LL(1)

CFG	First	Follow
$S \to L=R \mid abR \mid \epsilon$	$\{*, id, a, \epsilon\}$	$\{\$\}$
$L \to *R \mid id$	$\{*, id\}$	$\{\$, =\}$
$R \to L \mid \epsilon$	$\{*, id, \epsilon\}$	$\{\$, =\}$

LL(1) parsing Table:

NT \ T	=	a	b	*	id	$
S		$S \to abR$		$S \to L=R$	$S \to L=R$	$S \to \epsilon$
L				$L \to *R$	$L \to id$	
R	$R \to \epsilon$			$R \to L$	$R \to L$	$R \to \epsilon$

Here, there is no conflict in the parsing table because there is no multiple entries in a single cell of the table. So the CFG is LL(1)

Input string = * id = id $

Stack	Input	Action
$ S	* id = id $	S→L = R
$ R=L	* id = id $	L→ * R
$ R = R*	* id = id $	Match pop scan next
$ R = R	id = id $	R→L
$ R = L	id = id $	L→id
$ R = id	id = id $	Match pop scan next
$ R =	= id $	Match pop scan next
$ R	id $	R→L
$ L	id $	L→id
$ id	id $	Match pop scan next
$	$	Accept

So here we say that input string * id = id parsed with the LL(1) parser.

SLR(1)

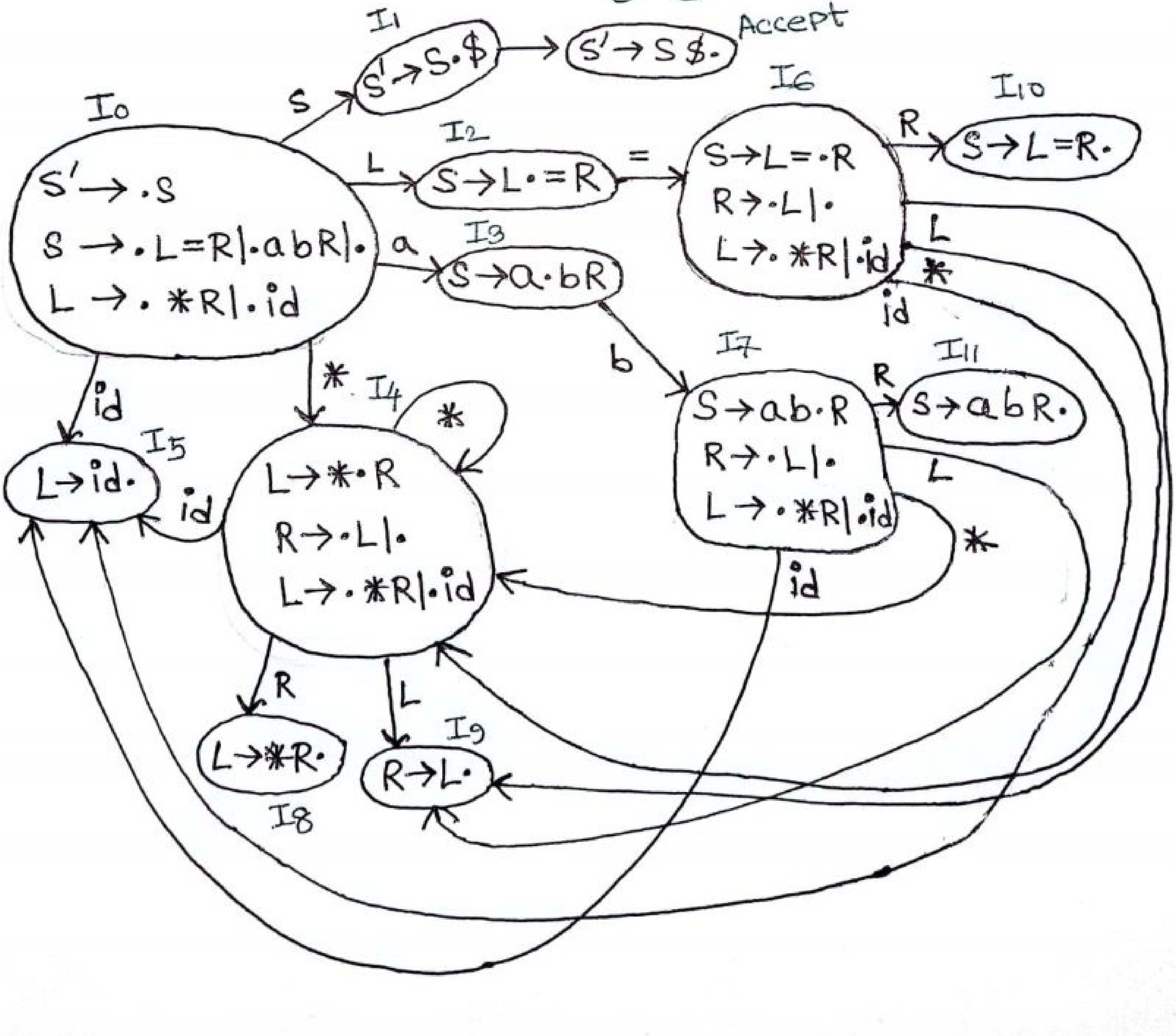
Grammer	First	Follow
S → L=R | abR | ∈	{*, id, a, ∈}	{$}
L → *R | id	{*, id}	{$, =}
R → L | ∈	{*, id, ∈}	{$, =}

Augmented grammer → S' → S ⓪

S → L=R ① | abR ② | ∈ ③

L → *R ④ | id ⑤

R → L ⑥ | ∈ ⑦



SLR(1) Parsing Table :

	Action						GoTo		
State	=	a	b	*	id	$	S	L	R
0		S3		S4	S5	r3	1	2	
1						Accept			
2	S6								
3			S7						
4	r7			S4	S5	r7		9	8
5	r5					r5			
6	r7			S4	S5	r7		9	10
7	r7			S4	S5	r7		9	11
8	r4					r4			
9	r6					r6			
10						r1			
11						r2			

Here, there is no conflict in the parsing table because there is no multiple entries in a single cell of the table. So the CFG is SLR(1)

Input string → *id = id

Stack	input	Action
0	* id = id $	Shift 4
0*4	id = id $	Shift 5
0*4id5	= id $	Reduce L→id
0*4L9	= id$	Reduce R→L
0*4 R8	= id$	Reduce L→*R
0L2	= id $	Shift 6
0L2=6	id $	Shift 5
0L2 =6id5	$	Reduce L→id
0L2 =6L9	$	Reduce R→L
0L2 =6 R10	$	Reduce S→L=R
0S1	$	Accept

So, here we say that input string * id = id parsed with the SLR(1) parser

CLR(1)

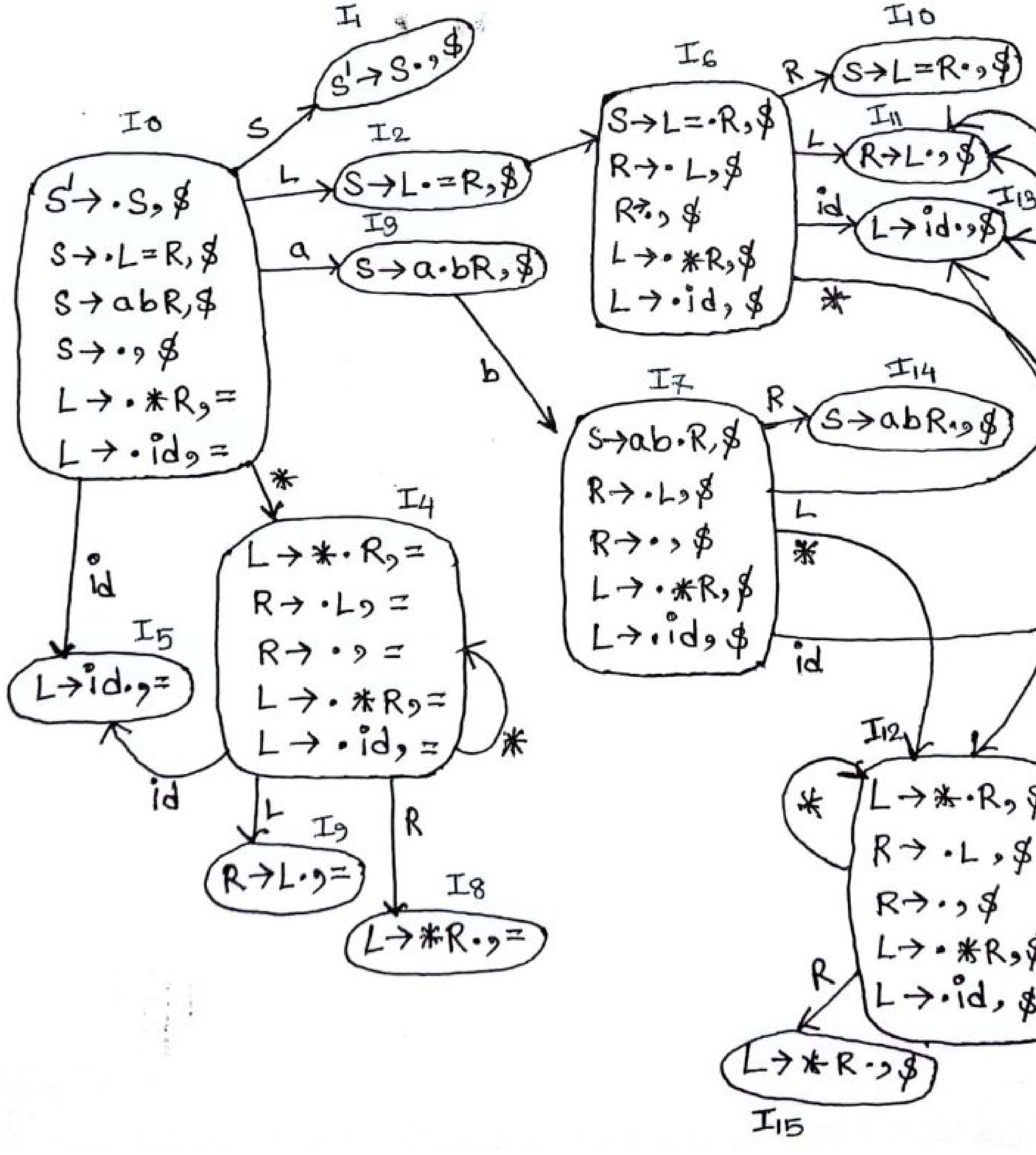
CFG	First	Follow
S → L=R | abR | ∈	{*, id, a, ∈}	{$}
L → *R | id	{*, id}	{$, =}
R → L | ∈	{*, id, ∈}	{$, =}

Augmented grammer → S' → S ⓪

S → L①=R | a②bR | ∈ ③

L → ④*R | id ⑤

R → ⑥L | ∈ ⑦



CLR(1) Parsing Table:

	Action						Goto		
State	=	a	b	*	id	$	S	L	R
0		S3		S4	S5	r3	1	2	
1						Accept			
2	S6								
3			S7						
4	r7			S4	S5			9	8
5	r5								
6				S12	S13	r7		11	10
7				S12	S13			11	14
8	r4								
9	r6								
10						r1			
11						r6			
12				S12	S13	r7		11	15
13						r5			
14						r2			
15						r4			

Here, there is no conflict in the parsing table because there is no mutiple entries in a single cell of the table. So the CFG is CLR(1).

Input string → *id=id

Stack	Input	Action
0	*id=id$	Shift 4
0*4	id=id$	Shift5
0*4id5	=id$	reduce L→id
0*4L9	=id$	reduce R→L
0*4R8	=id$	reduce L→*R
0L2	=id$	Shift 6
0L2=6	id$	Shift13
0L2=6id13	$	reduce L→id
0L2=6L11	$	reduce R→L
0L2=6R10	$	reduce S→L=R
0S1	$	Accept

So, here we say that input string *id=id parsed with CLR(1) parser.

LALR(1)

$S \to L = R \mid abR \mid \epsilon$

$L \to *R \mid id$

$R \to L \mid \epsilon$

In CLR(1) items, the number of states has increased compared with LR(0) items.

In terms of LR(0) items both I_4 and I_{12} are same.
Also both I_5 and I_{13} are same. Again $I8$ and $I15$ are same. similarly both $I9$ and $I11$ are same.

So in LALR(1) parsing table, we marge these states.

- $I_4, I_{12} \longrightarrow I_{412}$
- $I_5, I_{13} \longrightarrow I_{513}$
- $I_9, I_{11} \longrightarrow I_{911}$
- $I_8, I_{15} \longrightarrow I_{815}$

LALR(1) Parsing Table:

	Action						Goto		
State	=	a	b	*	id	$	S	L	R
0		S3		S412	S513	r3	1	2	
1						Accept			
2	S6								
3			S7						
412	r7			S412	S513	r7		911	815
513	r5					r5			
6				S412	S513	r7		911	10
7				S412	S513			911	14
815	r4					r4			
911	r6					r6			
10						r1			
14						r2			

Here there is no conflict in the parsing table because there is no multiple entries in a single cell of the table. So the CFG is LALR(1)

Input String : *id=id

Stack	Input	Action
0	*id=id $	Shift 412
0*412	id=id $	Shift 513
0*412 id 513	= id $	Reduce L→id
0*412 L 911	= id $	Reduce R→L
0*412 R 815	=id $	Reduce L→*R
0 L 2	= id $	Shift 6
0 L 2 = 6	id $	Shift 513
0 L 2 = 6 id 513	$	Reduce L→id
0 L 2 = 6 L 911	$	Reduce R→L
0 L 2 = 6 R 10	$	Reduce S→L=R
0 S 1	$	Accept

So, here we say that input string *id = id parsed with LALR(1) parser.