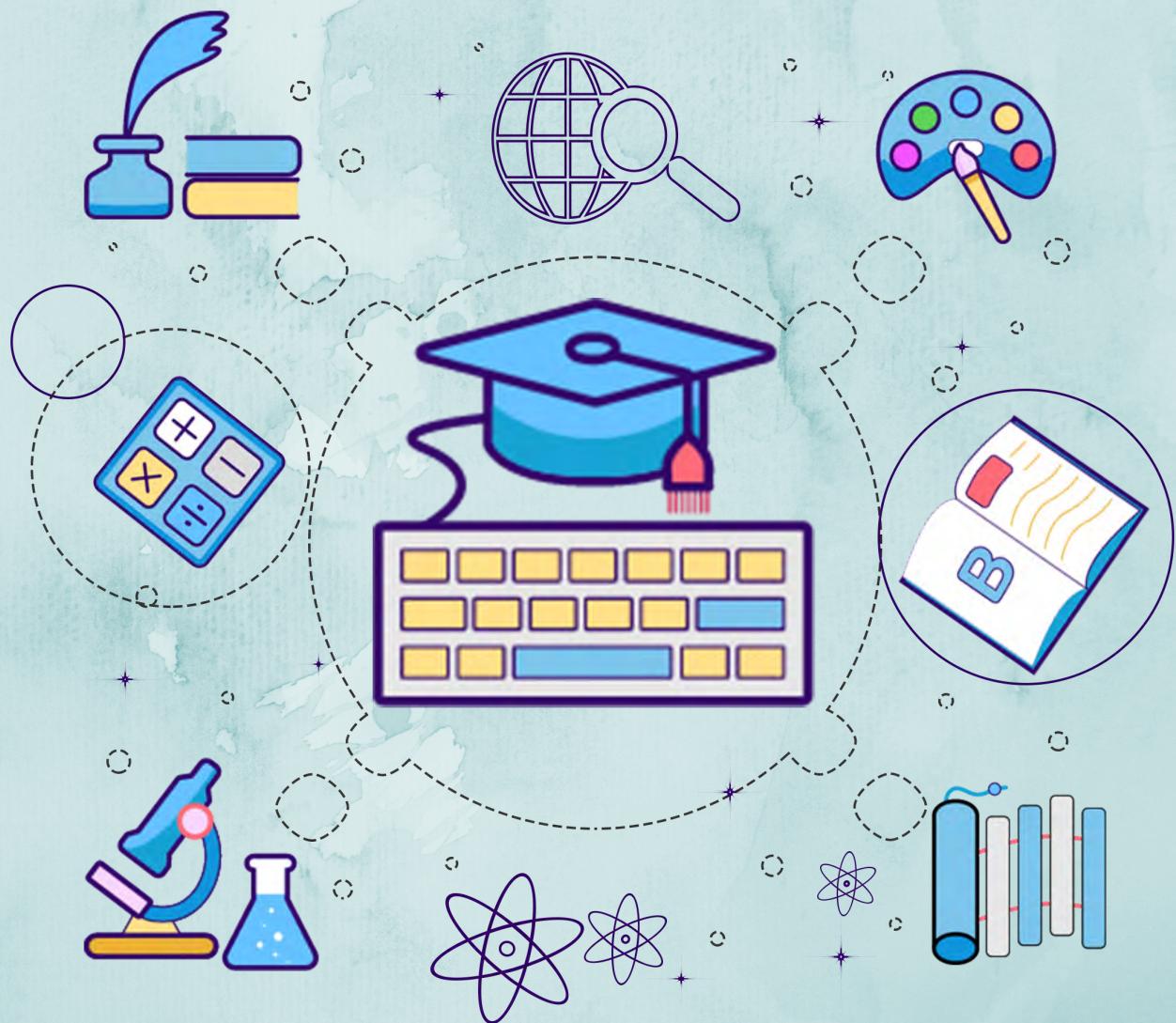


**APJ ABDUL KALAM TECHNOLOGICAL UNIVERSITY**

# Kerala Notes



**SYLLABUS | STUDY MATERIALS | TEXTBOOK**

**PDF | SOLVED QUESTION PAPERS**



## KTU STUDY MATERIALS

### COMPILER DESIGN

### CST 302

# Module 3

#### Related Link :

- KTU S6 CSE NOTES | 2019 SCHEME
- KTU S6 SYLLABUS CSE | COMPUTER SCIENCE
- KTU PREVIOUS QUESTION BANK S6 CSE SOLVED
- KTU CSE TEXTBOOKS S6 B.TECH PDF DOWNLOAD
- KTU S6 CSE NOTES | SYLLABUS | QBANK | TEXTBOOKS DOWNLOAD

## MODULE-3

### BOTTOM UP PARSING

#### Syllabus

Handle pruning. shift Reduce parsing. Operator precedence parsing (concept only). LR Parsing - constructing SLR, LALR and canonical LR Parsing tables.

#### Bottom up parsing

2 types :- operator precedence.  
LR parser.

Operator precedence can handle ambiguous grammar. A general form of bottom-up syntax analysis is called shift reducing parsing.

Operator precedence parsing is a form of shift reducing parsing.

Consider the grammar :

$$S \rightarrow aABe$$

$$A \rightarrow Abc/b$$

$$B \rightarrow d.$$

Consider the sentence abbcde.

a b b cde

a A b cde

a A d e

a A B e

→ S

Handle

A handle of a string is a substring that matches the right side of a production, and whose reduction to the non-terminal on the left side of the production represents one step along the reverse of a right most derivation.

Stack implementation of shift reducing parser

$$\begin{aligned} E &\rightarrow E+E \\ &\rightarrow E \cdot E \\ &\rightarrow C E ) \\ &\rightarrow E I E \\ &\rightarrow id \end{aligned}$$

Stack	input	Actions
\$	id + id * id \$	shift
\$ id	+ id * id \$	Reduce
\$ E	+ id * id \$	shift
\$ E +	id * id \$	shift
\$ E + id	* id \$	Reduce.
\$ E + E	* id \$	Reduce shift
\$ E + E *	id \$	shift
\$ E + E * id	\$	Reduce
\$ E + E + E	\$	Reduce
\$ E + E	\$	Reduce.
\$ E	\$	

Implement the shift reduce grammar for the input string id + id + id

Stack	Input	Action.
\$	id + id + id \$	Shift
\$ id	+ id + id \$	Reduce.
\$ E	+ id + id \$	Shift
\$ E +	id + id \$	Shift
\$ E + id	+ id \$	Reduce.
\$ E + E	+ id \$	Shift
\$ E + E +	id \$	Shift
\$ E + E + id	\$	Reduce.
\$ E + E + E	\$	Reduce
\$ E + E	\$	Reduce.
\$ E	\$	

The primary operation of the parser are shift & reduce.  
 Those are actually <sup>4</sup> possible actions a shift-reduce parser can make.

- 1) Shift - In a shift action the next i/p symbol is shifted onto the top of the stack.
- 2) In a reduce action, the parser know the right end of the handle is at the top of the stack. It must locate the left end of the handle within the stack and decide with what non-terminal to replace the handle.
- 3) In an accept action the parser announces successful completion

## 1. parsing -

In an error actions the parser discovers that a syntax error has occur and it will called a error recovery routine.

## Conflicts during shift Reduce parser

Every shift reduce parser can reach a configuration in which a parser, knowing the entire stack contents and the next I/p symbol  $\epsilon$  can't decide whether to shift or to reduce (called as shift-reduce conflict or S-R<sup>1</sup>) Cannot decide which of the several reductions to be made called as the reduce-reduce conflict

## Operator precedence parsing.

### Operator grammar

A grammar is said to be operator grammar  $\Leftrightarrow$  when it has  $\geq 2$   $\epsilon$  production on the right hand side or has  $\geq 2$  adjacent non-terminal.

Consider the following grammar

$$E \rightarrow EA\epsilon \mid CE\epsilon \mid -E \mid id$$

$$A \rightarrow + \mid - \mid * \mid / \mid \uparrow$$

This is not an operator grammar. because it has two adjacent & non-terminal  $EA, AE$

So it can convert into a operator grammar.

So :- Operator grammar handle-

Property :- operator grammar can handle ambiguous grammar.

### Operator Precedence relation. table

$\text{id} + \text{id} * \text{id} \$$

$\text{id}$	$+$	$*$	$\$$
-------------	-----	-----	------

$\text{id}$	$>$	$>$	$>$
$+$	$<$	$>$	$\leftrightarrow$
$*$	$<$	$>$	$>$
$\$$	$<$	$<$	$<$

$\text{id} + \text{id} * \text{id} \$$ .

$\$ < \text{id} \cdot > + < \text{id} \cdot > * < \text{id} \cdot > \$$

Q.  $\text{id} + \text{id} * \text{id} \$$

$\Rightarrow \text{id} + \text{id} * \text{id} \$$



$\text{id} + \text{id} * \text{id} \$$



Push  $\text{id}$   
Increment + 1 (+)

$\text{id} + \text{id} * \text{id} \$$

↑

\$	+	
----	---	--

pop id

reduce id to E

increment + 1 (id)

$\text{id} + \text{id} * \text{id} \$$

↑

\$	+	id
----	---	----

increment id

$\text{id} + \text{id} * \text{id} \$$

↑

\$	+	
----	---	--

reduce id to E

increment + 1

$\text{id} + \text{id} * \text{id} \$$

↑

\$	+	*	
----	---	---	--

increment + 1

$\text{id} + \text{id} * \text{id} \$$

↑

\$	+	*	id
----	---	---	----

reduce id to E

increment (\$)

E  
|  
 $\text{id} + \text{id} * \text{id} \$$

E E  
| |  
 $\text{id} + \text{id} * \text{id} \$$

E E E  
| | |  
 $\text{id} + \text{id} * \text{id} \$$

$\text{id} + \text{id} * \text{id} \$$

\$	+		*		
----	---	--	---	--	--

Pop \*

Push \*

Reduce (CE\*E + E)

\$	+	
----	---	--

Pop +

Push

Reduce CE\*E + E

Q.  $\text{id} + \text{id} * \text{id} \$$ .

→ 

\$		id		
----	--	----	--	--

 Reduce id

$\text{id} + \text{id} * \text{id} \$$

\$	*		
----	---	--	--

$\text{id} * \text{id} + \text{id} \$$ .

↑

\$	*		id
----	---	--	----

$\text{id} * \text{id} + \text{id} \$$

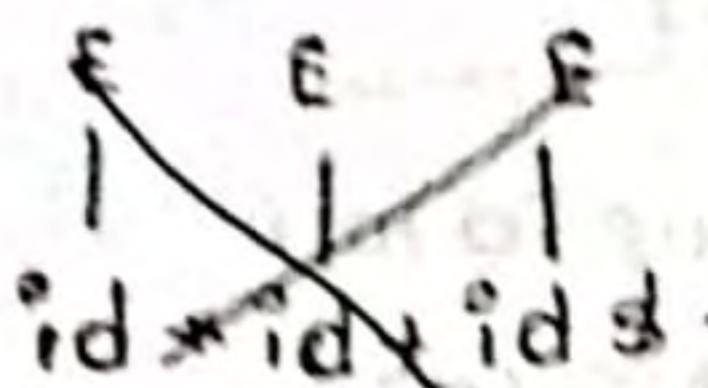
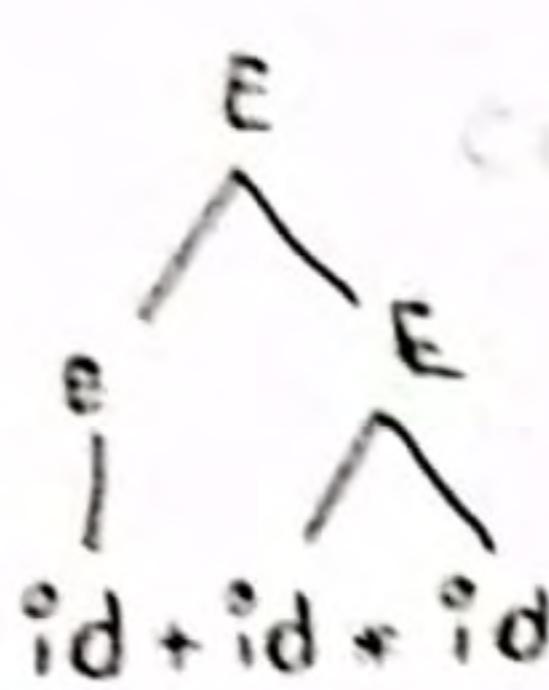
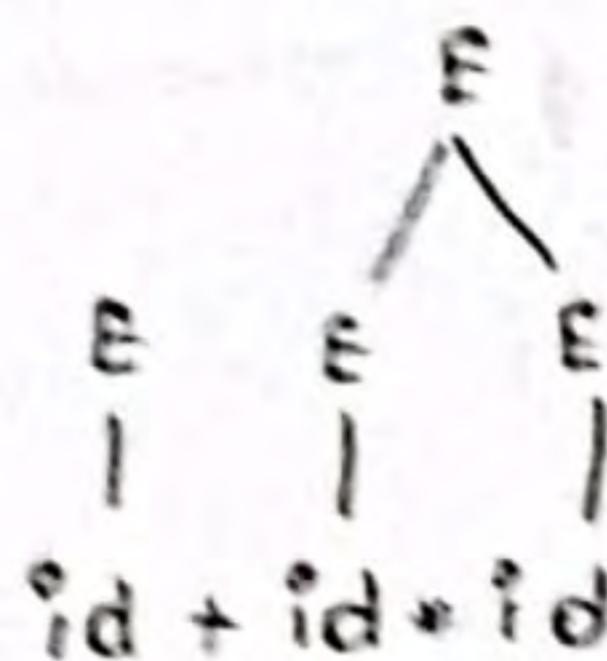
↑

\$	*		*
----	---	--	---

fill the stack

is higher precedence

so pop \* & push +

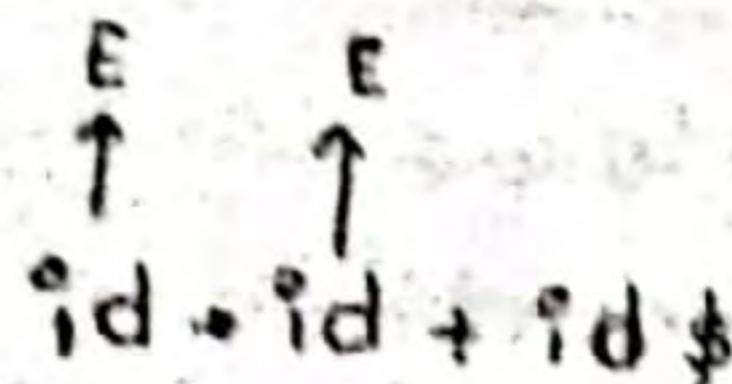


$\text{id} * \text{id} + \text{id} \$$

\$	-		+		id
----	---	--	---	--	----

$\text{id} * \text{id} + \text{id} \$$

A	-		+
---	---	--	---



$\text{id} * \text{id} + \text{id} \$$

```

graph TD
    E1[E] --> E2[E]
    E1 --> E3[E]
    E1 --> P[+]
    E2 --> ID1[id]
    E3 --> ID2[id]
    P --> ID3[id]
  
```

$\circ \text{id} + \text{id} + \text{id} \Downarrow$

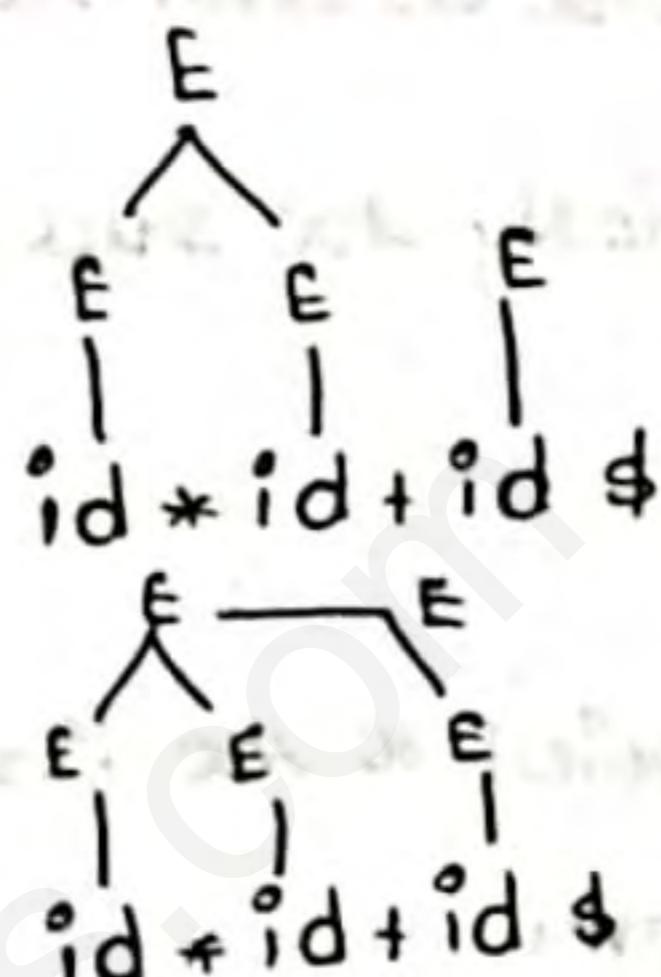
\$	+	id
----	---	----

$\text{id} * \text{id} + \text{id}$  \$  
↑

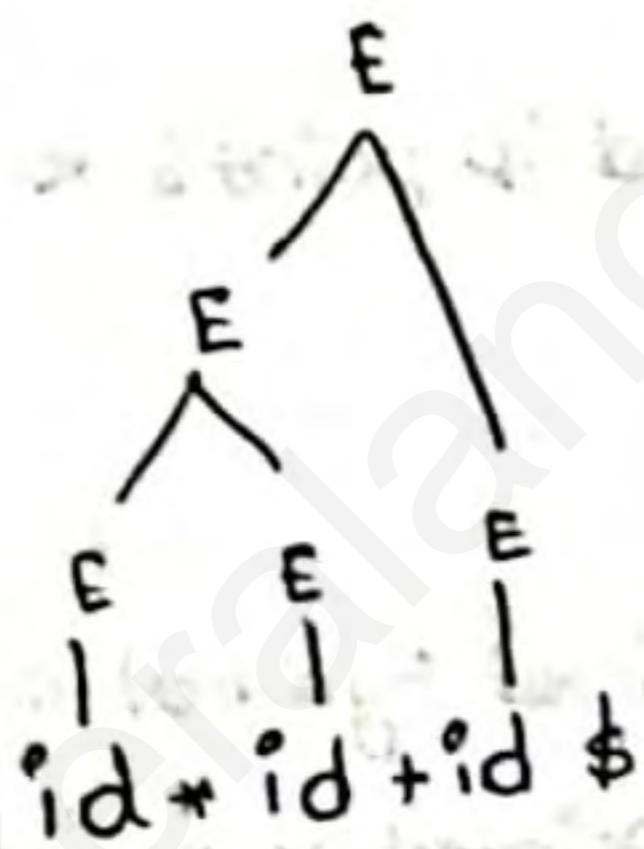
§ +

$\text{id} + \text{id} + \text{id}$  \$↑

A rectangular box divided into two sections by a vertical line. The left section contains a dollar sign (\$) symbol.



60



## Operator - Precedence parsing algorithm

Input :- An input string  $w$  and a table of precedence relations.

Output :- If  $w$  is a formed, a skeletal parse tree, with a place holder nonterminal  $E$  labeling all interior nodes;  
otherwise an error indication.

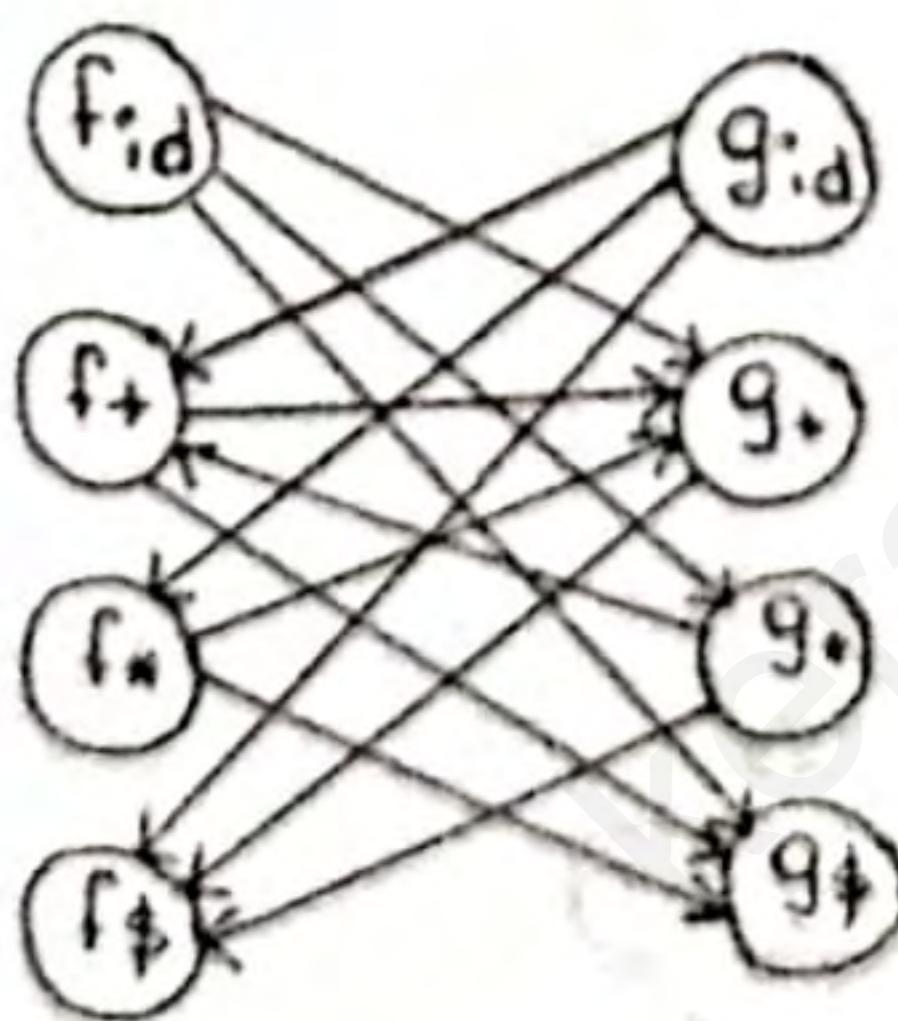
Method :- Initially the stack contains  $\$$  and the i/p string  $w\$$ .

Steps :-

- 1) set ip to point to the first symbol of  $w\$$ ;
- 2) repeat forever
- 3) if  $\$$  is on top of the stack and ip points to  $\$$  then
- 4) return
5. else begin
- 5) let  $a$  be the topmost terminal symbol on the stack  
and let  $b$  be the symbol pointed to by ip;
- 6) if  $a < b$  ( $<$  = less than) or  $a \equiv b$  then begin.
- 7) Push  $b$  onto the stack;
- 8) Advance ip to the next input symbol;
- end;
- 9) else if  $a > b$  ( $>$  = greater than) then / reduce.
- 10) repeat.
- 11) Pop the stack.
- 12) until the top stack terminal is related by to the  
terminal most recently popped.

13) else error  
end

<del>f</del> g →	id	+	*	\$
id	→	→	→	
+	<	→	<	→
*	<	→	→	→
\$	<	<	<	

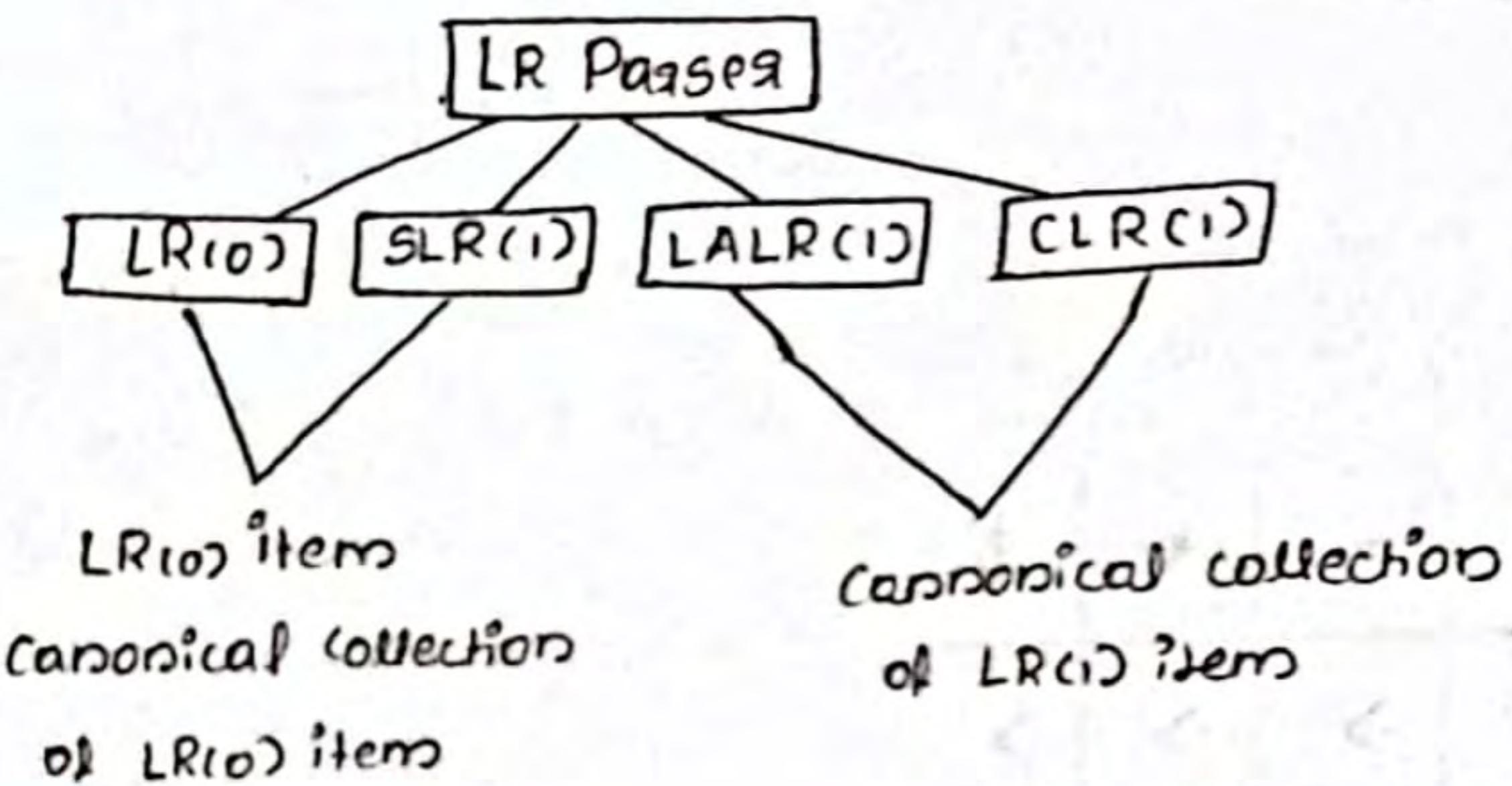


	+	*	id	\$
f	2	4	4	0
g	1	3	5	0

### LR Parser

LR Parser is classified into 4 :

- LR(0)
- SLR(1)
- LALR(1)
- CLR(0)



### → LR<sub>(0)</sub> parser

$$S \rightarrow AA$$

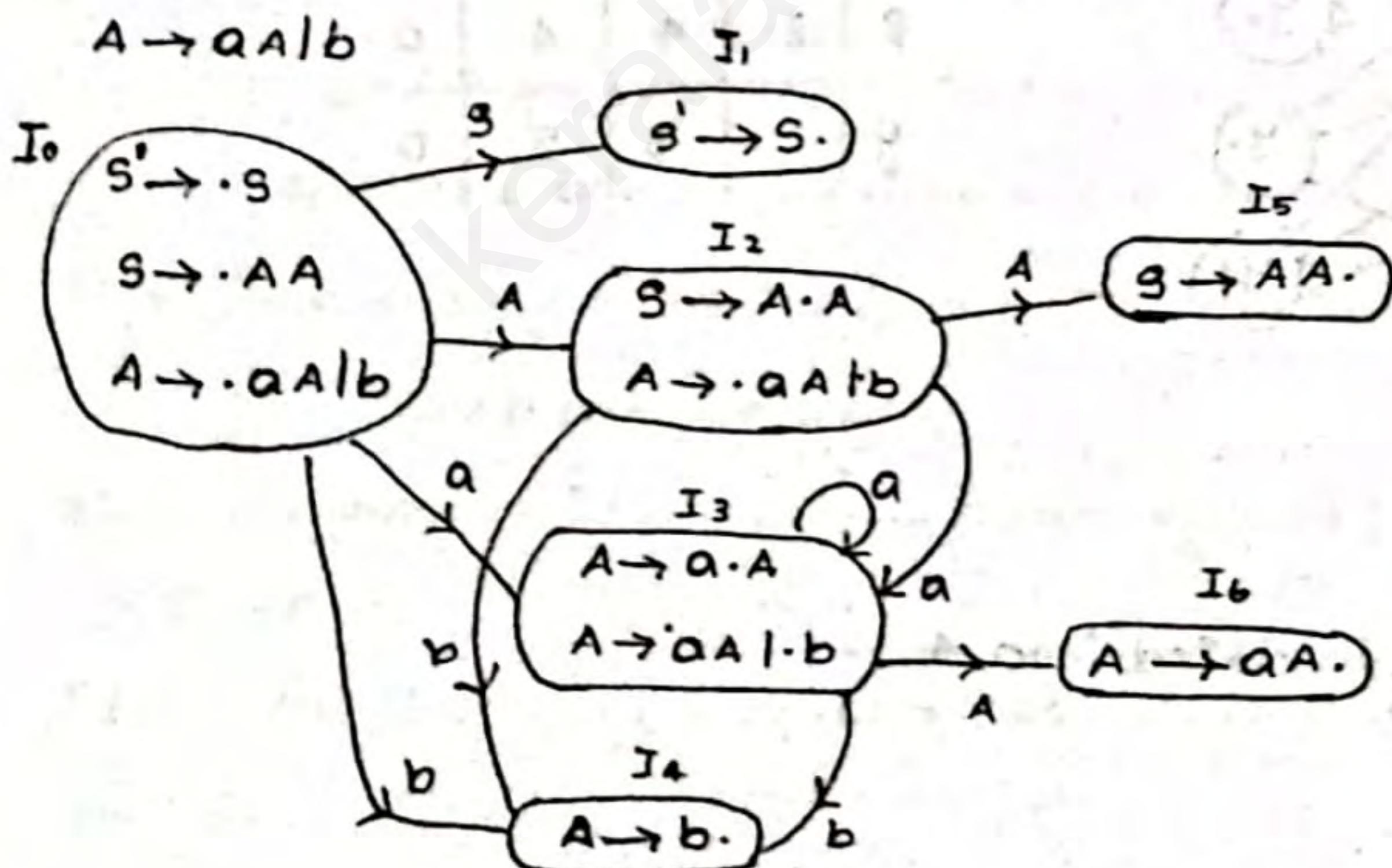
$$A \rightarrow aA/b$$

Augmented graphs

$$S' \rightarrow S$$

$$S \rightarrow A A$$

$$A \rightarrow aA/b$$



	Actions			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	$\gamma_3$	$\gamma_3$	$\gamma_3$		
5	$\gamma_1$	$\gamma_1$	$\gamma_1$		
6	$\gamma_2$	$\gamma_2$	$\gamma_2$		

$S \rightarrow AA \rightarrow ①$   
 $A \rightarrow aA \rightarrow ②$   
 $A \rightarrow b \rightarrow ⑤$

$\gamma$  - reduce  
 $s$  - shift

$$Q) E \rightarrow E + T / T$$

$$T \rightarrow T * F / F$$

$$F \rightarrow (E) / id$$

Bad Rule

$$S \rightarrow AA$$

$$A \rightarrow aA/b$$

→ SLR(1) Passes

$$S \rightarrow AA$$

$$A \rightarrow aA/b$$

Augmented grammar

$$S' \rightarrow S$$

$$S' \rightarrow S$$

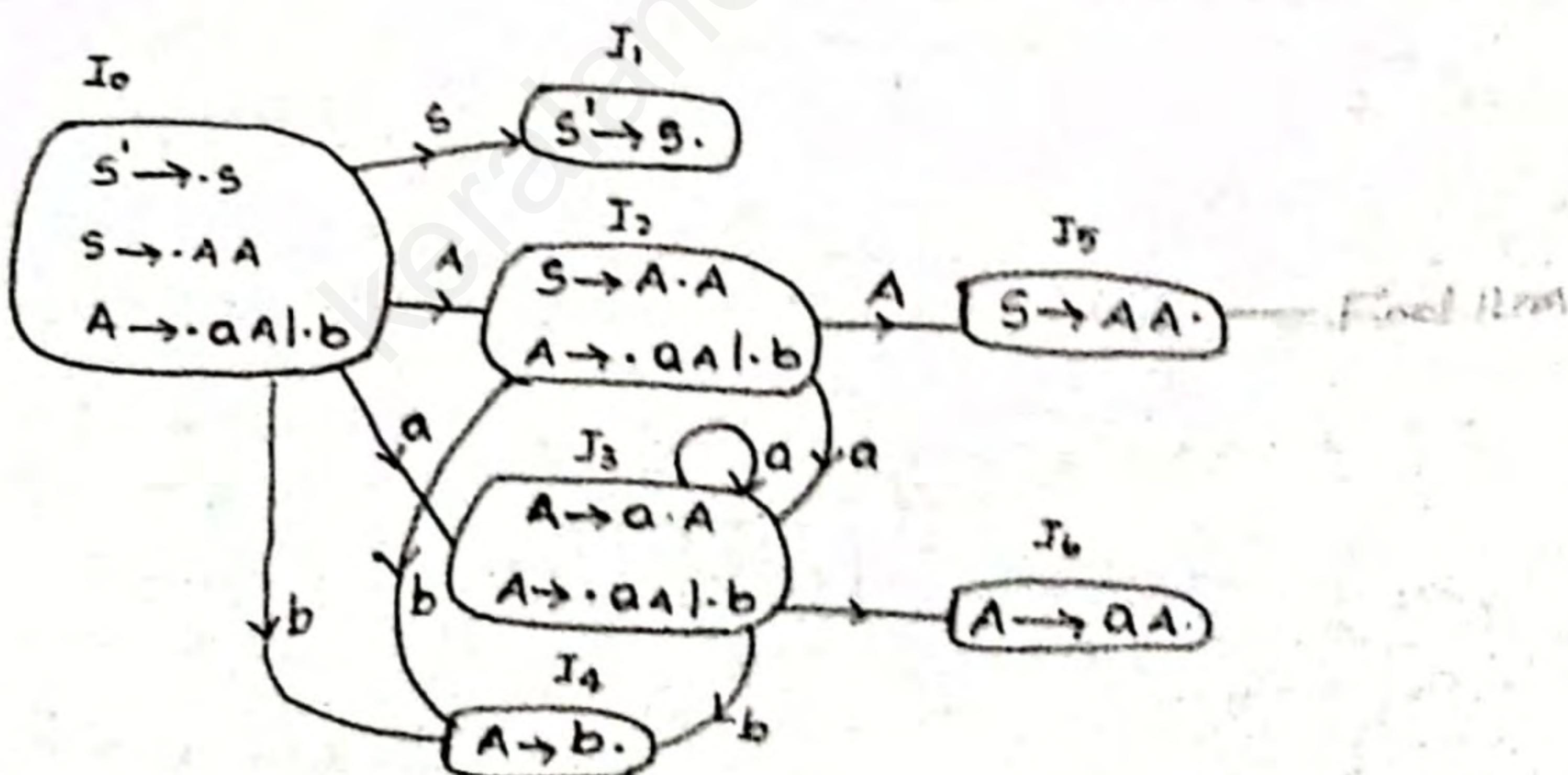
$$S \rightarrow \cdot AA$$

$$A \rightarrow \cdot aA/b$$

$$S \rightarrow A A$$

$$A \rightarrow a A/b$$

$$A \rightarrow \cdot a A/b$$



Action	Go to
a    b    \$	A    S
0 $s_3$ $s_4$	2    1
	Acpt
1	
2 $s_3$ $s_4$	5
3 $s_3$ $s_4$	6
4 $r_3$ $r_3$ $r_3$	8
5	
6 $r_2$ $r_2$ $r_2$	8

First      Follow

$$S \rightarrow \{a, b\} \quad \{a, b, \$\}$$

$$A \rightarrow \{a, b\}$$

String: aabb\$

0

aabbs

↑  
check 0 on a  $\rightarrow s_3$ 

0 a

aabbs

↑  
check 3 on a  $\rightarrow s_3$ 

0 a | 3

aabbs

↑  
check 3 on b  $\rightarrow s_4$ 

0 a | 3 | a | 3

aa bb\$

↑  
check 4 on b  $\rightarrow r_3$ 

0 a | 3 | a | 3

pop 2

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

reduce b

push A

0 a | 3 | a | 3 | b | 4 | \*

aabbs

0	a	3	a	3	A
---	---	---	---	---	---

check 3 on A → 6

push 6

0	a	3	a	3	A	6
---	---	---	---	---	---	---

check 6 on b → z<sub>2</sub>

reduce 2.

0	a	3	g	z	#	4
---	---	---	---	---	---	---

Pop A element

push A

0	a	3	A
---	---	---	---

check 3 on A → 6.

push 6

0	g	z	A	4
---	---	---	---	---

check 6 on b → z<sub>2</sub>.

POP 4, push A

0	A
---	---

check 0 on A

push 2

0	A	2
---	---	---

check 2 on b → z<sub>4</sub>

shift

aabb\$

↑  
check b on 4 → z<sub>3</sub>

reduce 5, pop 2

0	A	2	A	5
---	---	---	---	---

check 2 on A → 5

push 5

0	A	2	A	5
---	---	---	---	---

check 5 on \$ → z<sub>1</sub>

reduce 1

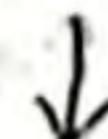
0	5
---	---

Pop A, push 5

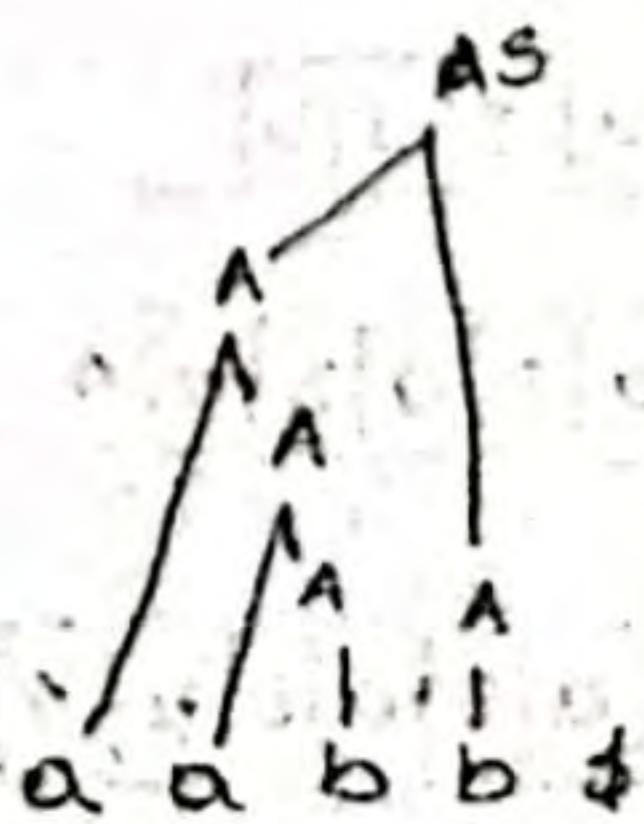
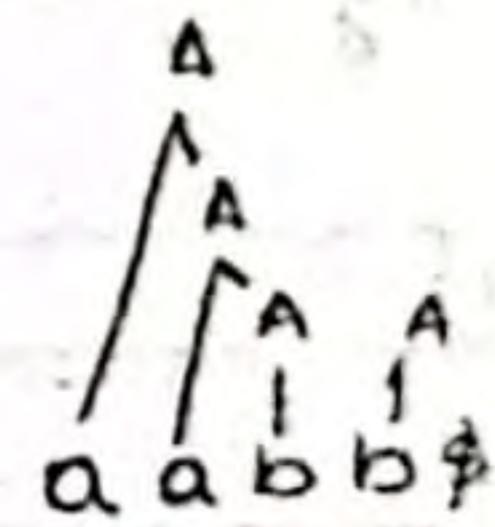
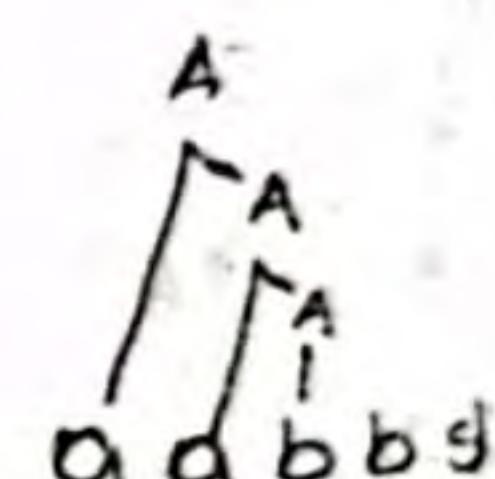
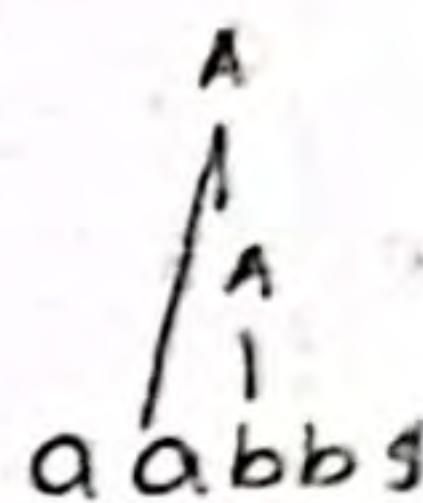
0	5
---	---

check 0 on 5 → 1

check 1 on \$



accept



LALR(1) CLR(1)



LR(1) items

= LR(0) item + lookahead

$A \rightarrow \alpha \cdot BB, a/b, B \rightarrow \gamma$

closure(B) B  $\rightarrow \cdot \gamma, \text{first}(B)$

$\rightarrow \cdot \gamma, c/d$

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

$B \rightarrow \cdot \gamma, \text{first}(a/b)$

$\rightarrow \cdot \gamma, a/b$

### LALR(1)

$S \rightarrow AA$

$A \rightarrow aA/b$

Augmented grammar:  $s' \rightarrow s$ .  
Augmented graph:  $s' \rightarrow s$ .

lookahead  $s' \rightarrow \cdot s, \$$

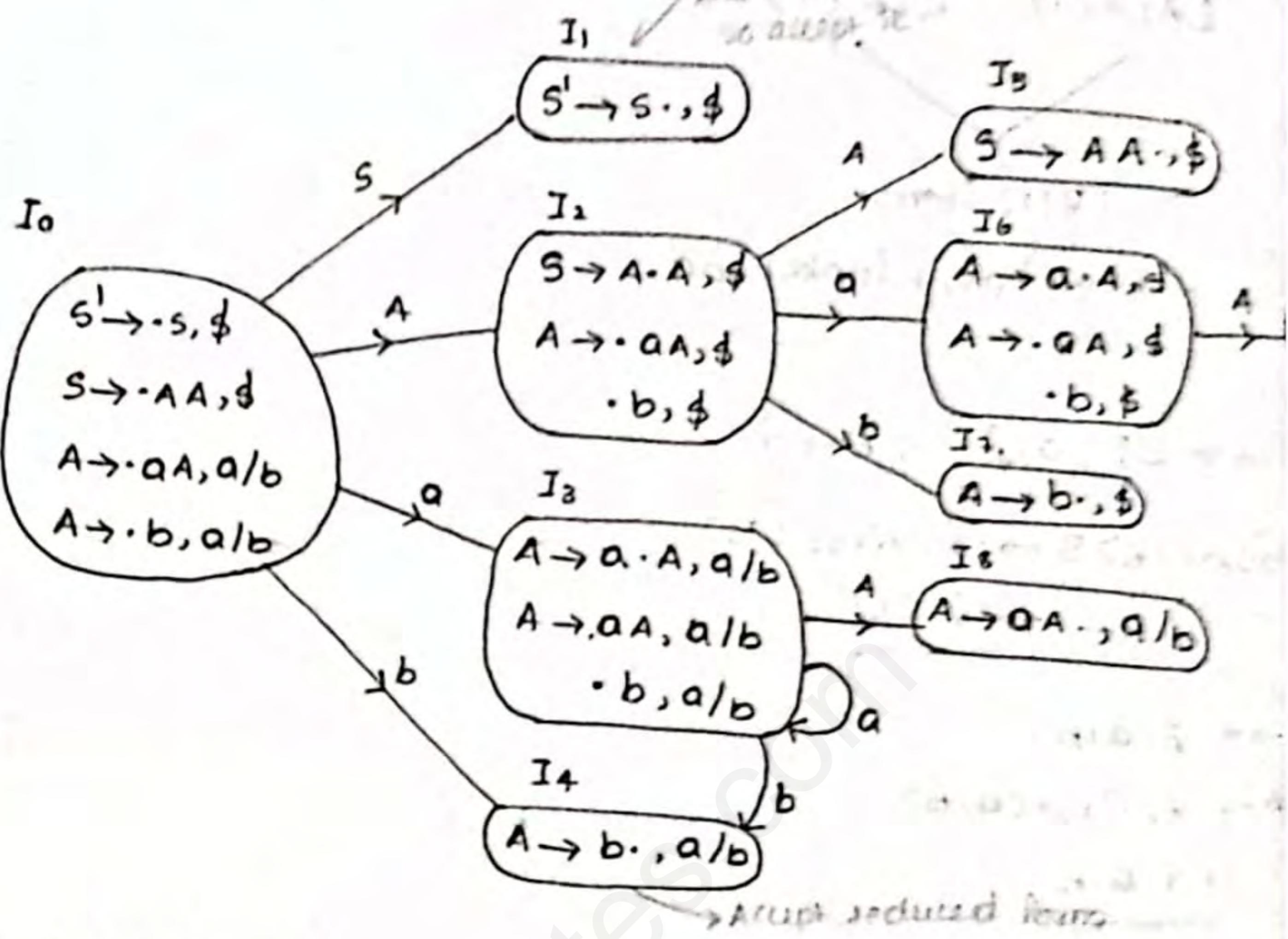
$s \rightarrow \cdot s, \$$

$s \rightarrow \cdot AA, \$$  closure of S

$A \rightarrow \cdot aA, a/b$  closure of A

$A \rightarrow \cdot b, a/b$

Lookahead symbol is  
always +



Passing table

	Action	goto
	a    b    \$	A    S
0	S <sub>3</sub> S <sub>4</sub>	2   1
1		Acpt
2	S <sub>6</sub> S <sub>7</sub>	5
3	S <sub>3</sub> S <sub>4</sub>	8
4	r <sub>3</sub> r <sub>3</sub>	
5		r <sub>1</sub>
6	S <sub>6</sub> S <sub>7</sub>	9
7		r <sub>3</sub>
8	r <sub>2</sub> r <sub>2</sub>	r <sub>1</sub>
9		r <sub>1</sub>

CLR (D).

	Actions			goto
	a	b	\$	A    S
0	$s_{36}$	$s_{47}$		2
1			acpt	
2	$s_{36}$	$s_{47}$		5
36	$s_{36}$	$s_{47}$		89
47	$\gamma_3$	$\gamma_3$	$\gamma_3$	
5			$\gamma_1$	
89	$\gamma_2$	$\gamma_2$	$\gamma_2$	

aabbs

LALR(D)00|a|3|0|a|3|a|3|0|a|3|a|3|s|1|A|

aabbs

↑

check 0 on a  $\rightarrow s_3$ 

aabbs

↑

check 3 on a  $\rightarrow s_3$ .

aabbs

↑

check 3 on b  $\rightarrow s_4$ .check 4 on b  $\rightarrow \gamma_3$ 

reduce

Pop 2

Push A

 $S \rightarrow AA - ①$  $A \rightarrow aA - ②$  $A \rightarrow b - ③$

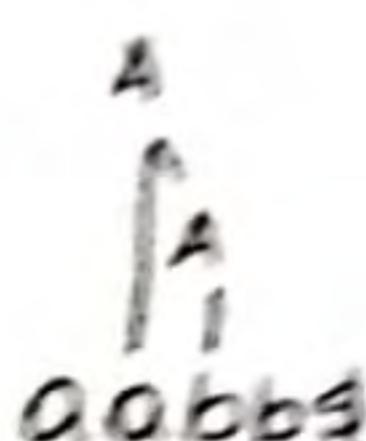
0 0 3 # E F A 6

check 3 on A  $\rightarrow \tau_3$   
check 3 on B  $\rightarrow \tau_2$

POP 4 elements (3, A, 3, 0).

0 0 3 | A

Push A.



aabbs

0 0 3 | A | 3

check 3 on A  $\rightarrow \tau_3$

Push 3.

0 0 3 | A | 3

3 on A  $\rightarrow \tau_2$

is A  $\rightarrow \tau_1$ .

POP 4 elements.

Push A.



0 4 | 3 | 4 | 8 | A

check 4 on B  $\rightarrow \tau_2$

0 | A | 2 | B | A

aabbs



0 | A | 2 | E |

check 2 on E

0 | 5 | 1

$\rightarrow$  Output.

CLR.

0	a	36
---	---	----

0	a	36	a
---	---	----	---

0	a	36	a	36
---	---	----	---	----

0	a	36	a	36	b	47
---	---	----	---	----	---	----

0	a	36	a	36	b	47
---	---	----	---	----	---	----

0	a	36	a	36	A
---	---	----	---	----	---

0	a	36	a	36	A	89
---	---	----	---	----	---	----

0	a	36	A
---	---	----	---

0	A	36	A	89
---	---	----	---	----

0	A
---	---

0	A	2
---	---	---

0	A	2	47	b
---	---	---	----	---

0	A	2	47	b	A
---	---	---	----	---	---

0	A	2	A	5
---	---	---	---	---

0	A	2	A	5	\$	5
---	---	---	---	---	----	---

0	s
---	---

aabb\$



36 on a

aabbs



aabbs

47 on b  $\gamma_3 \rightarrow$ 

reduce

A.

aabbs

36 on A - 89

89 on -  $\gamma_2$ .

POP 4

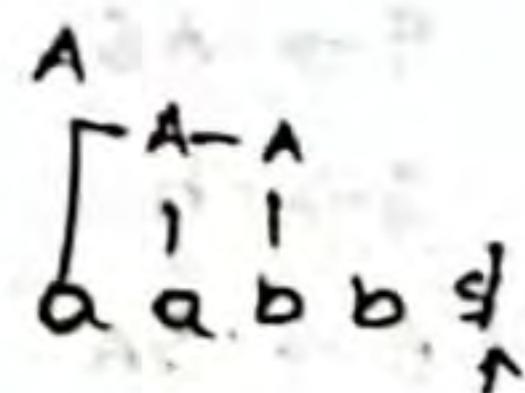
push A

aabbs.

89 on b.  $\gamma_2$ 

POP 4

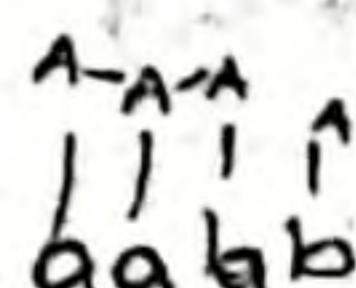
push A



2 on A - 5

5 on \$ -  $\gamma_1$ 

POP 4.



accept

