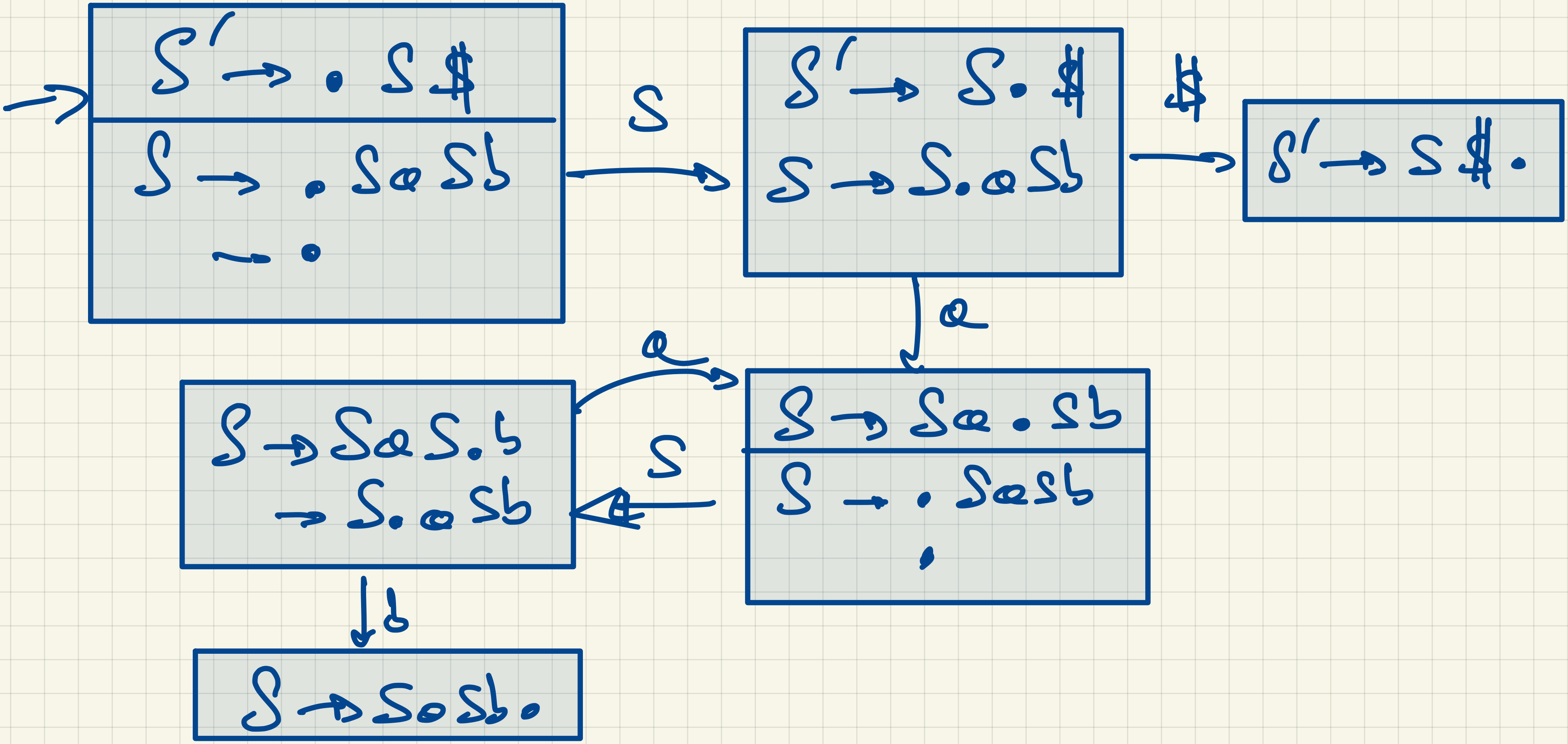
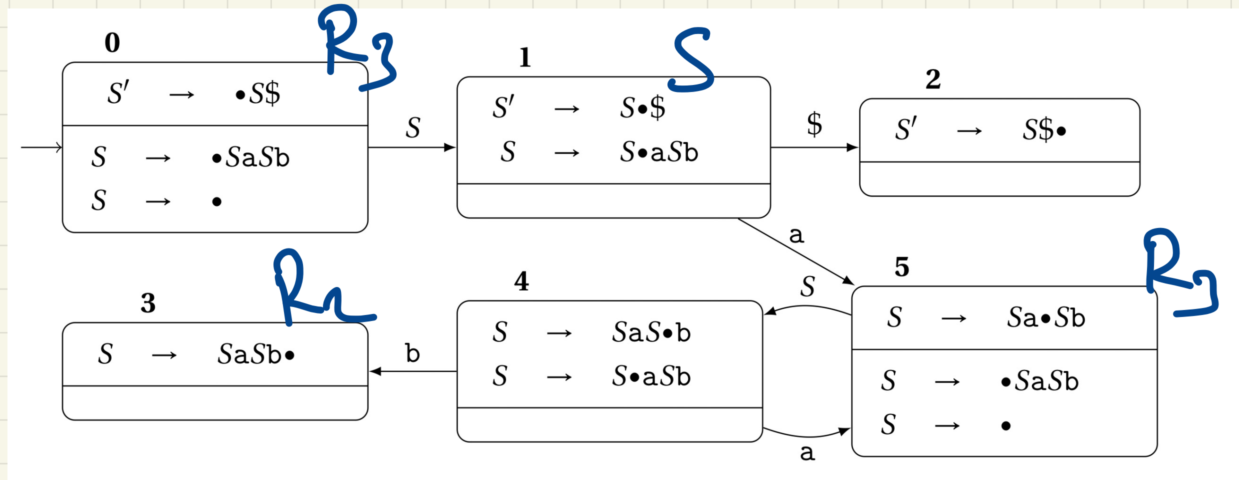
November, 14th

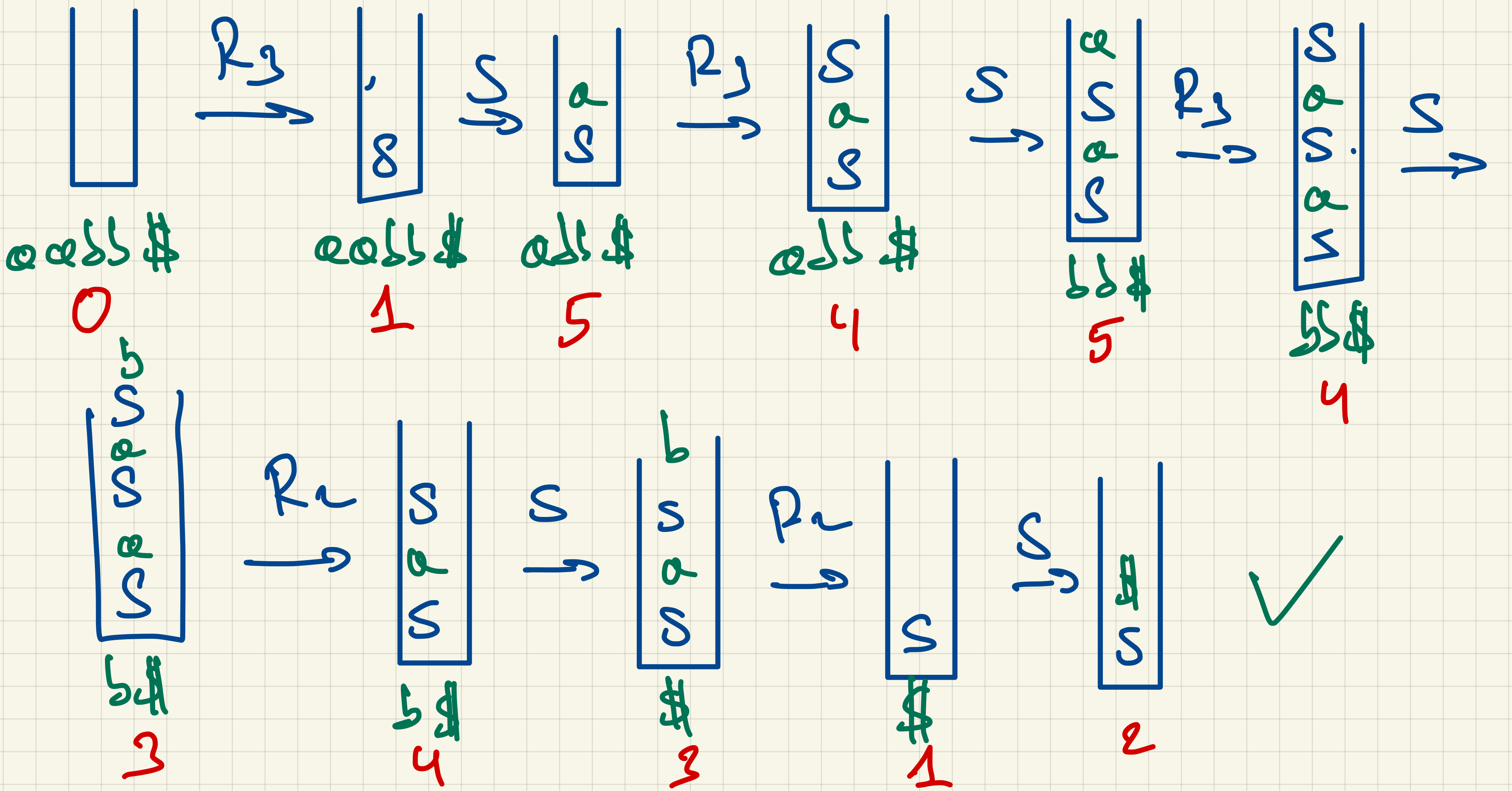
(1)	S'	$\rightarrow$	$S\$$
(2)	S	$\rightarrow$	$S\mathtt{a}S\mathtt{b}$
(3)		$\rightarrow$	ε



(1)	S'	$\to$	$S\$$
(2)	S	$\to$	$SaSb$
(3)		$\to$	ε

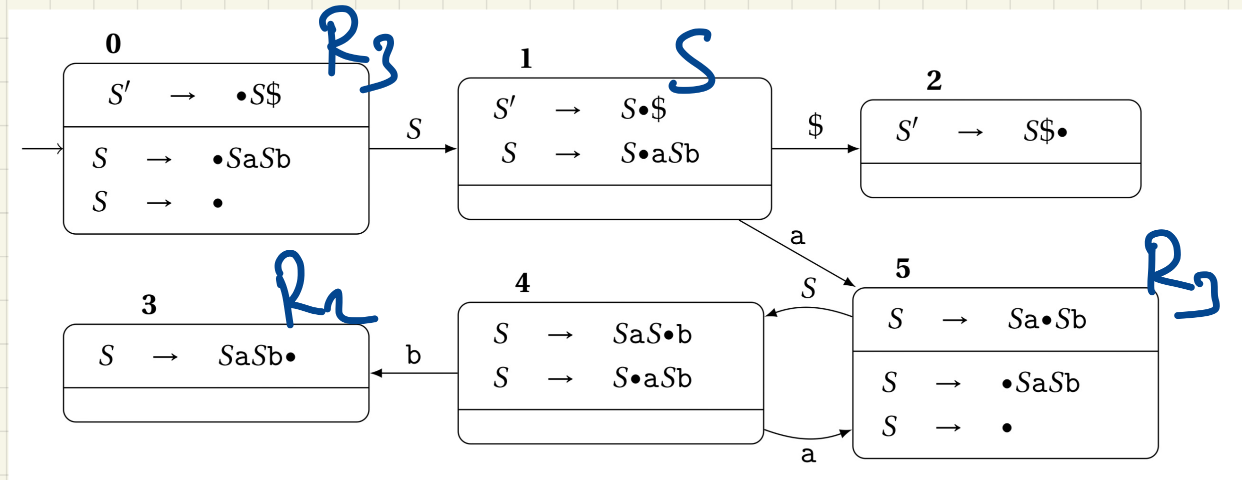
6.5: An LR(0) grammar whose s





Action Table

State	Action
0	R_3
1	S
2	Accept
3	R_2
4	S
5	R_3

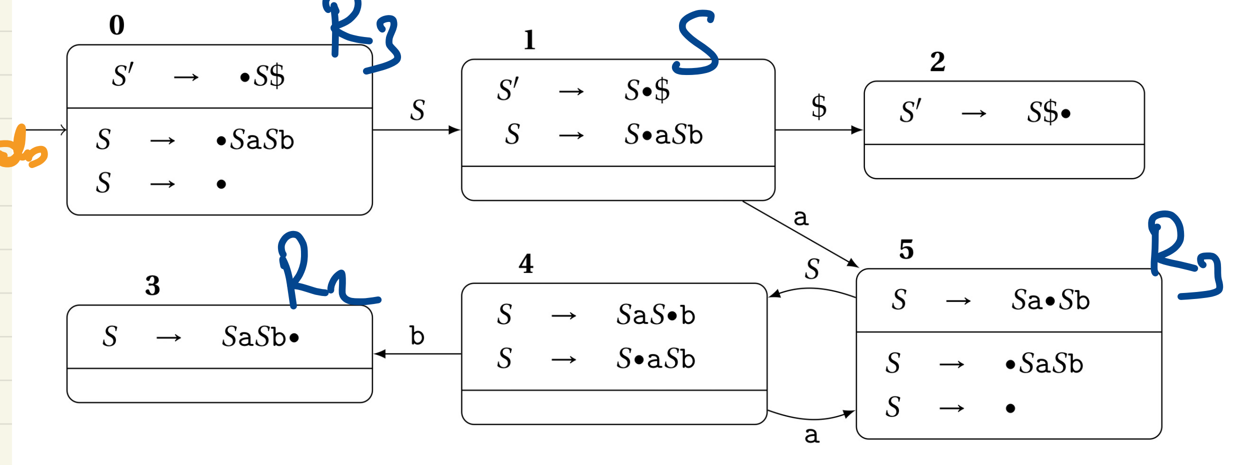


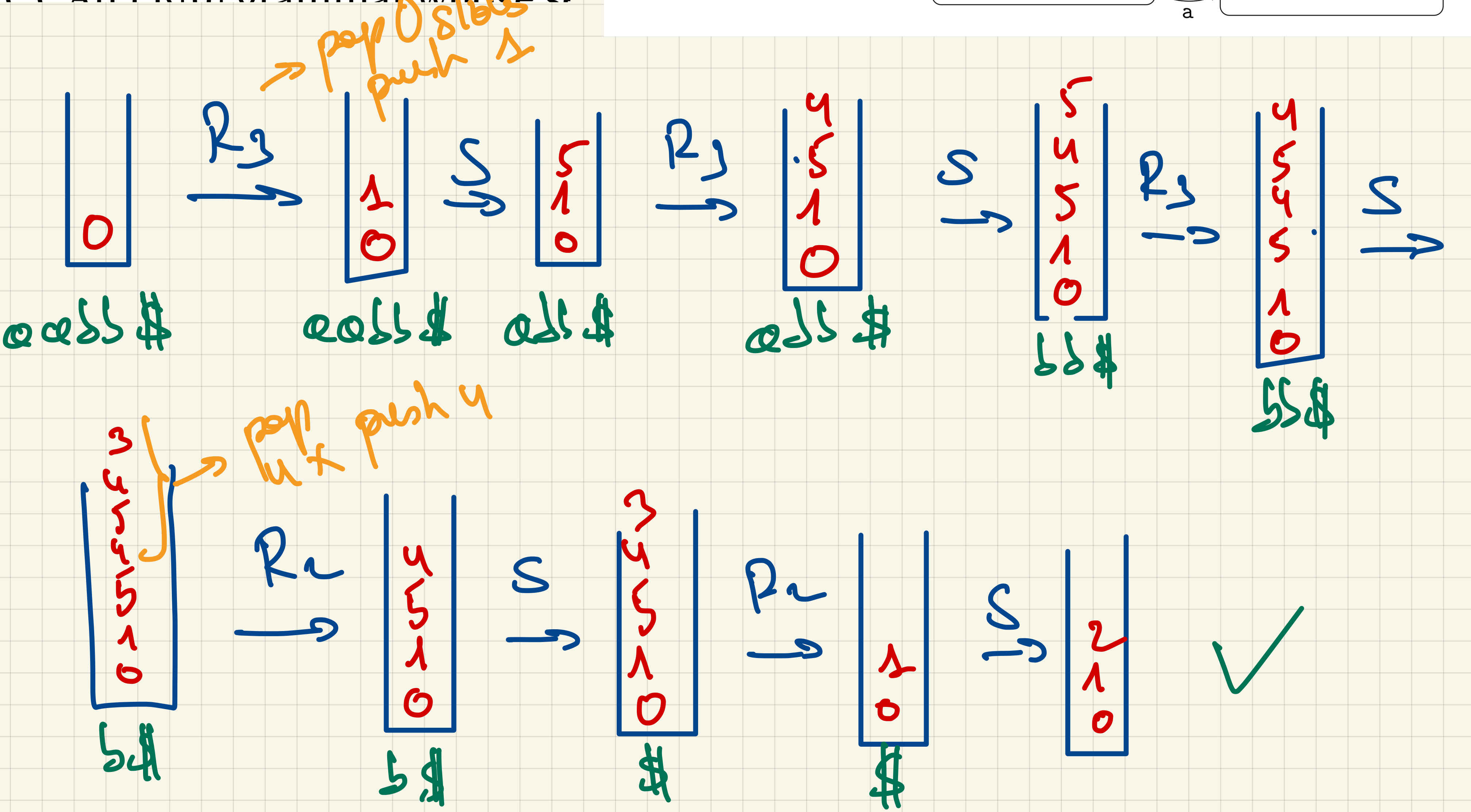
We can remember only the state on the stack!

(1)	S'	$\rightarrow$	$S\$$
(2)	S	$\rightarrow$	$\boxed{SaSb}$
(3)		$\rightarrow$	ε

4 Symbols

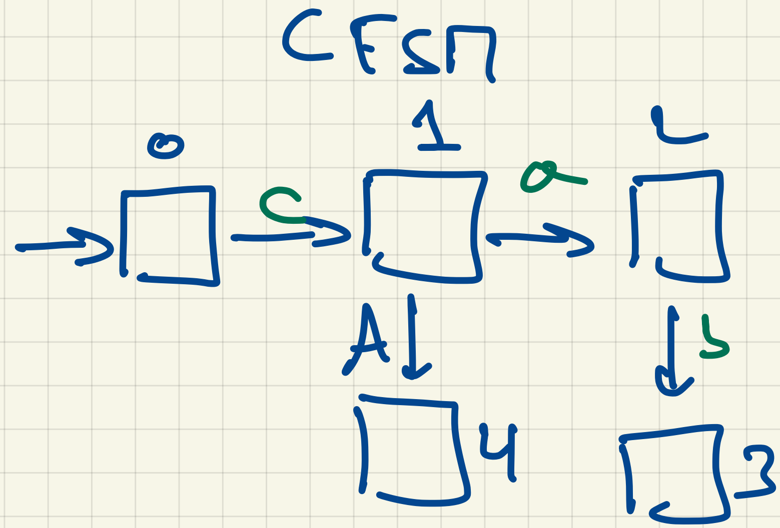
6.5: An LR(0) grammar whose s



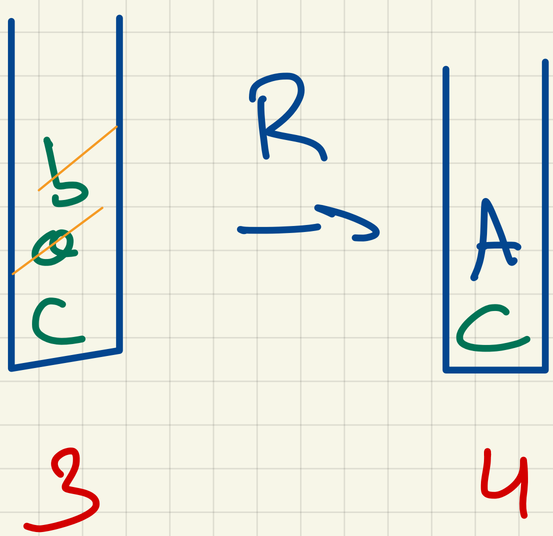


Reduce

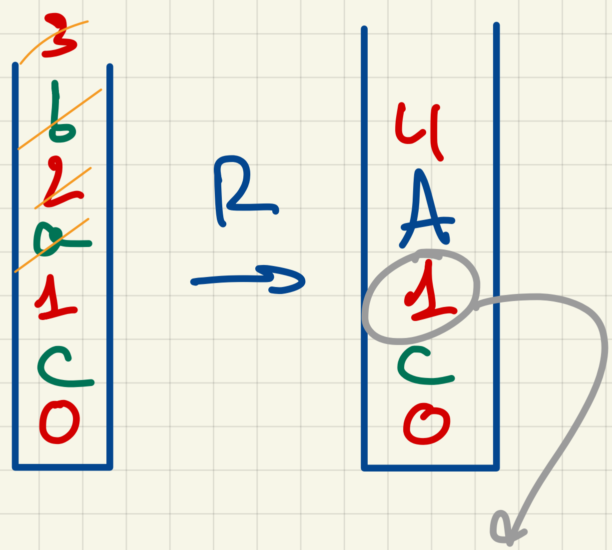
$A \to ab$



V_1



V_2



State reached in the CFSM after reading c

Adding look-ahead...

(1)	S	$\to$	Exp\$
(2)	Exp	$\to$	Exp + Prod
(3)		$\to$	Prod
(4)	Prod	$\to$	Prod $*$ Atom
(5)		$\to$	Atom
(6)	Atom	$\to$	Id
(7)		$\to$	(Exp)

What about this guy?

$S \to \,.\, \text{Exp}\$$

$\text{Exp} \to \,.\, \text{Exp+Prod}$
$\qquad\quad\;\; .\, \text{Prod}$
$\text{Prod} \to \,.\, \text{Prod*Atom}$
$\qquad\quad\;\; .\, \text{Atom}$
$\text{Atom} \to \,.\, \text{Id}$
$\qquad\quad\;\; .\, \text{(Exp)}$

Prod

$\text{Exp} \to \text{Prod}\,.$ $\quad R_3$
$\text{Prod} \to \text{Prod}\,.\,{*}\text{Atom}$ $\quad S$

↳ Shift/Reduce conflict!!

(1)	S	$\rightarrow$	Exp\$
(2)	Exp	$\rightarrow$	Exp + Prod
(3)		$\rightarrow$	Prod
(4)	Prod	$\rightarrow$	Prod $*$ Atom
(5)		$\rightarrow$	Atom
(6)	Atom	$\rightarrow$	Id
(7)		$\rightarrow$	(Exp)

$*$ is the only symbol to shift to keep viable prefixes

State	\$	+	*	Id	(	)
2	R_3	R_3	S			R_3

2

$$\boxed{\begin{array}{l} Exp \rightarrow Prod\,\bullet \\ Prod \rightarrow Prod\,\bullet * Atom \end{array}}$$

R_3

S

$\downarrow *$

Follow (Exp) = {\$, +,)}

We will use the Follow(Exp) to decide when to Reduce.

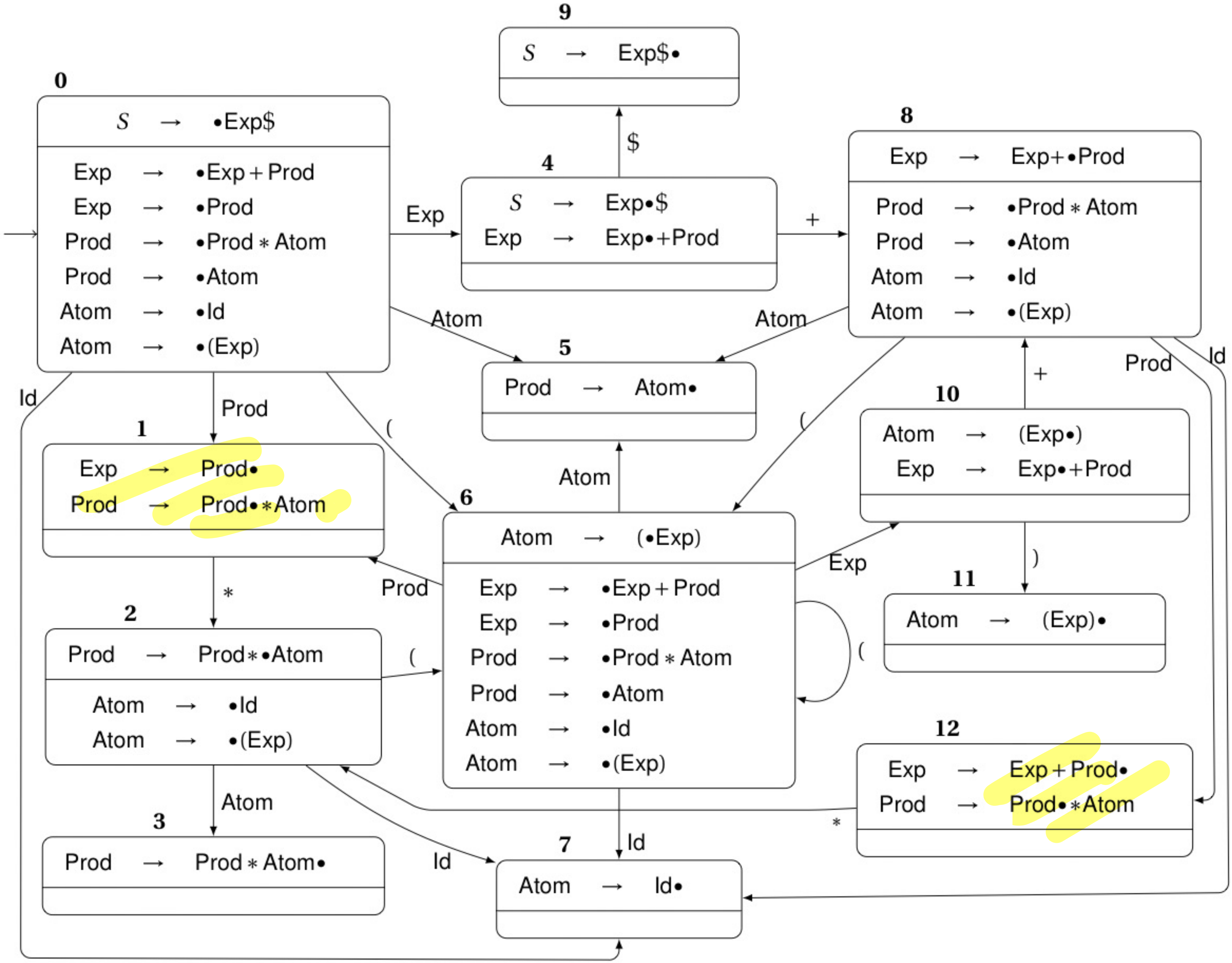


Figure 6.8: The CFSM for the grammar generating expressions.

M	+	*	Id	(	)	$	ε
0			S	S			
1	3	S			3	3	
2			S	S			
3	4	4			4	4	
4	S					S	
5	5	5			5	5	
6			S	S			
7	6	6			6	6	
8			S	S			
9							A
10	S				S		
11	7	7			7	7	
12	2	S			2	2	

left scanning
right parsing

S LR(1)

Simple.

look-ahead.

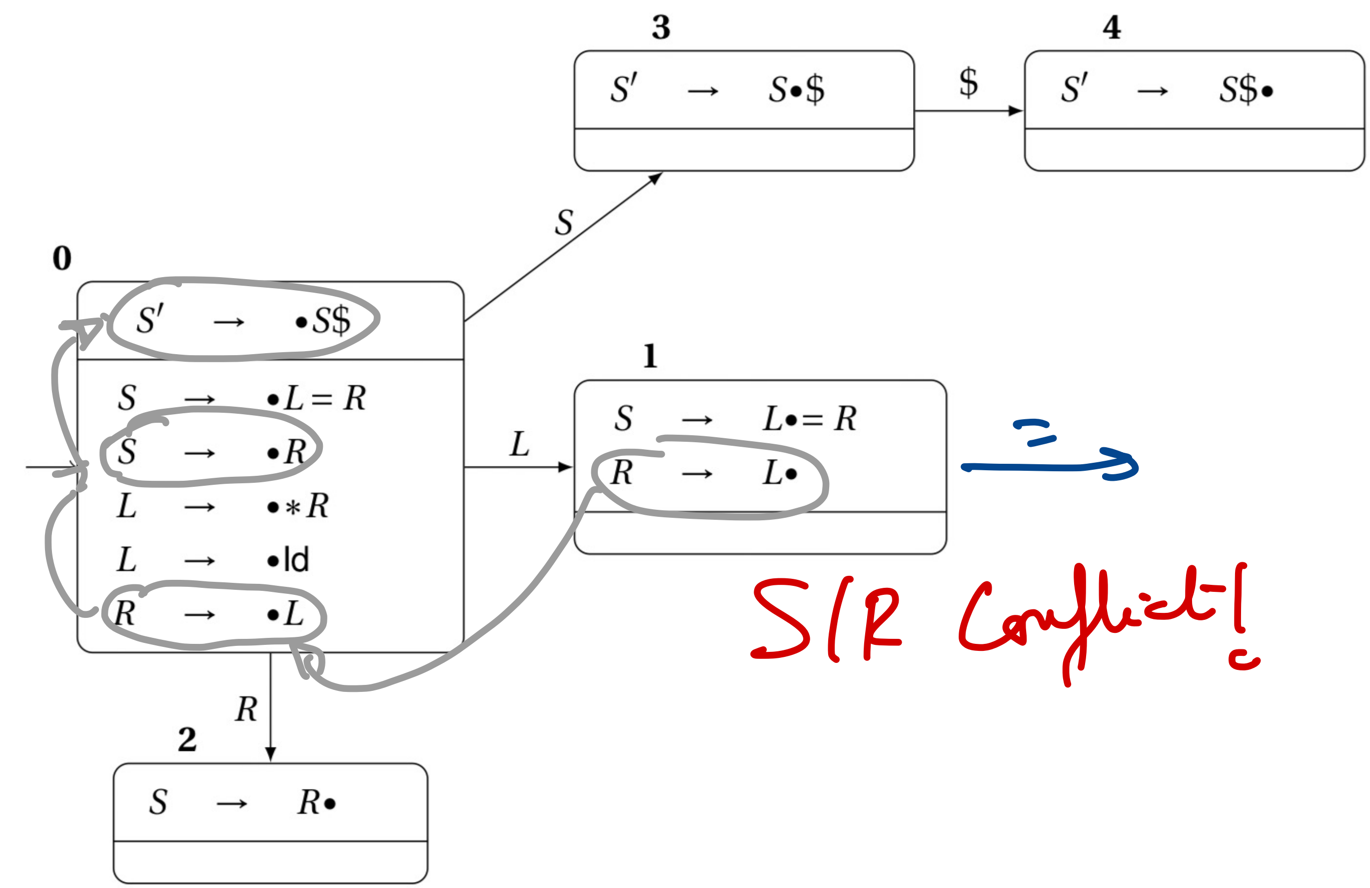
Beyond SLR(1)

(1)	S'	$\longrightarrow$	$S\$$
(2)	S	$\longrightarrow$	$L = R$
(3)	S	$\longrightarrow$	R
(4)	L	$\longrightarrow$	$*R$
(5)	L	$\longrightarrow$	Id
(6)	R	$\longrightarrow$	L

	\$	=	*	Id
1		S/R6		

Follow(R) = {\$, =}

"global Follow"



S/R Conflict!

Problem: the SLR(1) technique is too coarse in this case.

We will use information from the CFSM to obtain a finer notion of Follow

"local Follow"

"context".

LR(k) CFSM

We extend the notion of item by adding potential contexts

$\subseteq$ Follow(A)
but more precise

$$A \to \alpha_1 \cdot \alpha_2 , \underbrace{\{ \beta_1 , \beta_2 , \ldots \beta_n \}}$$

what I expect on the input when reducing $\alpha_1 \alpha_2$ as A.

(1)	S'	$\rightarrow$	$S\$$
(2)	S	$\rightarrow$	$L = R$
(3)	S	$\rightarrow$	R
(4)	L	$\rightarrow$	$*R$
(5)	L	$\rightarrow$	Id
(6)	R	$\rightarrow$	L

	=	\$
1	S	R_6

$S' \rightarrow \bullet S\$$	ε
$S \rightarrow \bullet L = R$	$\{\$\}$
$S \rightarrow \bullet R$	$\{\$\}$
$L \rightarrow \bullet * R$	$\{=, \$\}$
$L \rightarrow \bullet \mathrm{Id}$	$\{=, \$\}$
$R \rightarrow \bullet L$	$\{\$\}$

$\xrightarrow{L}$

1

$S \rightarrow L \bullet = R$	$\{\$\}$
$R \rightarrow L \bullet$	$\{\$\}$

we add the item with L but in a different context!!

3

Closure

$$A \to \alpha_1 \bullet B \alpha_2 \, x$$

$$B \to \bullet \beta_1$$

$$B \to \bullet \beta_2$$

$$B \to \beta_1$$

$$B \to \beta_2$$

$$\mathrm{First}(\alpha_2 x)$$

LALR(1)

Technique used to "compact" the LR(1) automaton.

Idea: we merge states that have the same "heart"

$A \to \alpha_1 \cdot \alpha_2 \; x_1$

$B \to \beta_1 \cdot \beta_2 \; x_2$

$A \to \alpha_1 \cdot \alpha_2 \; y_1$

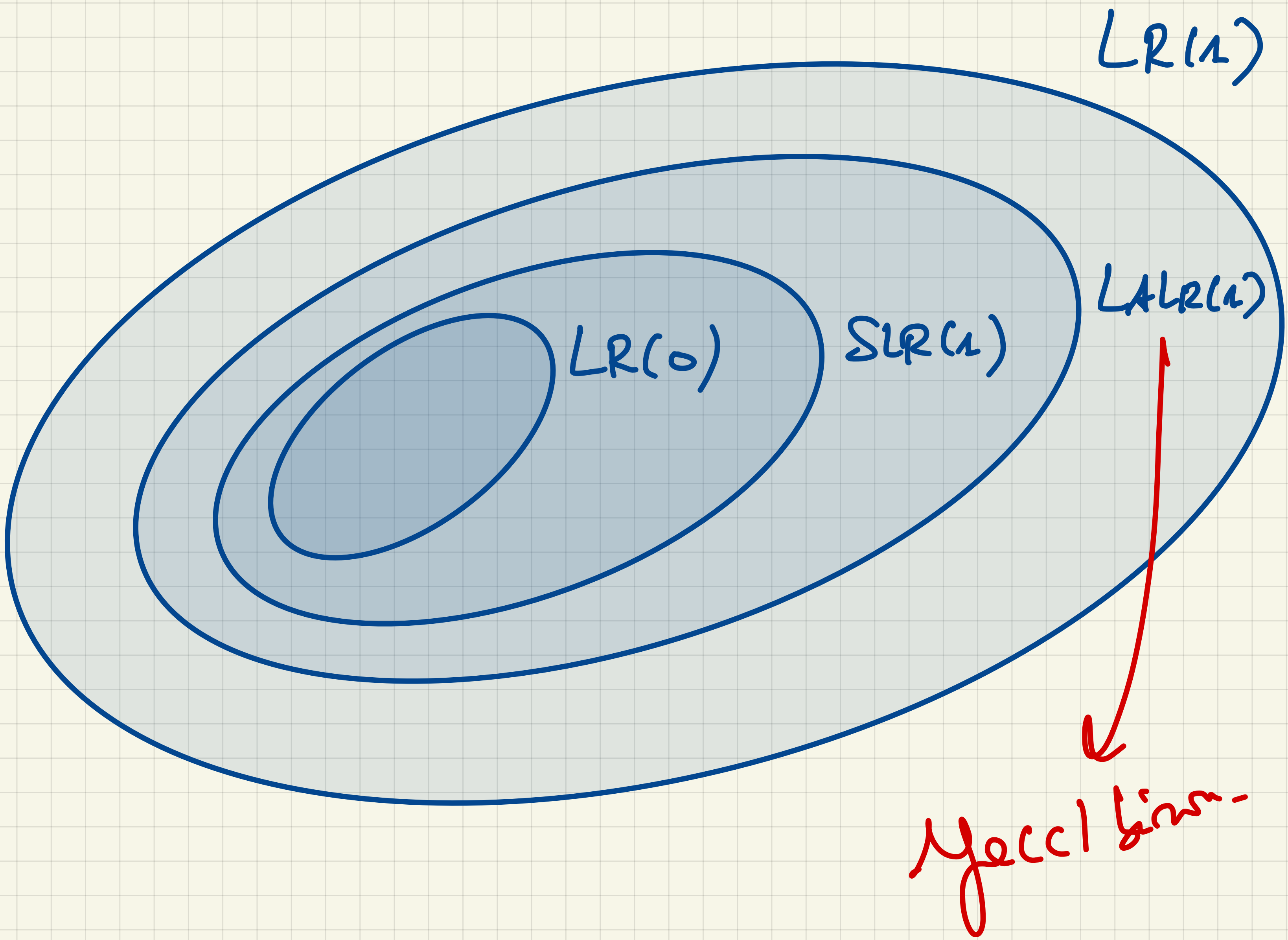
$B \to \beta_1 \cdot \beta_2 \; y_2$

$\Longrightarrow$

$A \to \alpha_1 \cdot \alpha_2 \; \{x_1, y_1\}$

$B \to \beta_1 \cdot \beta_2 \; \{x_2, y_2\}$

Might re-introduce conflicts!!


LR(1)
LALR(1)
SLR(1)
LR(0)
Yacc / bison

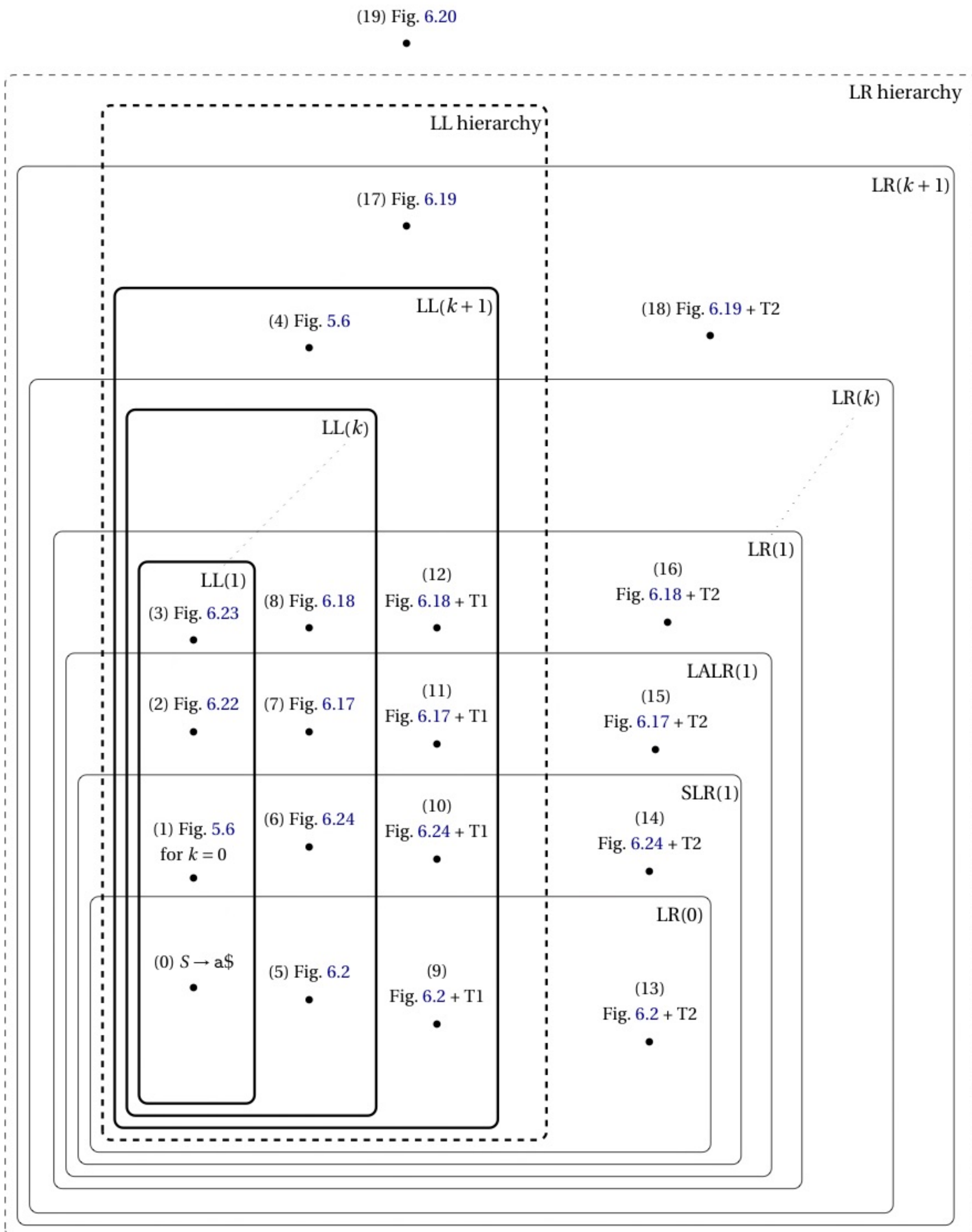


Figure 6.21: Comparison of different (syn-