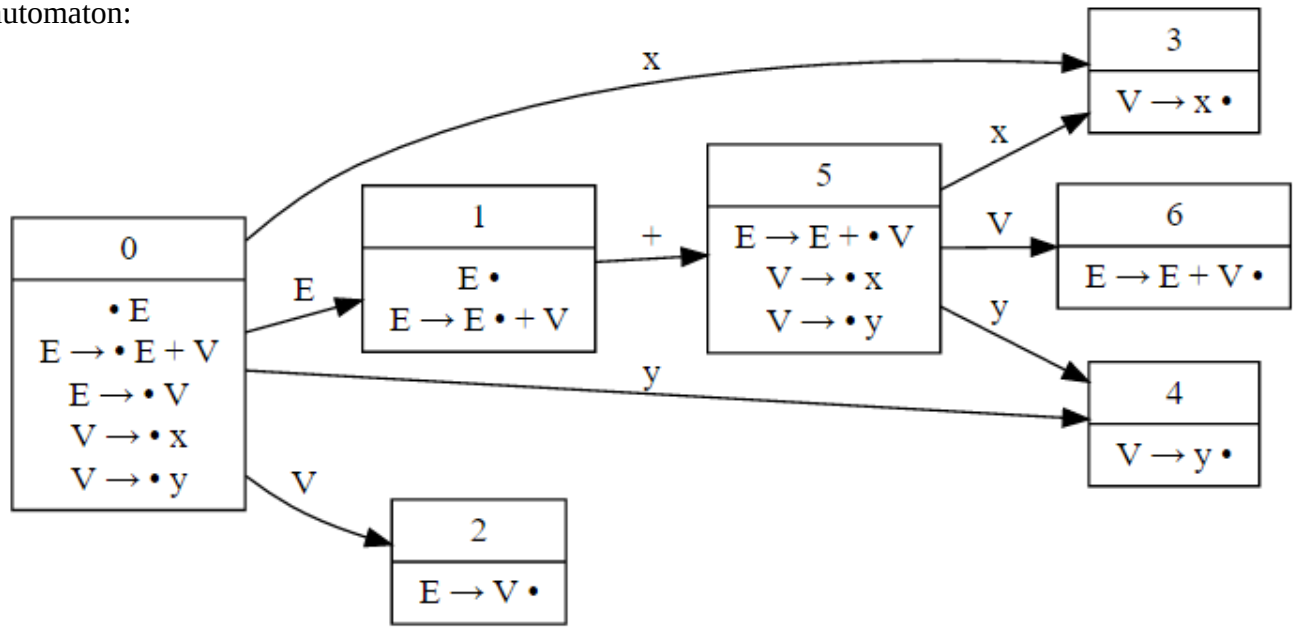
```
S' → E
E  → E + V
   | V
V  → x | y
```

```
FOLLOW(E) = { +, $ }
FOLLOW(V) = { +, $ }
```

LR(0) item sets and automaton:



SLR tables:

ACTION GOTO

State	+	x	y	$	*E*	*V*
0		shift(3)	shift(4)		1	2
1	shift(5)			accept		
2	reduce($E \rightarrow V$)			reduce($E \rightarrow V$)		
3	reduce($V \rightarrow$ x)			reduce($V \rightarrow$ x)		
4	reduce($V \rightarrow$ y)			reduce($V \rightarrow$ y)		
5		shift(3)	shift(4)			6
6	reduce($E \rightarrow E + V$)			reduce($E \rightarrow E + V$)		

"x"

STACK	LA	ACTION
> 0	x	shift 3
> 0 3	$	reduce **V** → x (pop 1)
> 0		goto 2
> 0 2	$	reduce **E** → V (pop 1)
> 0		goto 1
> 0 1	$	accept

"x+y"

STACK	LA	ACTION
> 0	x	shift 3
> 0 3	+	reduce **V** → x (pop 1)
> 0		goto 2
> 0 2	+	reduce **E** → V (pop 1)
> 0		goto 1
> 0 1	+	shift 5
> 0 1 5	y	shift 4
> 0 1 5 4	$	reduce **V** → y (pop 1)
> 0 1 5		goto 6
> 0 1 5 6	$	reduce **E** → E + V (pop 3)
> 0		goto 1
> 0 1	$	accept

"x+y+y"

STACK	LA	ACTION
> 0	x	shift 3
> 0 3	+	reduce **V** → x (pop 1)
> 0		goto 2
> 0 2	+	reduce **E** → V (pop 1)
> 0		goto 1
> 0 1	+	shift 5
> 0 1 5	y	shift 4
> 0 1 5 4	+	reduce **V** → y (pop 1)
> 0 1 5		goto 6
> 0 1 5 6	+	reduce **E** → E + V (pop 3)
> 0		goto 1
> 0 1	+	shift 5
> 0 1 5	y	shift 4
> 0 1 5 4	$	reduce **V** → y (pop 1)
> 0 1 5		goto 6
> 0 1 5 6	$	reduce **E** → E + V (pop 3)
> 0		goto 1
> 0 1	$	accept