Bottom-Up Parser. Date-18/10/22

LR Parser

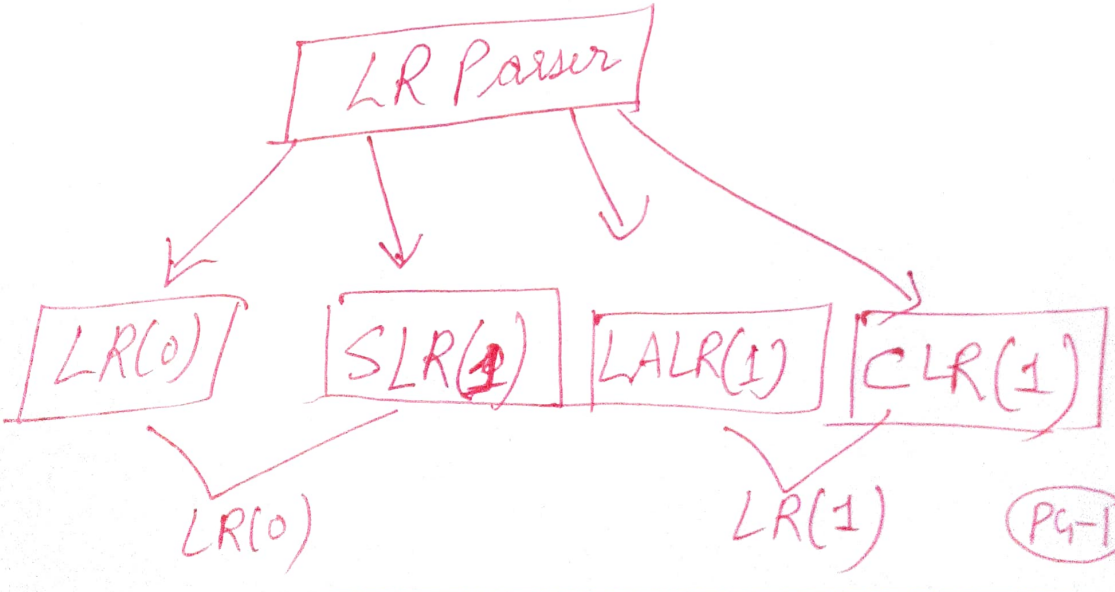
(1.) LR parser take bottom-Up Approach. That means leave to Root or terminals to non-terminals.

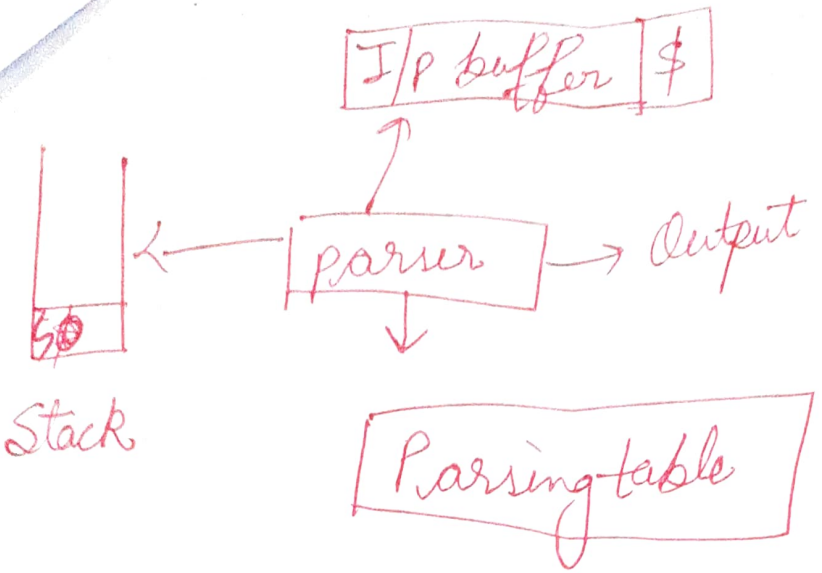
(2) It's perform reduction peration means terminal is reduced by particular production.

(3) LR stands for

L → Left to Right Scan the input.

R → Right most derivation in Reverse order.





Input buffer:- Input buffer contain string to be parse. which end with $.

Stack — It's a data structure and has 2 operation (push and pop) performed. Initially, contain ($)

Parsing table — it's 2D array / table. and has 2 parts ① Action ② Go to.

Action → list of terminals
Go to → list of non-terminals.

Parser → which parser Algorithm is used for construction Parsing table. (Parser is same in all)

Output → If, it's accepted the input string by production, then generate parser tree.

LR(0) items [LR(0), SLR(1)]

LR(1) items [LALR(1), CLR(1)]

LR(0) items → Any production we (·) dot on RHS of part. that is LR(0) item end.

$E \rightarrow \cdot aA$

LR(1) – → Look ahead.

$E \rightarrow \cdot aA, a/b$

LR Parser

Construction of LR parsing table.

Steps are following below.

(1) Writing Augmented grammar.

(2) LR(0) / LR(1) collection of items to be found.

(3) Defining 2 function; Goto (List of Non-terminals) & Action (List of Terminals) in the parsing table.

Rules for LR(0) / LR(1) items.

(1) If any non-terminals has (•) preceding it, we has to write all its production and (•) preceding each of its production.

(2) From each state to the next state. then (•) shifts to one place to the right.

LR(0) - Construction LR(0) parsing table for given CFG. And check the string w (w = aabb) is accepted by this grammar or not.

$S \rightarrow AA$
$A \rightarrow aA/b$

Answer

Grammar are -

$S \rightarrow AA$ — production (1)
$A \rightarrow aA$ — production (2)
$A \rightarrow b$ — production (3)

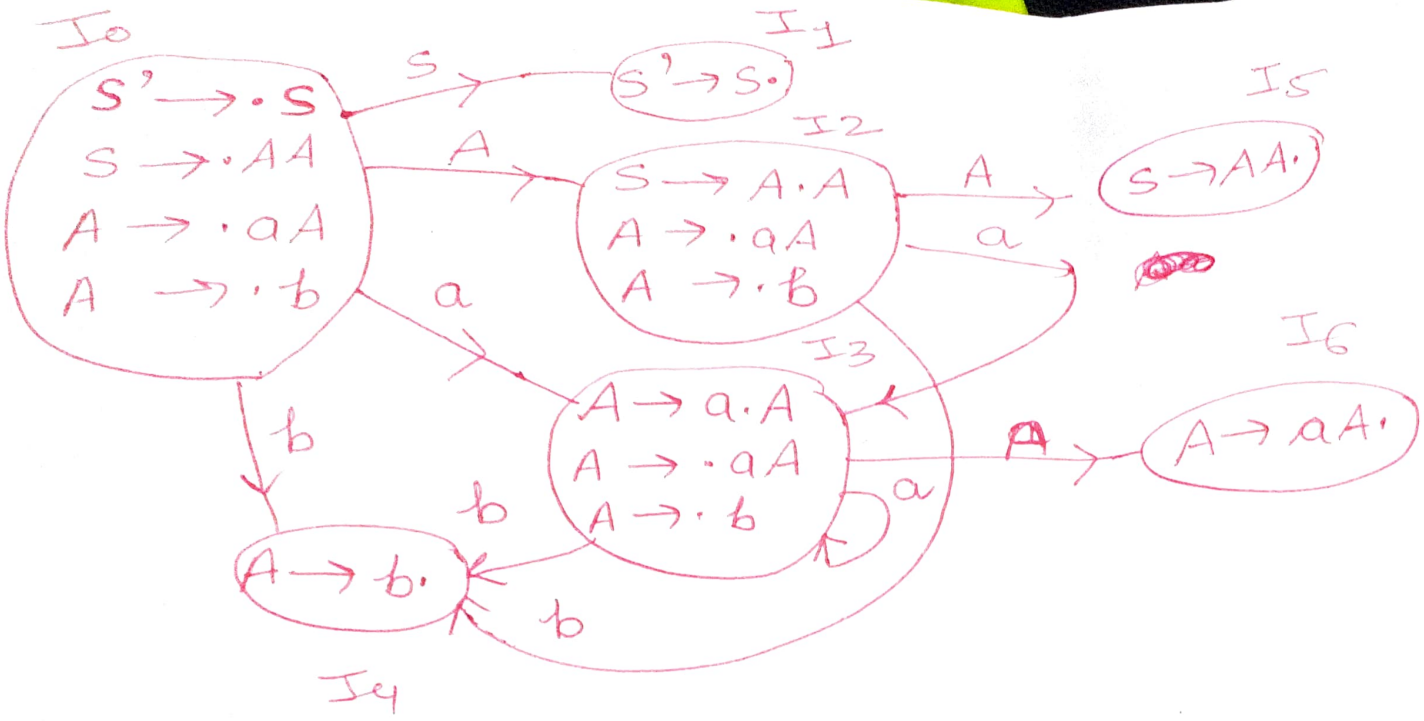
Step-1 Find Augmented Grammar.

$S' \rightarrow S$
$S \rightarrow AA$
$A \rightarrow aA$
$A \rightarrow b$

Step-2 Find LR(0) items collections.

we find LR(0) items using Augmented Grammar

find colsure


I0
S' → ·S
S → ·AA
A → ·aA
A → ·b
S
I1
S' → S·
A
I2
S → A·A
A → ·aA
A → ·b
a
I3
A → a·A
A → ·aA
A → ·b
b
I4
A → b·
b
b
a
A
I5
S → AA·
A
I6
A → aA·
a
A

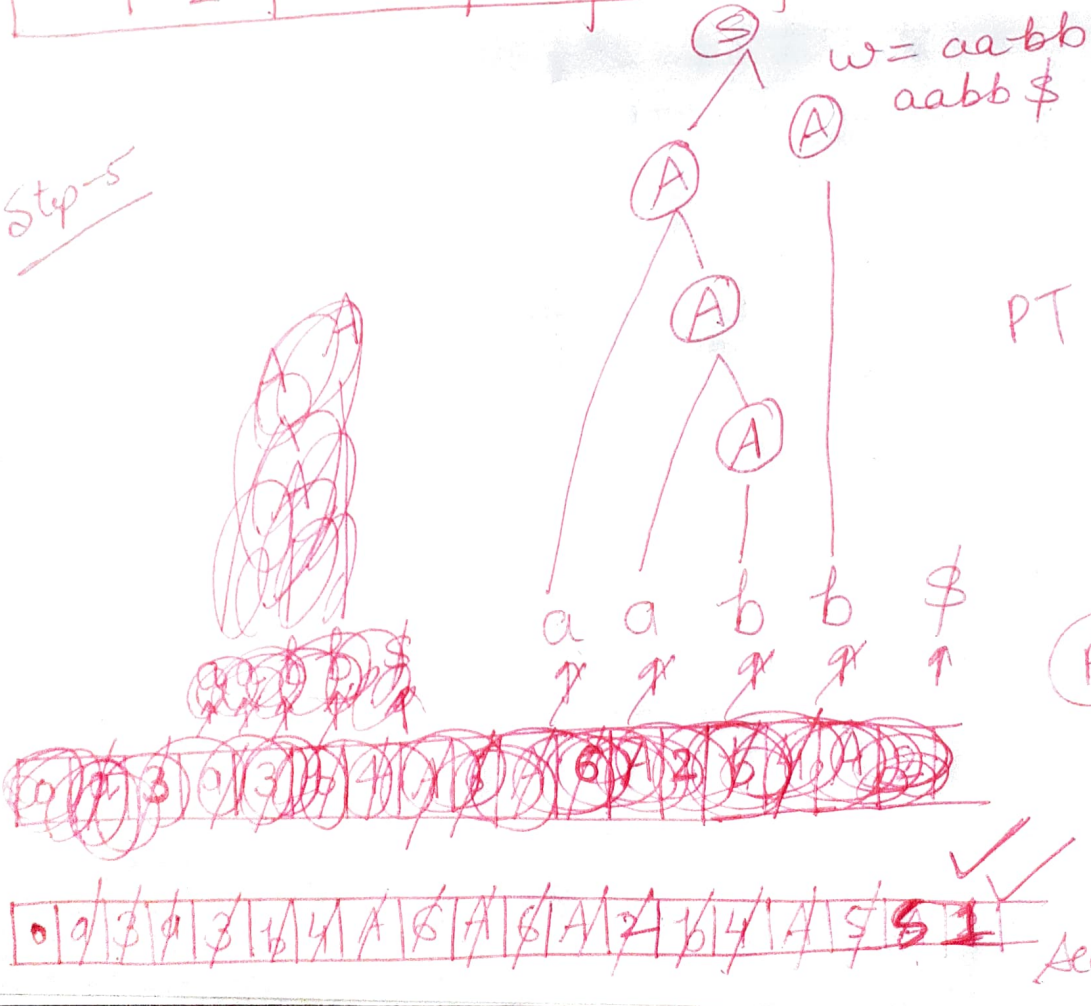
Pg-6

Step-4 Make LR(0) Table.

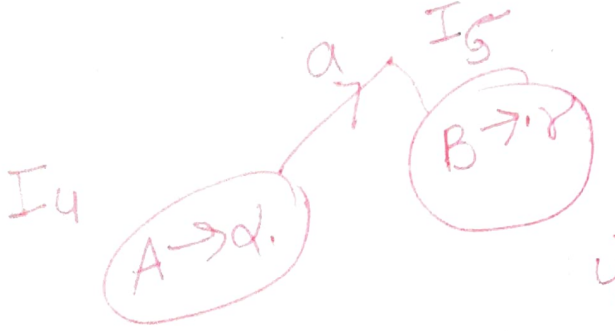
$S \rightarrow AA$ — (1)
$A \rightarrow aA$ — (2)
$A \rightarrow b$ — (3)

States	Action			Goto	
	a	b	$	S	A
0	S_3	S_4		1	2
1			Accepted		
2	S_3	S_4			5
3	S_3	S_4			6
4	R_3	R_3	R_3		
5	R_1	R_1	R_1		
6	R_2	R_2	R_2		

Step-5

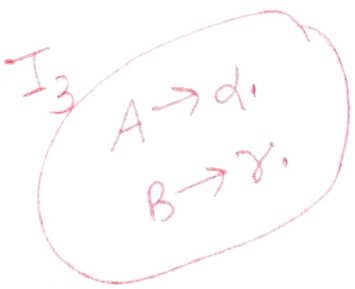


Any Grammar is not in LR(0). if SR/RR conflit.
By Two conflit.



	a	b
4	S5/~~R1~~	

Shift Reduce conflit



	a	b
I_3	r_1/r_2	

RR conflit.

Shweta Tiwari

Shweta Tiwari