-> Given Grammar is

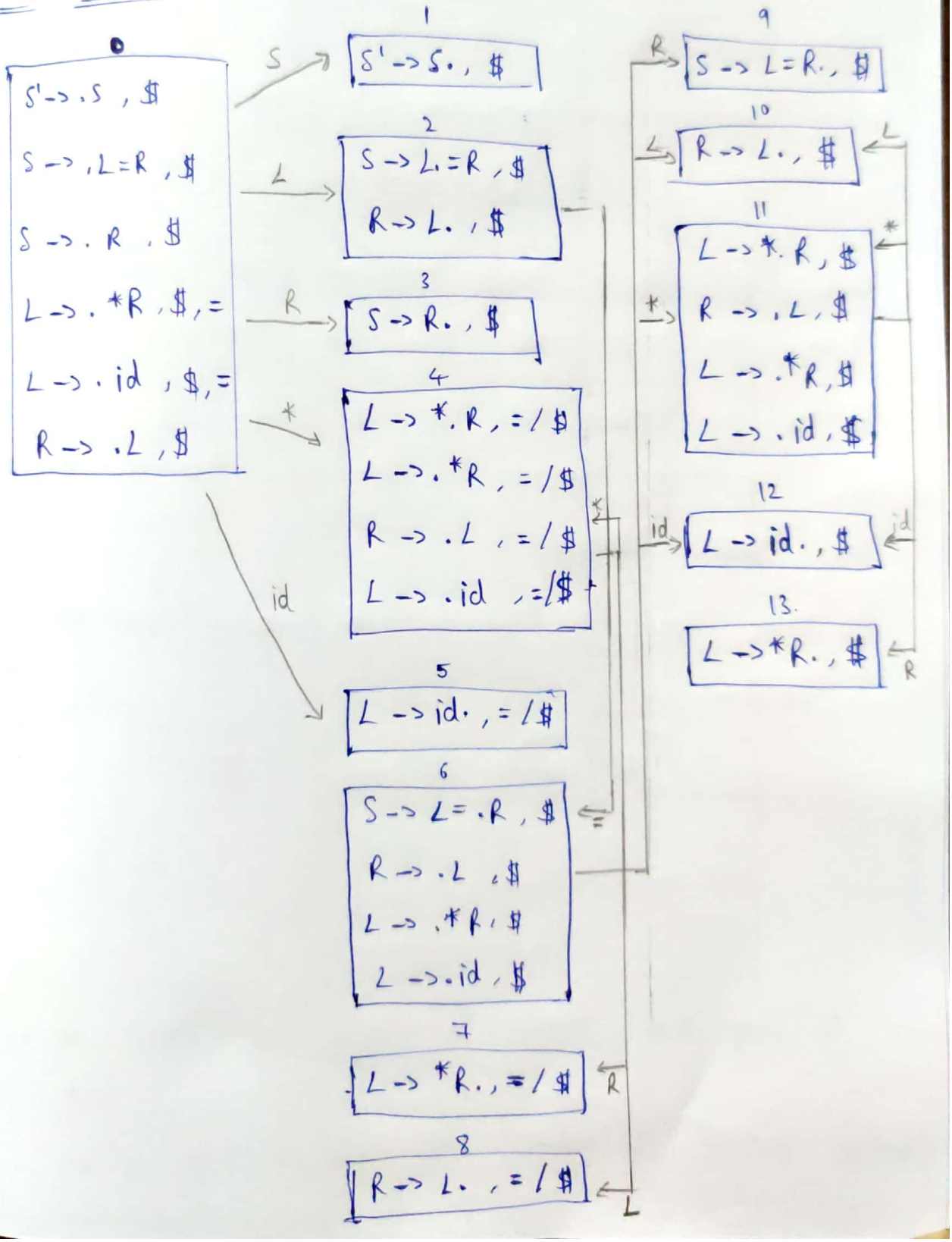
S' -> S

S -> L = R | R

L -> *R | id

R -> L

LR(1) automaton :-



Scanned with CamScanner

The parse table for the above LR(1) automaton is

State	$	*	=	id	L	R	S
0		S_4		S_5	S_2	S_3	S_1
1	Accept						
2	R->L		S_6				
3	S->R						
4		S_4		S_5	S_8	S_7	
5	L->id		L->id				
6		S_{11}		S_{12}	S_{10}	S_9	
7	L->*R		L->*R				
8	R->L		R->L				
9	S->L=R						
10	R->L						
11		S_{11}		S_{12}	S_{10}	S_{13}	
12	L->id						
13	L->*R						

-> We follow the concept of CLOSURE and GOTO of a given item.

For a LR(1) grammar, the closure of I is defined

Scanned with CamScanner

as, for (each item [A -> α.Bβ, a] in I)

for (each production B -> β in G')

for (each terminal b in First(βa))

add [B -> .β, b] to set I.

do until no more productions are added to closure

Goto of a set is defined as.

for (each item [A -> α.Xβ, a] in I)

add item [A -> αX.β, a] to set J.

With the help of closure and Goto we compute LR(1) items for each states and the corresponding DFA.

In the parse table, goto set is used for shift transition and we can reduce for the production ~~if~~ only for the given lookahead.

As there are no S-R (or) R-R conflicts the grammar is LR(1).

Scanned with CamScanner