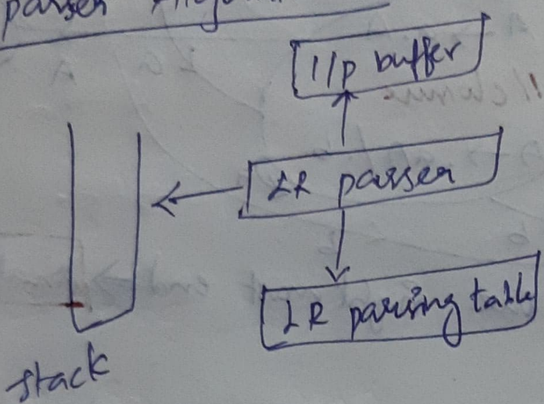
LR parsers:

LR(0) | SLR(1) Simple LR | LALR(1) Look-Ahead LR | CLR Canonical LR

LR-parser Algorithm:-



To construct LR parsing table

for 1. LR(0) & SLR(1) - we need canonical collections of LR(0) items

2. LALR & CLR - we need canonical collections of LR(1) items

Example for LR(0):-

$S \to AA$
$A \to aA/b$
actual grammar

$\longrightarrow$

$S' \to S$
$S \to AA$
$A \to aA/b$
Augmented grammar

step 1: augmented grammar

• In any production is called item → helps in reduction

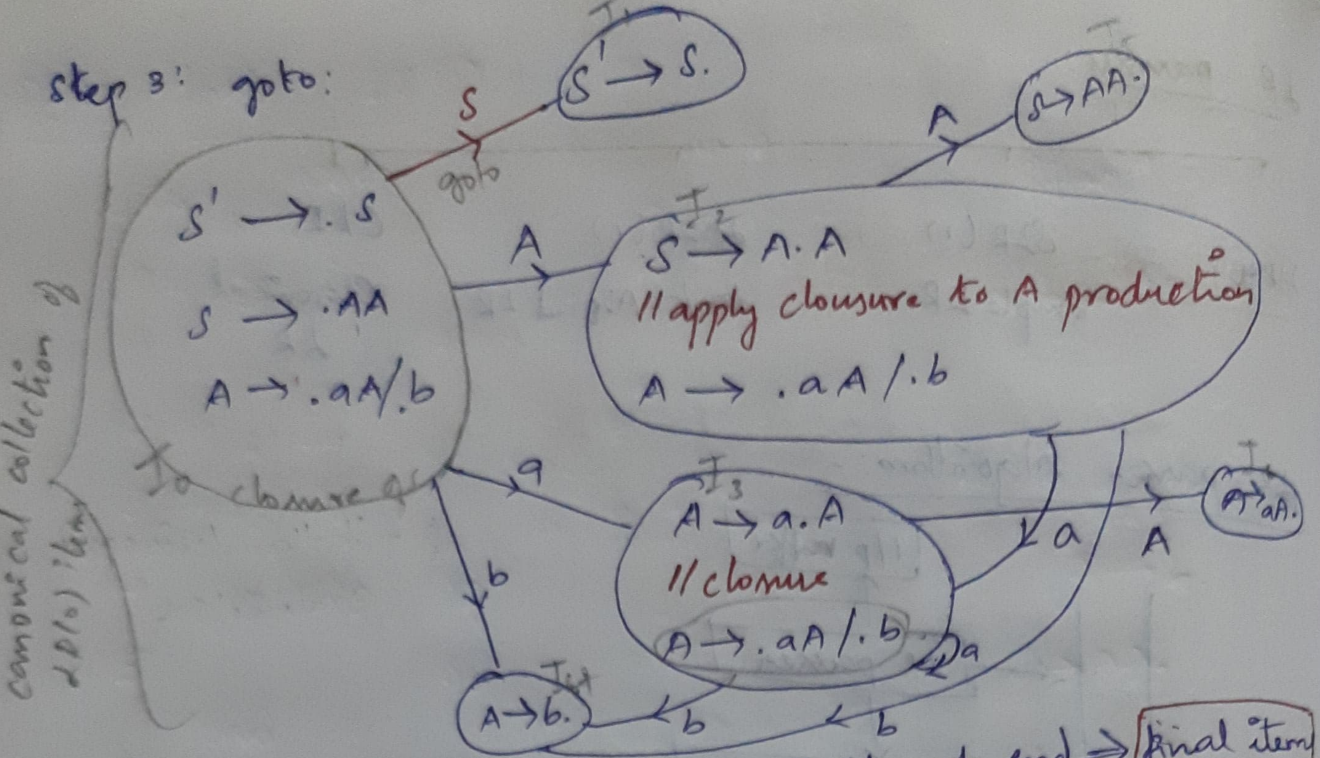
eg: $S \to \overset{\text{item}}{\cdot} AA$

$\to S' \to .S$

step 2: closure

$$\left.\begin{array}{l} S' \to .S \\ S \to .AA \\ S \to .aA/.b \end{array}\right\} \text{Closure of } S'$$

step 3: goto:



⇒ when the dot is at the rightmost end ⇒ final item

Original grammar ⟶ Augmented grammar ⟶ item to Augmented grammar ⟶ closure ⟶ goto & closure ⟶ final item

helps in making decision, (we have seen all, let reduce it)

parse table:-

	action			Goto	
	a	b	$	A	S
0	s_3	s_4		2	1
1			accept*		
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		

$S' \to S$

$S \to \underset{1}{AA}$

$S \to \underset{2}{aA} / \underset{3}{b}$

* ⟶ final item of augmented production $\xrightarrow{\text{then}}$ accept

Coz the final item of augmented production implies we reached starting symbol

Input string: illustration:

w = aabb$

| 0 | a | 3 | a | 3 | b | 4 | b | 4 |

w = aabb$

| 0 | a | 3 |

S_3 means shift the symbol & state no to stack & increase the ptr.

w = aabb$

| 0 | a | 3 | a | 3 |

w = aabb$

| 0 | a | 3 | a | 3 | b | 4 |

reduce the previous symbol

w = aabb$

| 0 | a | 3 | a | 3 | b | 4. |

r_3 ⇒ reduce it by using 3rd production (A→b) & pop the n elements from stack

n = 2(length of (production))

here, n = 2 * |b|

n = 2 * 1 = 2

push the LHS of the rule

| 0 | a | 3 | a | 3 | A | 6 |

w = aabb$

| 0 | a | 3 | A | 6 |

w = aabb$

| 0 | A | 2 |

w = aabb$

| 0 | A | 2 | b | 4 |

→ w = aabb$

| 0 | A | 2 | A | 5 |

w = aabb$

accept ⇐ | 0 | S | 1 |

w = aabb$

A.

oduction

A → aA.

→ final item

m to
mented
rammar

t reduce it

S → S

S → AA
1

S → aA / b
2 3

uction then ⇒ accept

parse table SLR (1) :- → no. of. Look aheads

	action			Goto	
	a	b	$	A	S
0	s_3	s_4		2	1
1			accept		
2	s_3	s_4		5	
3	s_3	s_4		6	
4	r_3	r_3	r_3		
5			r_1		
6	r_2	r_2	r_2		

follow(A) = {a, b, $}

follow(S) = {$}

place reduce in follow of

$A \rightarrow b.$

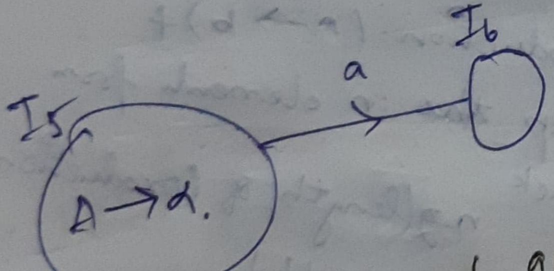
$S \rightarrow AA.$

$A \rightarrow aA.$

more blank entries → more error detecting speed

SLR(1) → more powerful than LR(0)

Condition for non LR(0) grammar :-



	action	
	a	b
5	s_6/r_1	r_1

sr conflict

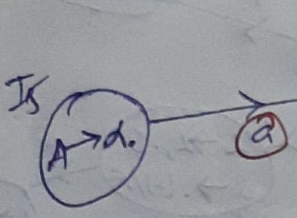
I5 $A \rightarrow \alpha.$ ① $B \rightarrow \beta.$ ②

	action		
	a	b	c
5	r_1/r_2	r_1/r_2	r_1/r_2

rr conflict

$\boxed{\text{more than 1 final items}}$

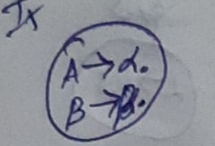
Condition for Non-SLR(1) grammar:-



	action	
	a	b
5	S6/r1	

Sr conflict

only if Follow(A) = {a}

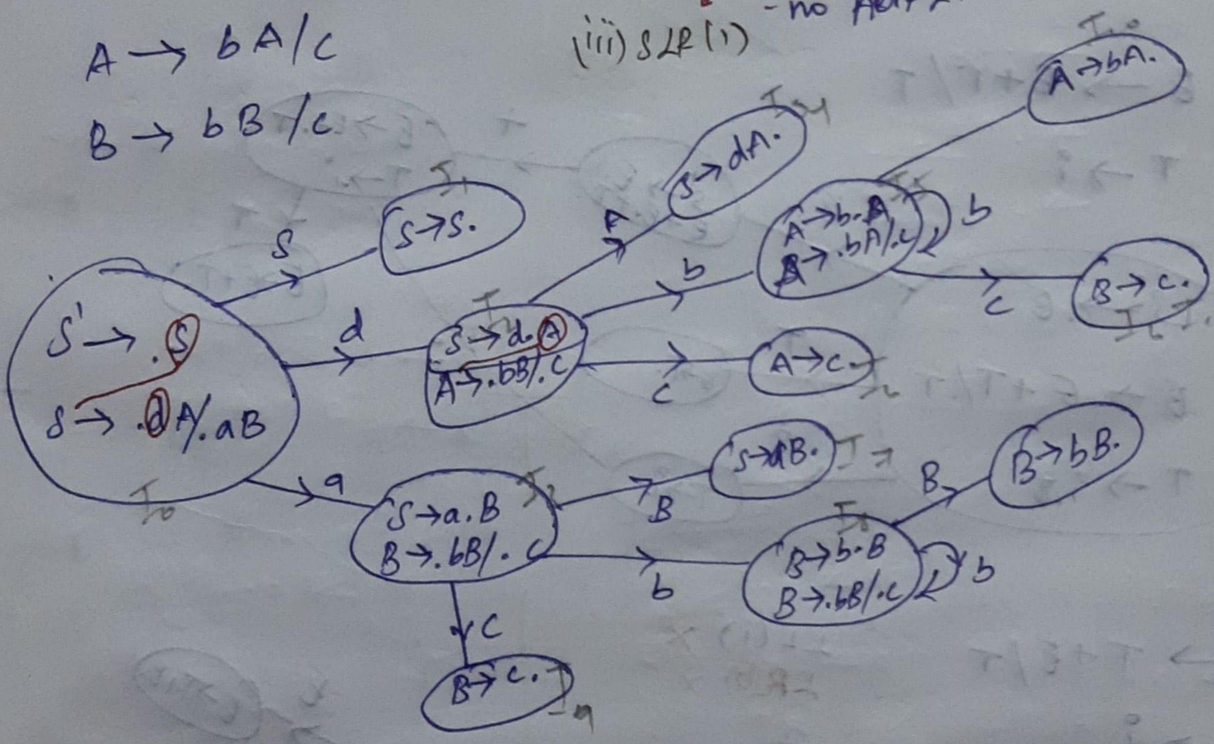


rr conflicts occurs only if Follow(A) ∩ Follow(B) ≠ ϕ

Examples:-

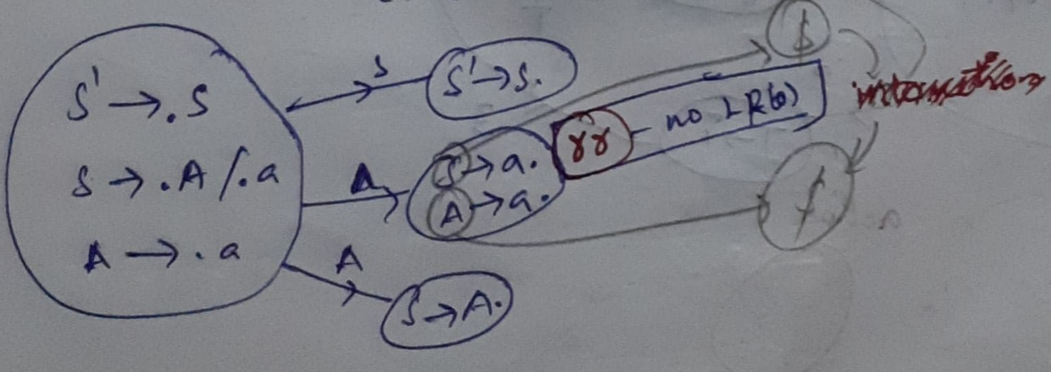
1. S → dA / aB
A → bA / c
B → bB / c

(i) LL(1) ✓
(ii) LR(0) - no sr & rr ⇒ ✓
(iii) SLR(1) - no LR(0) ⇒ ✓ obvious



2. S → A/a
A → a

(i) LL(1) ✗
(ii) LR(0) ✗
(iii) SLR(1) ✗



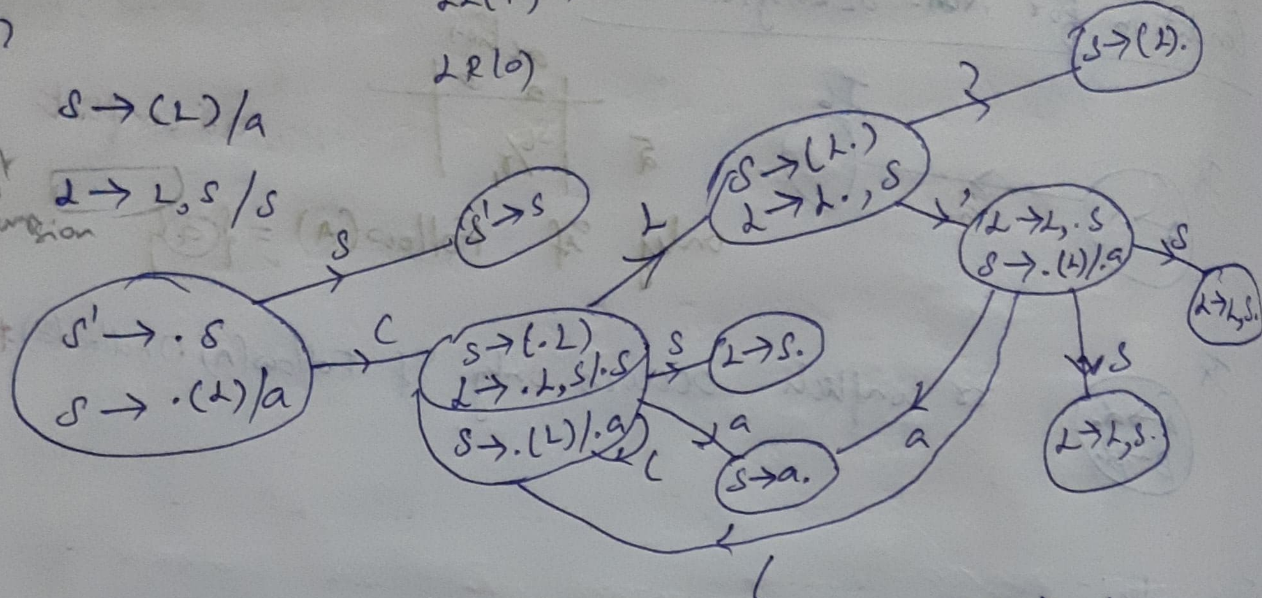
3)

LL(1) ✗

LR(0)

$S \to (L)/a$

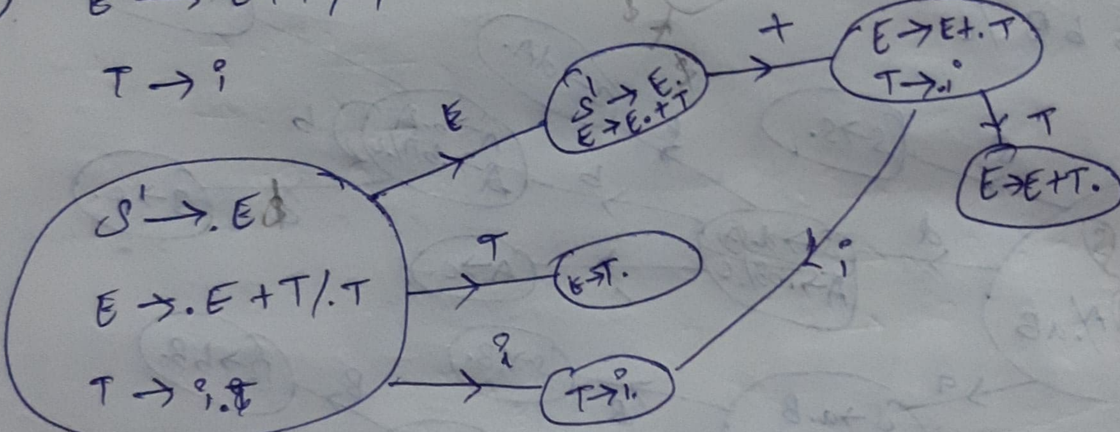
left recursion $L \to L,S/S$



no SR + RR conflict → hence its LR(0) ∴ it is SLR(1) also

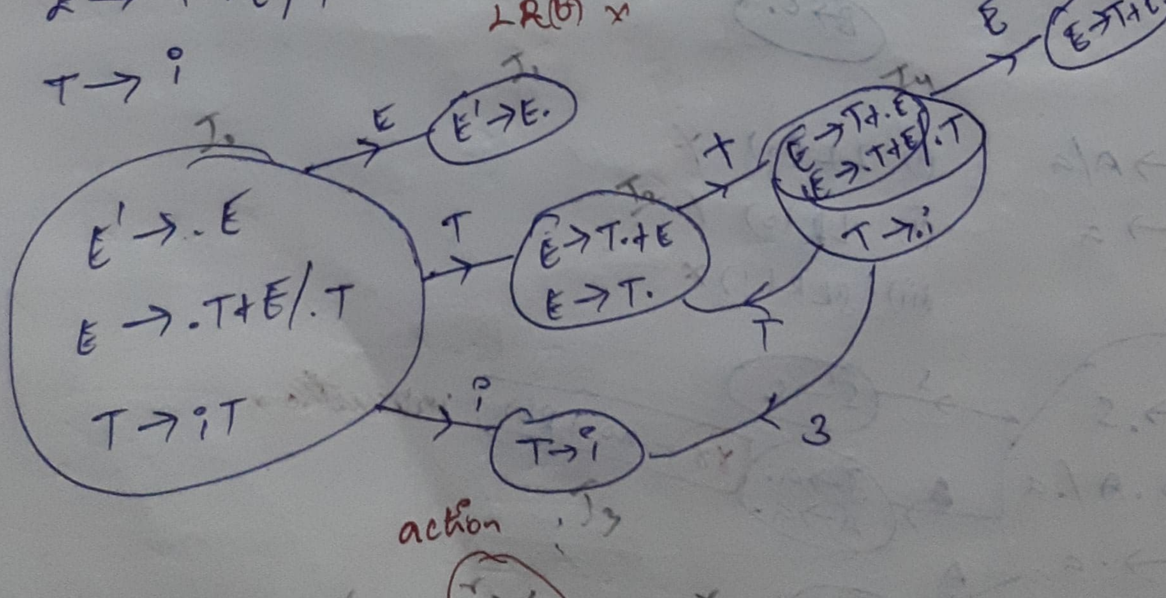
4) $E \to E+T/T$

$T \to i$



$E \to T+E/T$ LL(1) ✗ LR(0) ✗

$T \to i$



action

2 r_2 r_2/S_4 r_2

Follow of (E) = { $ }

Follow of (F) = { $, + }

$L0	i	+	$	E	T
0	S3			1	2
1			accepts		
2		S4	r2		
3		r3	r3		
4	S3			5	2
5			r1		

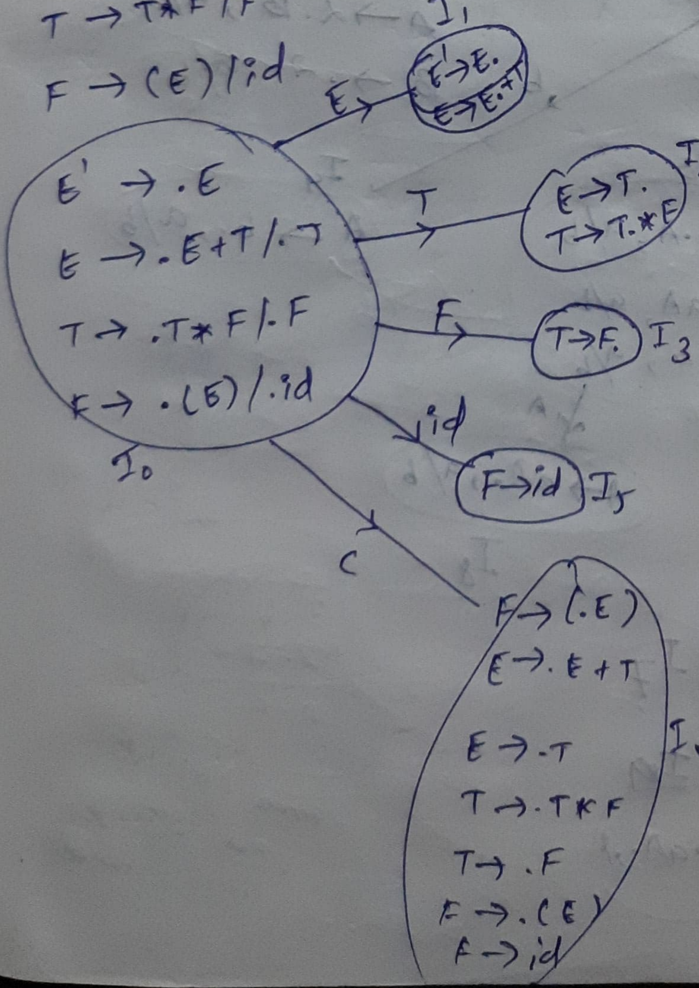
This grammar is SLR(1)

E' → E

E → E+T / T

T → T*F / F

F → (E) / id



LALR(1)   CLR(1)

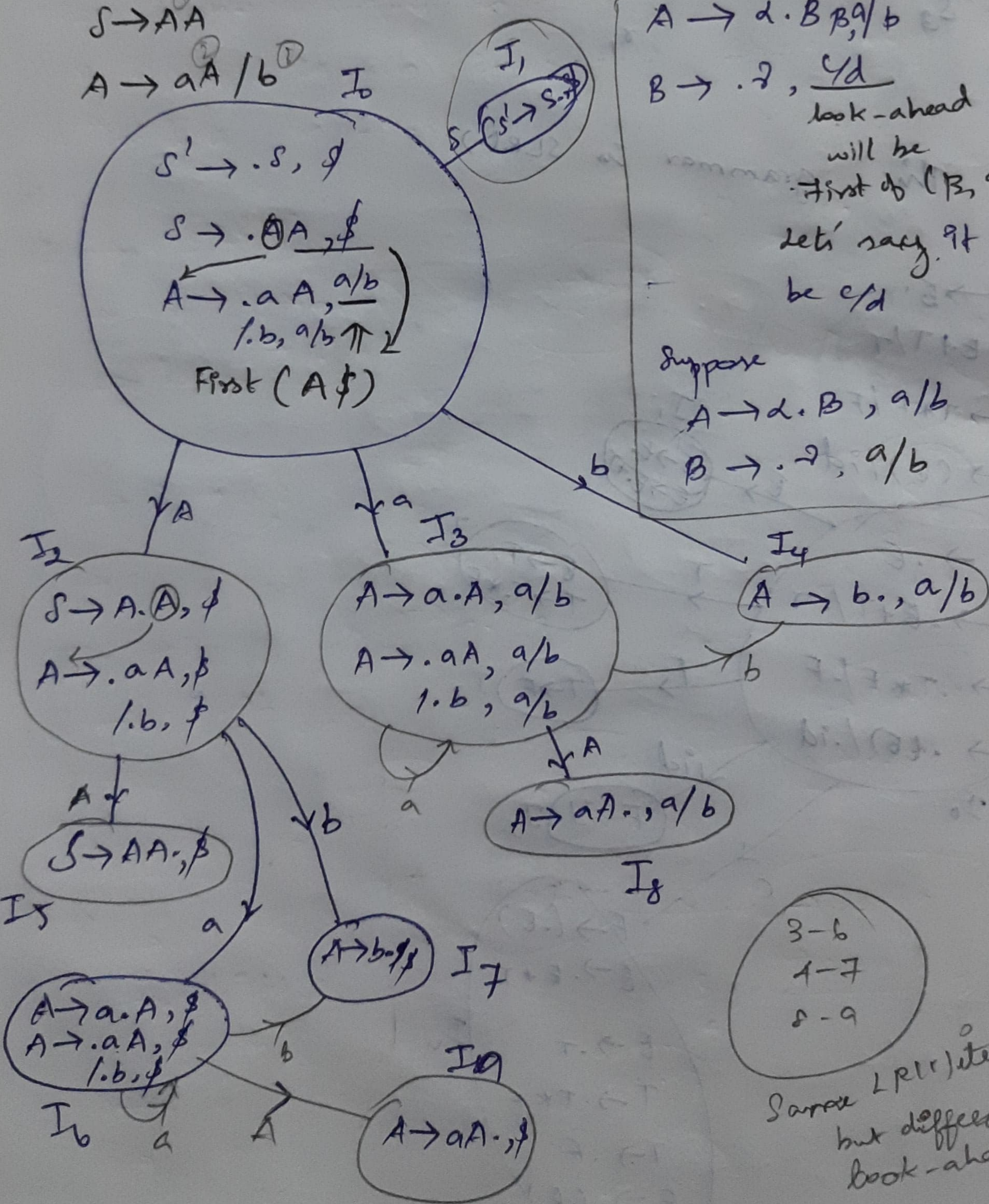
LR(1) item = LR(0) item + look ahead

$S \rightarrow .aA,\ a/b$ → look ahead

$\boxed{S \rightarrow aA.\ ,\ c/d}$ ⇒ implies that put reduce move only in look-ahead not in entire row or in First of LHS

$S \rightarrow AA$ (1)

$A \rightarrow aA$ (2) $/ b$ (3)



$A \rightarrow \alpha.B\beta, a/b$

$B \rightarrow .\gamma,\ \underline{c/d}$

look-ahead will be First of $(\beta, a/b)$ let's say it would be c/d

Suppose

$A \rightarrow \alpha.B,\ a/b$

$B \rightarrow .\gamma,\ a/b$

3-6

4-7

8-9

Same LR(1) items but different look-ahead

CLR(1) parsing table:

States	Action a	Action b	Action $	Goto S	Goto A
0	S_3	S_4			2
1					
2	S_6	S_7			5
3	S_3	S_4			8
4	r_3	r_3			
5			r_1		
6	S_6	S_7			9
7			r_3		
8	r_2	r_2			
9			r_2		

7 states in LR(0), SLR(1)

10 states in CLR(1)

No. of states in CLR(1) $\geq$ LR(0) = SLR(1) = LALR(1)

LALR(1):-

$I_3 + I_6 \Rightarrow I_{36}$

$I_4 + I_7 \Rightarrow I_{47}$

$I_8 + I_9 \Rightarrow I_{89}$

States	Action a	Action b	Action $	Goto S	Goto A
0	S_3	S_4			2
1					
2	S_6	S_7			5
36	S_3	S_4			89
47	r_3	r_3	r_3		
5			r_1		
89	r_2	r_2	r_2		

SR conflict

$A \to \alpha . a\beta , c/d$

$B \to \gamma . , a/\$$

RR conflict

$A \to \alpha . , \boxed{a}$

$B \to \alpha . , \boxed{a}$

* if the grammar is CLR(1), then it might be LALR(1).
* if there is no SR conflict in CLR(1), then there is no SR conflict in LALR(1) also.
* if no rr conflicts in CLR(1), there could be rr conflicts in LALR(1)

CLR(1)

$A \to \alpha . , a$
$B \to \beta . , b$

$A \to \alpha . , b$
$B \to \beta . , \alpha$

$A \to \alpha . , \boxed{a/b}$
$B \to \beta . , \boxed{a/b}$

$S \to AaAb / BbBa$

$A \to \varepsilon$

$B \to \varepsilon$

$S' \to .S$
$S \to .AaAb , \$$
$\quad / .BbBa , \$$
$A \to . , a$
$B \to . , b$

$\xrightarrow{S}$ $S' \to S.$

$\xrightarrow{A}$ $S \to A.aAb , \$$ $\xrightarrow{a}$ $S \to Aa.Ab , \$$, $A \to . , b$ $\xrightarrow{A}$ $S \to AaA.b , \$$ $\xrightarrow{b}$ $S \to AaAb. , \$$

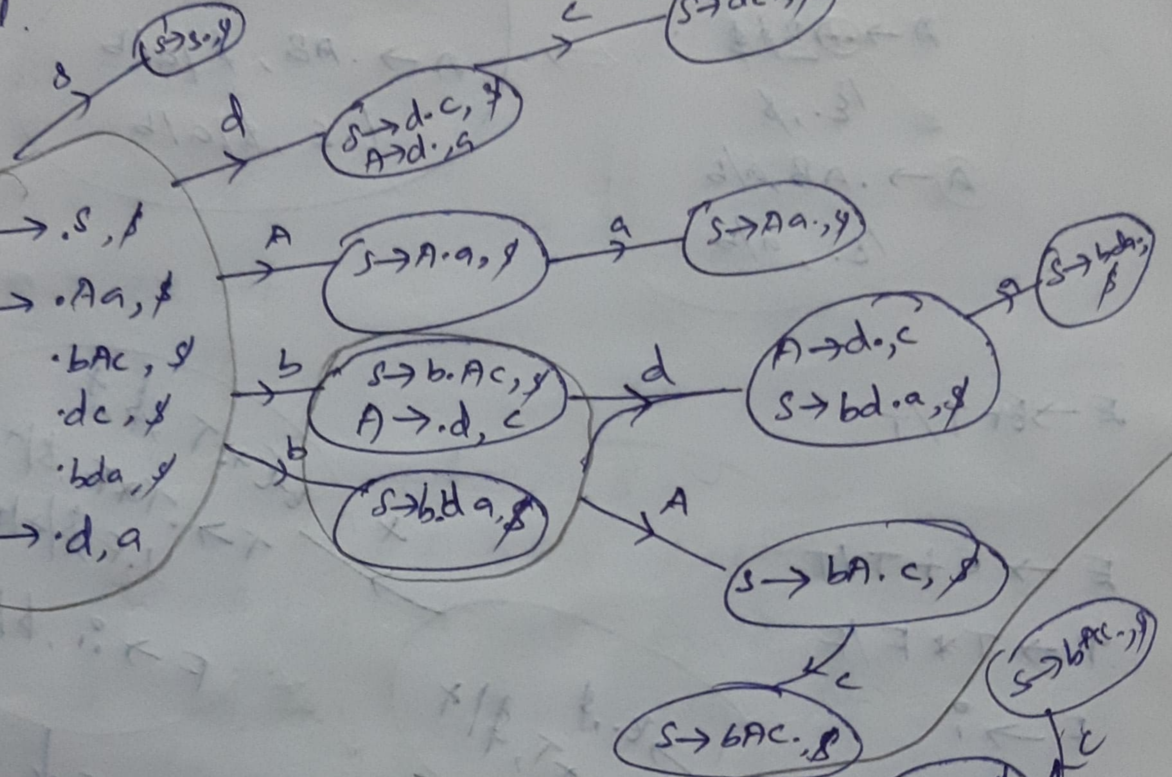
$\xrightarrow{B}$ $S \to B.bBa , \$$ $\xrightarrow{b}$ $S \to Bb.Ba , \$$, $B \to . , a$ $\xrightarrow{B}$ $S \to BbB.a , \$$ $\xrightarrow{a}$ $S \to BbBa. , \$$

This grammar is both CLR(1) & LALR(1)

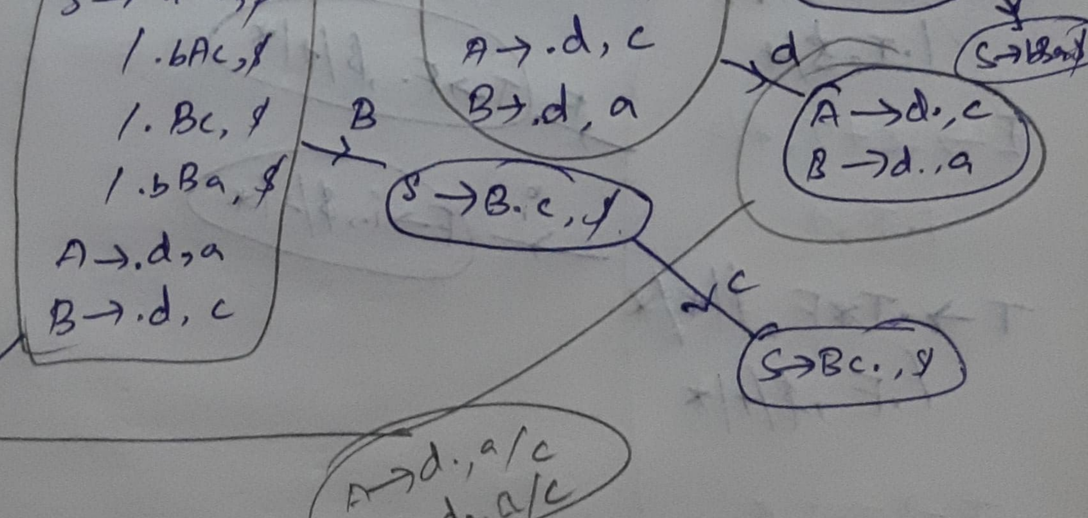
No. of states in $\boxed{CLR(1) = LALR(1)}$

$S \to Aa \mid bAc \mid dc \mid bda$

$A \to d.$



$S \to Aa$
$/ bAc$
$/ Bc$
$/ bBa$
$A \to d$
$B \to d$



no rr conflict in CLR(1)

but rr conflict in LALR(1)

S → A

A → AB/ε

B → aB/b

closure(S → .A, $)

S → .A, $

A → .AB, $

/ε., $

A → .AB, a/b

/ε., a/b

⟺

S → .A, $

A → .AB, $/a/b

/.ε, $/a/b

E → E+T/T

E → E+T/T

T → T*F/F

F → i

~~E → E. E~~

E' → .E, $

E → .E+T, $/+

/.T, $/+

E → .E+T, +

/.T, +

T → .T*F, $/+/*

/.F, $/+/*

T → .T*F, *

/.F, *

F → .i, $/+/*

E

E' → E., $

E → E.+T, $/+

+

E → E+.T, $/+

T → .T*F, $/+/*

/.F, $/+/*

F → .i, $/+/*

T

E → E+T., $/+

T → T.*F, $/+/*

next set

T

T → T.*F, $/+/*

E → T., $/+

F

T → F., $/+/*

i

F → i., $/+/*

$S \rightarrow A$

$A \rightarrow AB / \varepsilon$

$B \rightarrow aB / b$

$S' \rightarrow .S , \$$

$S \rightarrow .A , \$$

$A \rightarrow \cancel{\$}\varepsilon . , \$$

$/ \; .AB , \$$

$A \rightarrow .AB , a/b$

$/ . \varepsilon , a/b$

$\Rightarrow$

$S' \rightarrow S. , \$$

S

$S' \rightarrow .S , \$$

$S \rightarrow .A , \$$

$A \rightarrow .AB , \$/a/b$

$/ \varepsilon . , \$/a/b$

A

$S \rightarrow A. , \$$

$A \rightarrow A.B , \$/a/b$

$B \rightarrow .aB , \$/a/b$

$/ .b , \$/a/b$

B

$A \rightarrow AB. , \$/a/b$

b

$B \rightarrow b. , \$/a/b$

a

$B \rightarrow a.B , \$/a/b$

$B \rightarrow .aB , \$/a/b$

$/ .b , \$/a/b$