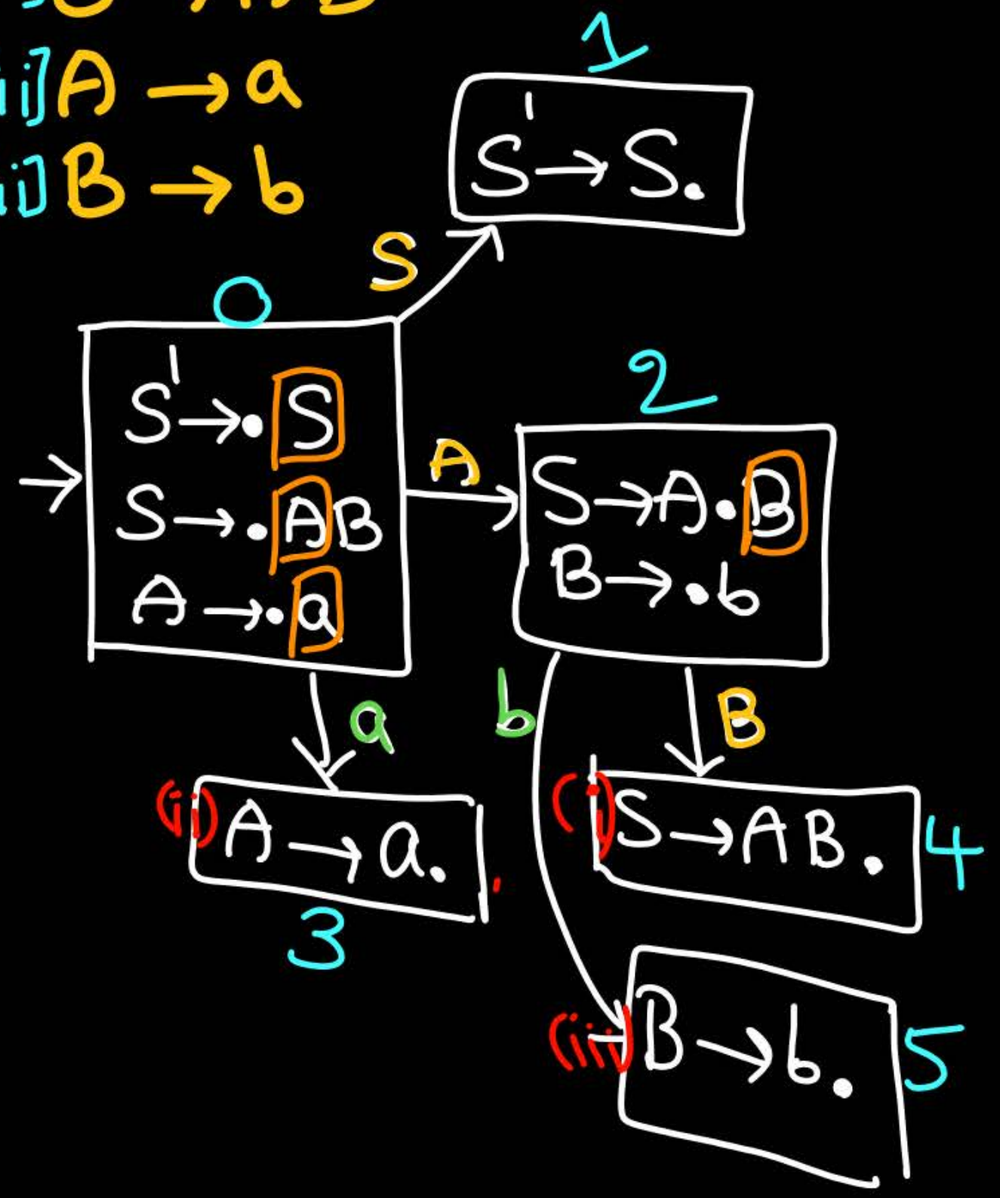
		t_1
I		R_{ii}

LR(0) Table :

[i] $S \to AB$

[ii] $A \to a$

[iii] $B \to b$



LR(0) States	Action: a	Action: b	Action: $	Goto: S	Goto: A	Goto: B
0	S_3			1	2	
1			ACCEPT			
2		S_5				4
3	R_{ii}	R_{ii}	R_{ii}			
4	R_i	R_i	R_i			
5	R_{iii}	R_{iii}	R_{iii}			

SLR(1) Table :

[i] $S \to AB$

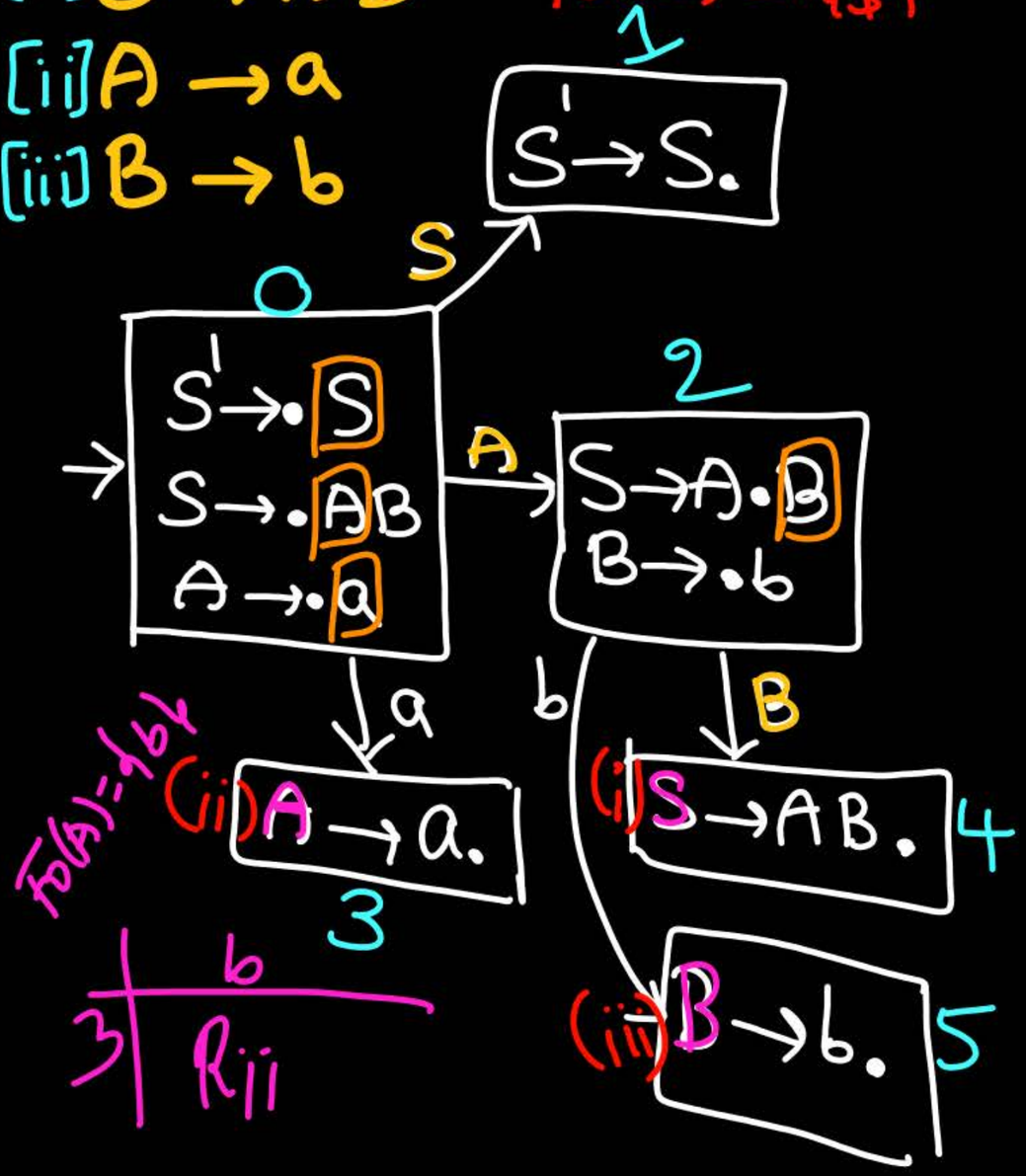
[ii] $A \to a$

[iii] $B \to b$

$Fo(S) = \{\$\}$

$Fo(A) = \{b\}$

$Fo(B) = \{\$\}$



$Fo(A) = \{b\}$

	b
3	R_{ii}

SLR(1)

	Action			Goto		
	a	b	$	S	A	B
0	S_3			1	2	
1			ACCEPT			
2		S_5				4
3		R_{ii}				
4			R_i			
5			R_{iii}			

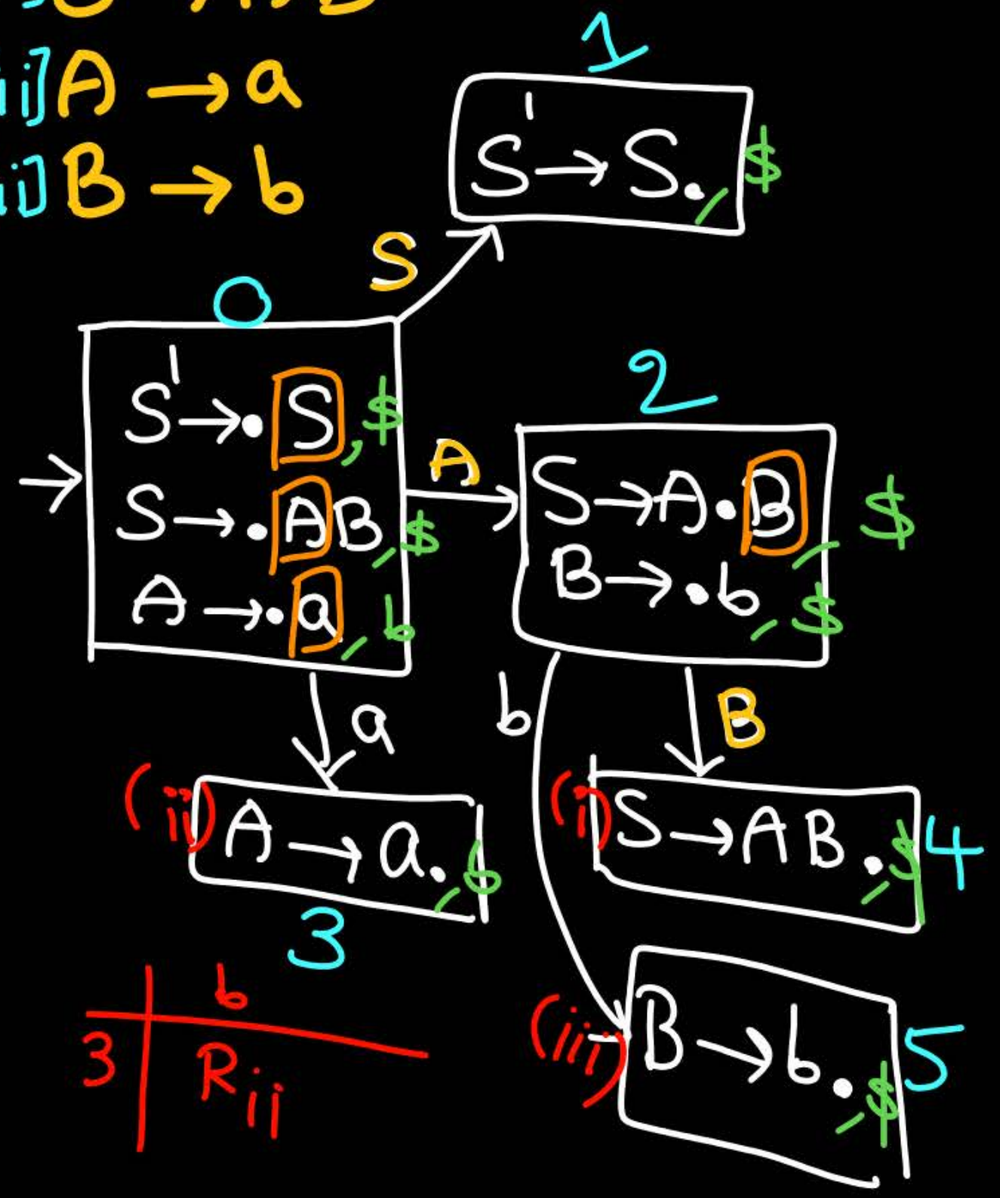
States

LALR(1) Table :

[i] $S \to AB$

[ii] $A \to a$

[iii] $B \to b$



	b
3	R_{ii}

CLR(1) or LALR(1)

	Action			Goto		
	a	b	$	S	A	B
0	S_3			1	2	
1			ACCEPT			
2		S_5				4
3		R_{ii}				
4			R_i			
5			R_{iii}			

States

If CLR DFA & LALR DFA same

then Table also same.

If CLR DFA has more states than LALR

then Table size also different

state numbering 0 1 2 3 4 . . .

production numbering i ii iii

S

(i) $A \to a.$

(ii) $B \to a.$

RR Conflict

4 RR Conflicts in LR(0)

$\underbrace{|T|+1}_{\underbrace{3+1}}$

	a	b	c	$
5	R_i R_{ii}	R_i R_{ii}	R_i R_{ii}	R_i R_{ii}

	LR(0) (n_1)	SLR (n_2)	LALR (n_3)	CLR (n_4)	
(No. of rows) No. of states	$n_1 = n_2 = n_3 \le n_4$				$n_i \rightarrow$ no. of states
Size of Table	$n_1 = n_2 = n_3 \le n_4$				$n_i \rightarrow$ size of table
No. of shift Entries	$n_1 = n_2 = n_3 \le n_4$				$n_i \rightarrow$ no. of shift entries
No. of goto entries	$n_1 = n_2 = n_3 \le n_4$				$n_i \rightarrow$ no. of goto entries
*** No. of Reduced Entries	$n_1 \ge n_2 \ge n_3 \ge n_4$				$n_i \rightarrow$ No. of Reduced entries
No. of columns	$n_1 = n_2 = n_3 = n_4$				$n_i \rightarrow$ no. of columns

Reduced Entries

- In LR(0) : no knowledge of look-a-heads

  All columns of action table will have reduced entries

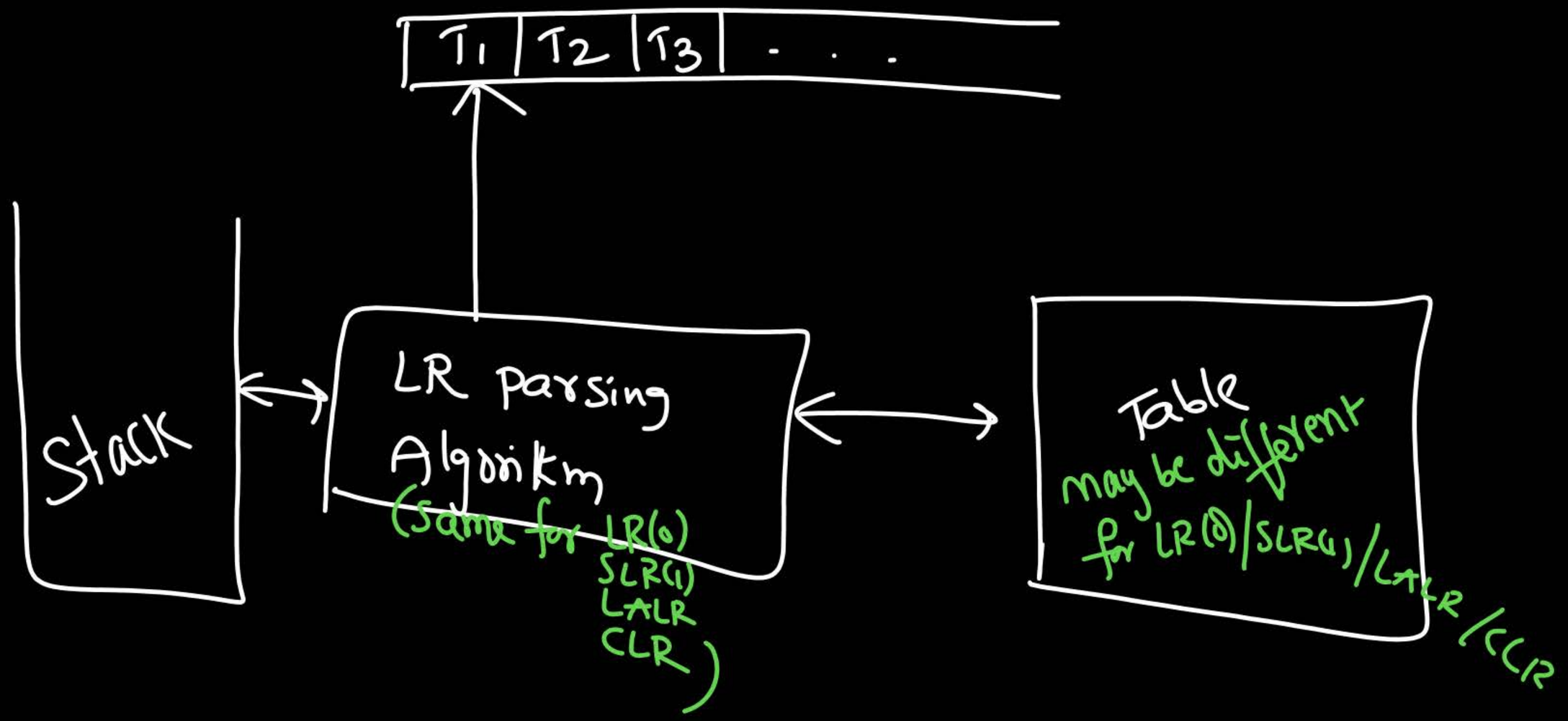
- In SLR(1) : Simple knowledge for computing look-a-heads using whole CFG.

  Follow set of Reduced Item decides Reduced Entries

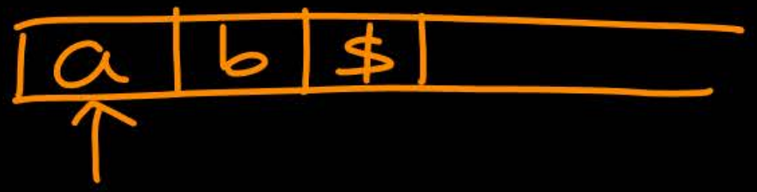
- In LALR(1) : Combined paths knowledge to compute look-a-heads

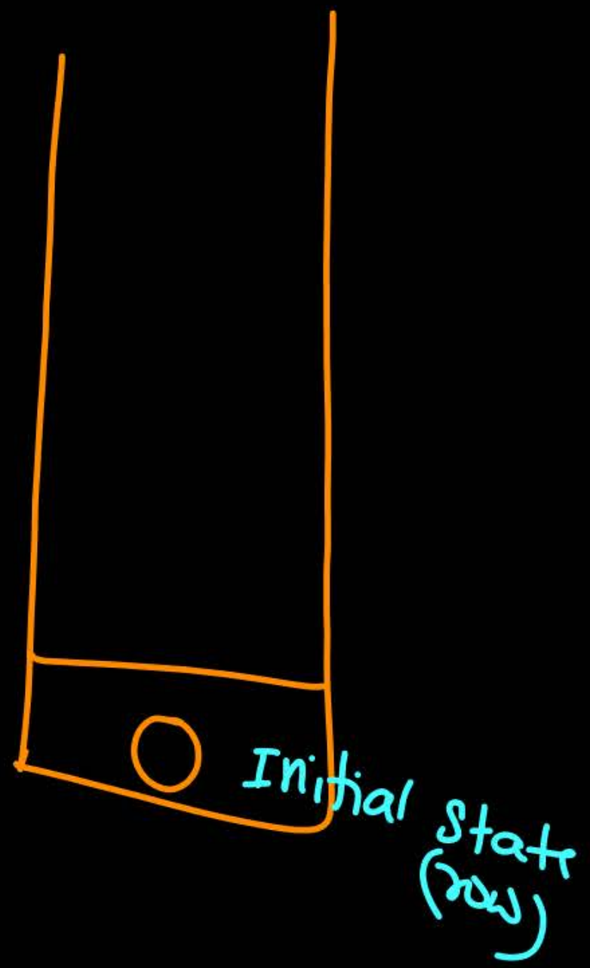
- In CLR(1) : Exact path knowledge (Canonical)

LR parser


T1 | T2 | T3 | . . .
Stack
LR parsing Algorithm
(Same for LR(0)
SLR(1)
LALR
CLR)
Table
may be different
for LR(0)/SLR(1)/LALR/CLR

LR Parsing Algorithm:





I) Shift Action

II) Reduced Action
(goto will be used internally for reduced action)

III) Accept Action

IV) Error Action

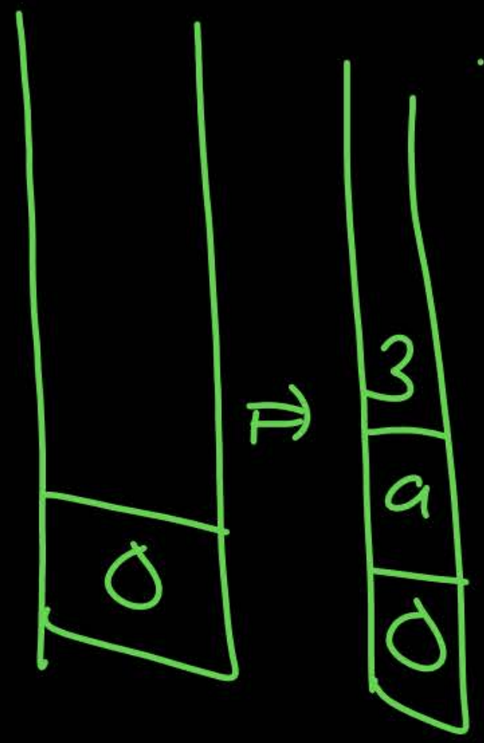
Actions

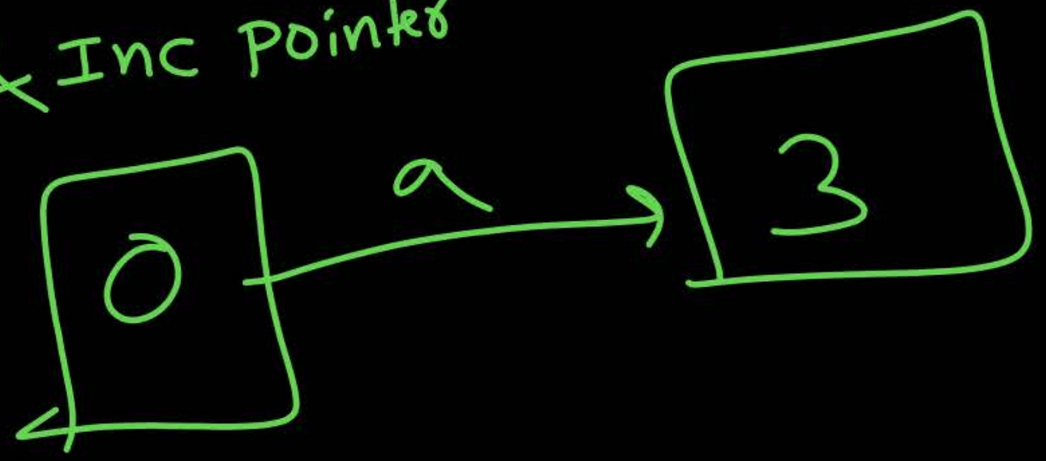
Shift Action:

$M[0, a] : S_3$

i) Push a & Inc pointer

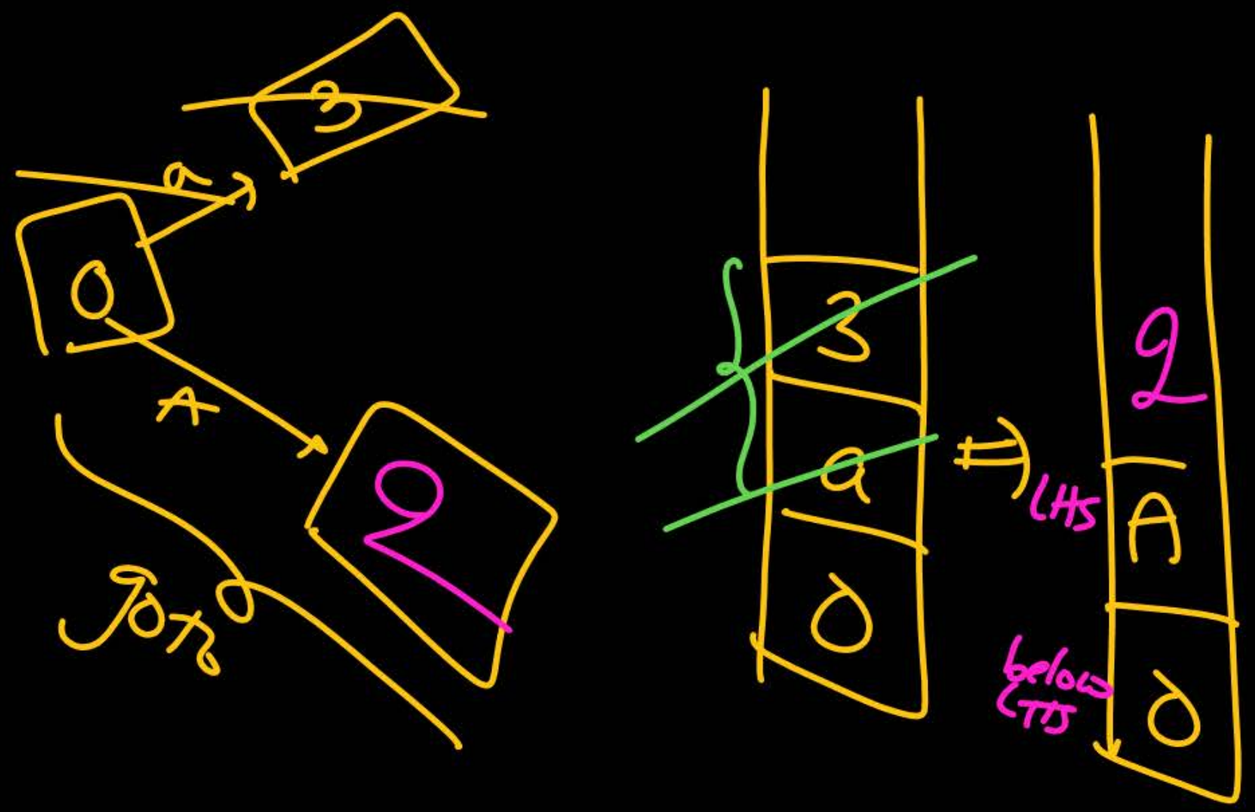
ii) Push 3





Reduced Action:

$M[3, b] : A \rightarrow a$



I) Pop $2 * |RHS|$
Pop $\underbrace{2 * |a|}_{2}$

II) Push LHS (A)

goto Table III) Push $M[\text{below LHS}, \text{LHS}]$
$M[0, A]$

① a b $

0

M[0,a]: S_3

② b $

3
a
0

i) Pop 2*|a| sym
ii) Push A
iii) Push M[0,A]

M[3,b]: A→a

③ b $

2
A
0

i) Push b, Inc
ii) Push 5

M[2,b]: S_5

④ $

5
b
2
A
0

i) Pop 2 sym
ii) Push B
iii) M[2,B] Push
4

M[5,$]: B→b

⑤ $

~~4~~
~~B~~
~~2~~
A
0

i) Pop 2*|AB| symbols
4 symbols
ii) Push S
iii) Push [0,S]
1

M[4,$]: S→AB

⑥ $

1
S
0

M[1,$]
=ACCEPT

Input: ab

	a	b	$	S	A	B
0	S_3			1	2	
1			ACCEPT			
2		S_5				4
3		A→a R_{ii}				
4			S→AB R_i			
5			B→b R_{iii}			

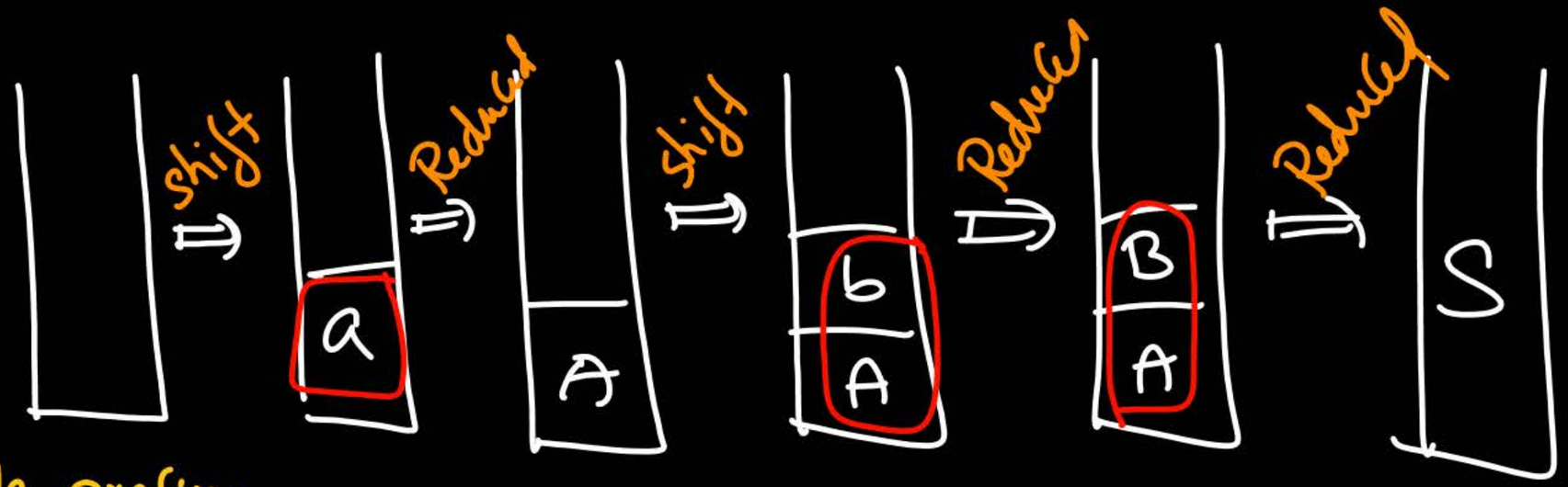
$S \to AB$

$A \to a$

$B \to b$

a	b	$

$w = ab$



S.

AB is reduced to S

$S \to \underbrace{AB}_{\text{Handle}}$

AB

b is reduced to B

$B \to \underbrace{b}_{\text{Handle}}$

A.b

Shift

a is Reduced to A

$A \to \underbrace{a}_{\text{Handle}}$

a b

Shift

Viable prefix:

→ prefix of Right Sentential form and top of stack sequence

→ Dot is not allowing to move further

→ Indication to perform Reduced Action

a, Ab, AB

bX

AX

BX

Summary

→ Next: Operator precedence

practice GATE PYQ


THANK YOU
GW
SOLDIERS !