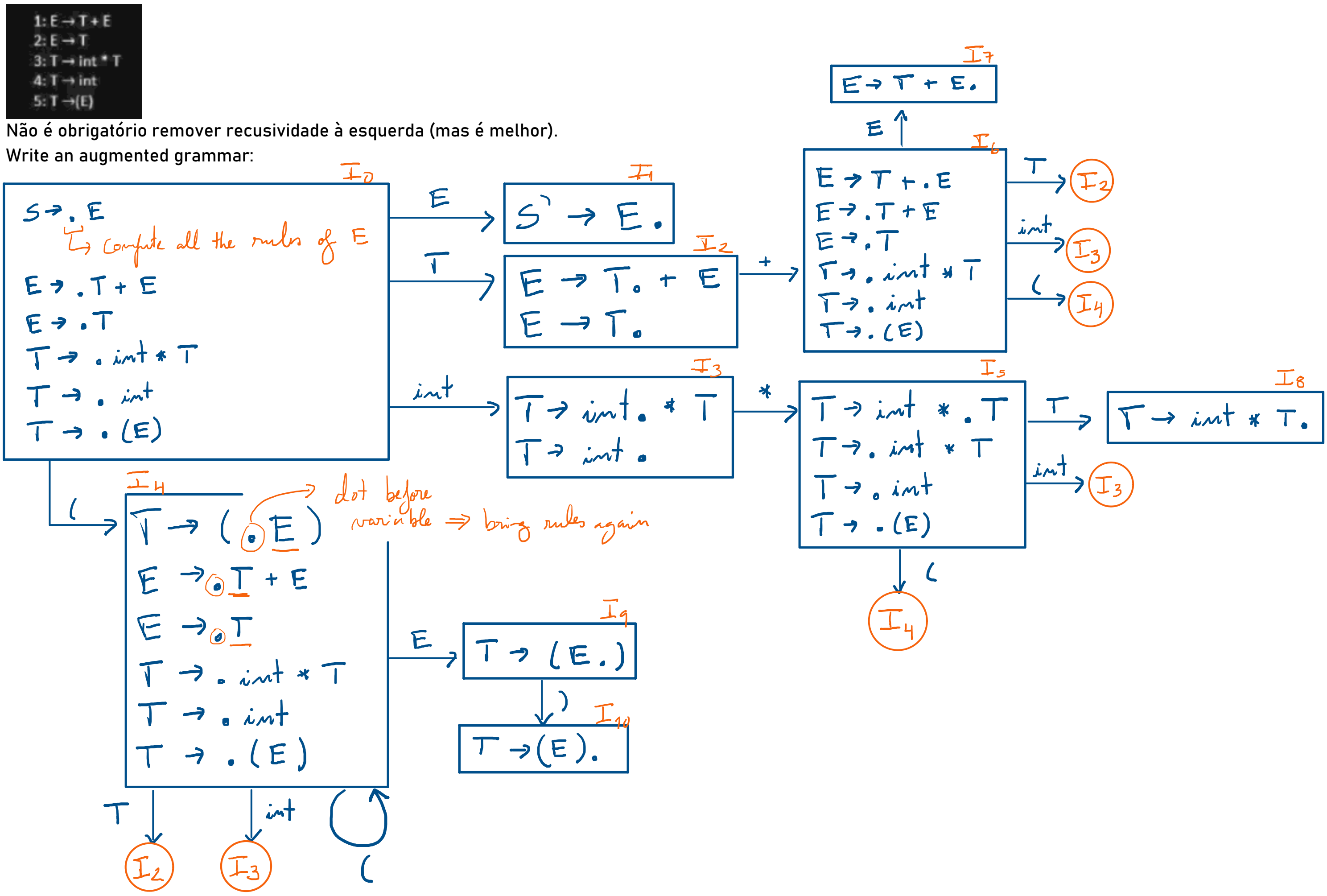
LR(0) & SLR(1) Automaton & Parsing Table

3 de junho de 2024 11:05

1: E → T + E
2: E → T
3: T → int * T
4: T → int
5: T → (E)

Não é obrigatório remover recusividade à esquerda (mas é melhor).

Write an augmented grammar:



SLR(1) Table

-	int	(	)	+	*	$	E	T
0	S3	S4					1	2
1						AC		
2			R2	S6		R2		
3			R4	R4	S5	R4		
4	S3	S4					9	2
5	S3	S4						8
6	S3	S4					7	2
7			R1			R1		
8			R3	R3		R3		
9			S10					
10			R5	R5		R5		

Follow(T) = {+,), $}

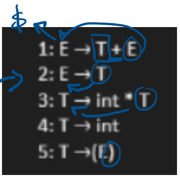
E → T.

= Reducing to 2

Follow(T) pois "int" é terminal

}+{ U Follow(E) U Follow(T)

$



1: E → T + E
2: E → T
3: T → int * T
4: T → int
5: T → (E)

⇒ {+} U {$,)}

S -> Shift

R -> Reduce -- used in accepting states ⇒ shift cannot be used ⇒ use the follow
ex.: Follow(E) = { $,) }

AC: (accept) -- stops the parsing

LR(0) Table -- the reduce is in all the terminal symbols (lame)

-	int	(	)	+	*	$	E	T
0	S3	S4					1	2
1						AC		
2	R2	R2	R2	R2/S6	R2	R2		
3	R4	R4	R4	R4	R4/S5	R4		
4	S3	S4					9	2
5	S3	S4						8
6	S3	S4					7	2
7	R1	R1	R1	R1	R1	R1		

Shift/Reduce Conflict!

8	R3	R3	R3	R3	R3	R3		
9			S10					
10	R5	R5	R5	R5	R5	R5		