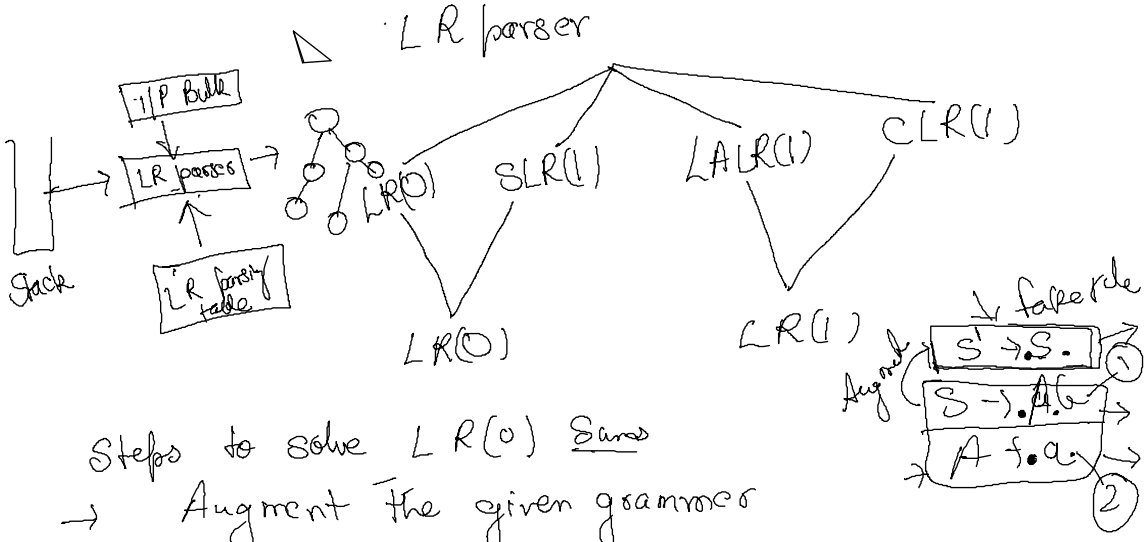
LR(0)

07 September 2022 08:38

LR parser → Bottom top approach



Steps to solve LR(0) Parser

→ Augment the given grammer
→ Number the production
→ Draw conical collection of LR(0) item (DFD
→ Create the parsing table → Grammer to be passed LR(0) or not
→ Stack implementation
→ Draw parse tree

I → cc dd $

Q
E → BB
B → cB
B → d → LR(0) or not?
cc dd $ → accepted or not - ?

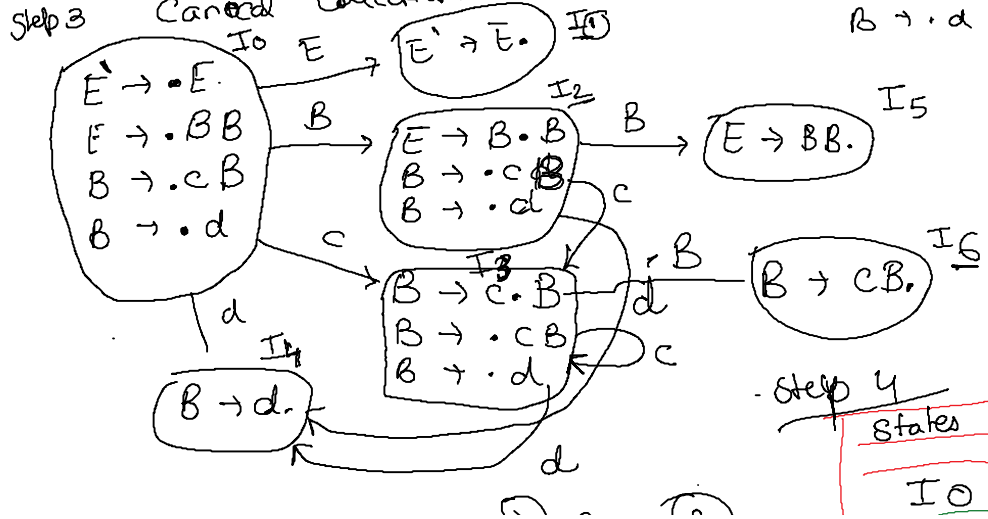
Solution

Step 1 → Augment the given grammer

E' → E ← Augmented
E → BB → (1)
B → cB → (2)
B → d → (3)

Step 2 ~~Draw the~~ Number the production

Step 3 Canonical Collection DFD

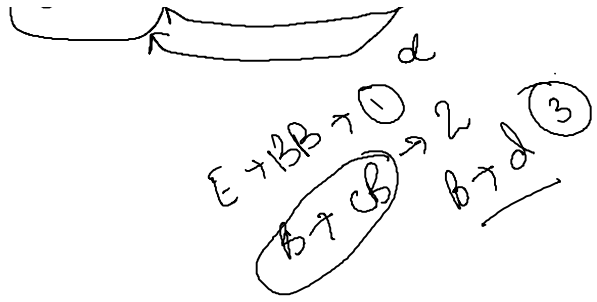


B → c.B
B → .cB
B → .d

Step 4

Parsing Table

States	Action			Goto	
	c	d	$	E	B
I0	S3	S4		1	2



States	c	d	$	E	B
I0	S3	S4		1	2
I1			Accept		
I2	S3	S4			5
I3	S3	S4			6
I4	r3	r3	r3		
I5	r1	r1	r1		
I6	r2	r2	r2		

Step 5

Stack implementation →

$E' \to E$

$E \to BB \to (1)$

$B \to CB \to (2)$

$B \to d \to (3)$

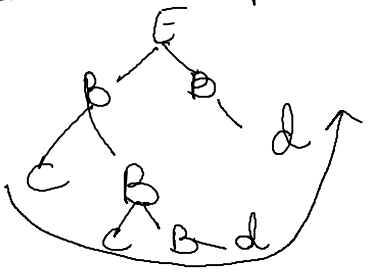
cc dd $

Stack	Input	Action
$0	ccdd$	Shift c in Stack get the state 3
$0c3	cdd$	Shift c in Stack get the state 3
$0c3c3	dd$	Shift d in stack get state 4
$0c3c3d4	d$	reduce r3 B → d
$0c3c3B6	d$	reduce r2 B → CB
$0c3B6	d$	reduce r2 B → CB
$0B2d4	$	Shift d → state 4
$0B2B5	$	reduce r3 B → d
$0E1	$	reduce r1 E → BB
		Accepted

Step 6

Draw the parse tree.

cc dd $



proved