

Compiladores

(IF688)

Leopoldo Teixeira
lmt@cin.ufpe.br | @leopoldomt

predictive parsers

- Simples, compactos, e eficientes
- A cada estágio do processo de *parsing*, sabem o conjunto de palavras que podem ocorrer como o próximo símbolo em uma entrada válida
- Por isto, costumam produzir mensagens de erro precisas e úteis

predictive parsers

- Maioria das construções de linguagens de programação podem ser expressas por LL(1)
- A principal desvantagem é o fato de não poder lidar com recursão à esquerda, o que dificulta modelar regras que associam à esquerda de maneira natural

Bottom-up parsing

top-down vs. bottom-up

- Técnicas de parsing vistas até agora usam análise *top-down*, ou descendente
- Técnicas *bottom-up*, ou ascendentes, consistem na construção da árvore de baixo pra cima, a partir das folhas

bottom-up

- mais complicada de implementar
- mais geral, impõe menos restrições à gramática
- recursão à esquerda e prefixos comuns não necessariamente constituem problemas
- ideia: converter (reduzir) o programa de entrada para o símbolo inicial

intuição

- o parser lê tokens até que tenha uma subpalavra w que case com o lado direito de alguma produção $A \rightarrow \beta$
- ao chegar neste estágio, substitui β por A **se isto resultar** em uma derivação válida
- processo de substituição é chamado de **redução**

Visualizando *bottom-up*

$S \rightarrow E\$$

$E \rightarrow T$

$E \rightarrow E + T$

$T \rightarrow \text{int}$

$T \rightarrow (E)$

int	+	(int	+	int	+	int)	\$
-----	---	---	-----	---	-----	---	-----	---	----

$S \rightarrow E\$$

$E \rightarrow T$

$E \rightarrow E + T$

$T \rightarrow \text{int}$

$T \rightarrow (E)$

int	+	(int	+	int	+	int)	\$
-----	---	---	-----	---	-----	---	-----	---	----

$S \rightarrow E\$$

$E \rightarrow T$

$E \rightarrow E + T$

$T \rightarrow \text{int}$

$T \rightarrow (E)$

T



int	+	(int	+	int	+	int)	\$
-----	---	---	-----	---	-----	---	-----	---	----

$S \rightarrow E\$$

$E \rightarrow T$

$E \rightarrow E + T$

$T \rightarrow \text{int}$

$T \rightarrow (E)$

E



T



int	+	(int	+	int	+	int)	\$
-----	---	---	-----	---	-----	---	-----	---	----

$S \rightarrow E\$$

$E \rightarrow T$

$E \rightarrow E + T$

$T \rightarrow \text{int}$

$T \rightarrow (E)$

E



T



int	+	(int	+	int	+	int)	\$
-----	---	---	-----	---	-----	---	-----	---	----

$S \rightarrow E\$$

$E \rightarrow T$

$E \rightarrow E + T$

$T \rightarrow \text{int}$

$T \rightarrow (E)$

E



T



int

+

(

T



int

+

int

+

int

)

\$

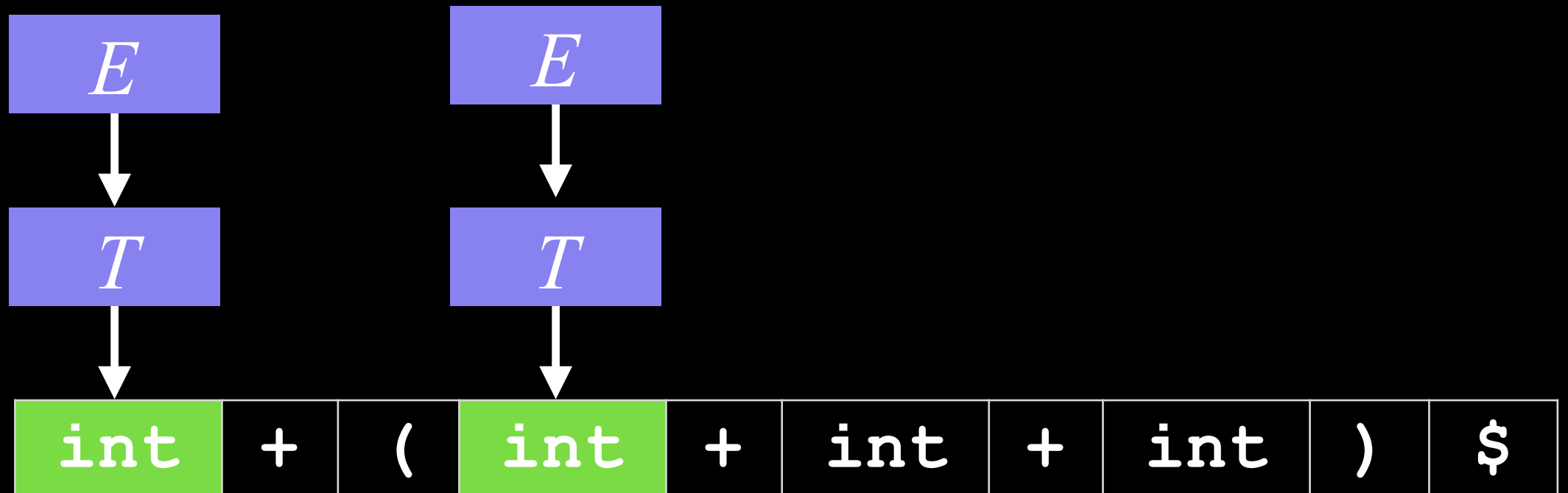
$S \rightarrow E\$$

$E \rightarrow T$

$E \rightarrow E + T$

$T \rightarrow \text{int}$

$T \rightarrow (E)$



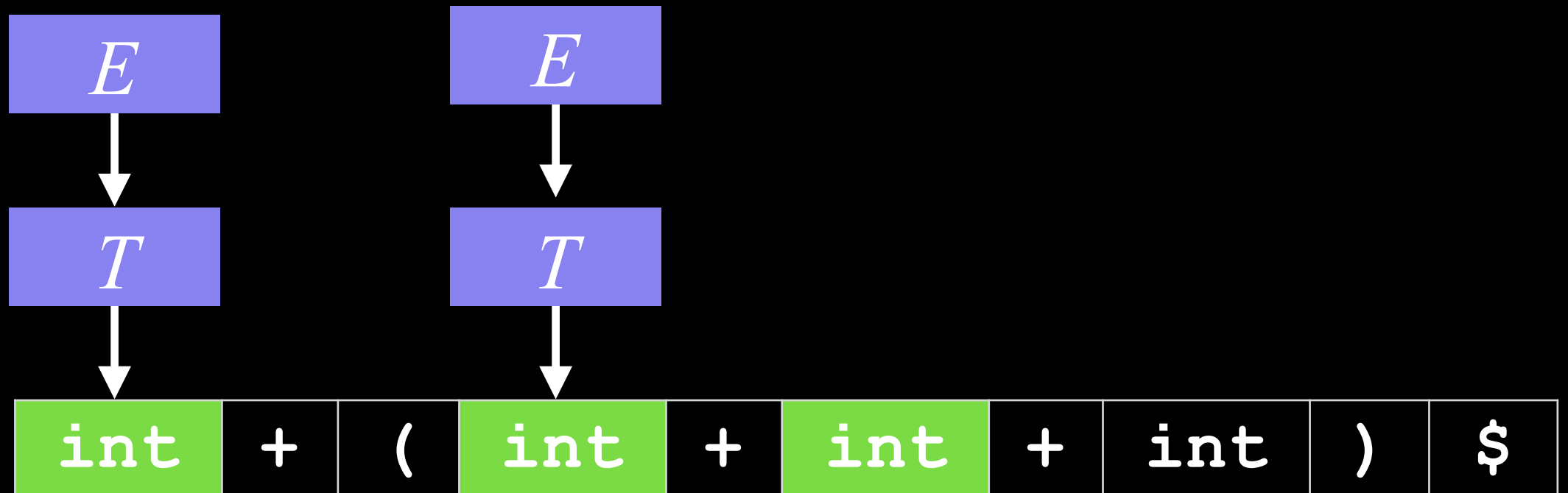
$S \rightarrow E\$$

$E \rightarrow T$

$E \rightarrow E + T$

$T \rightarrow \text{int}$

$T \rightarrow (E)$



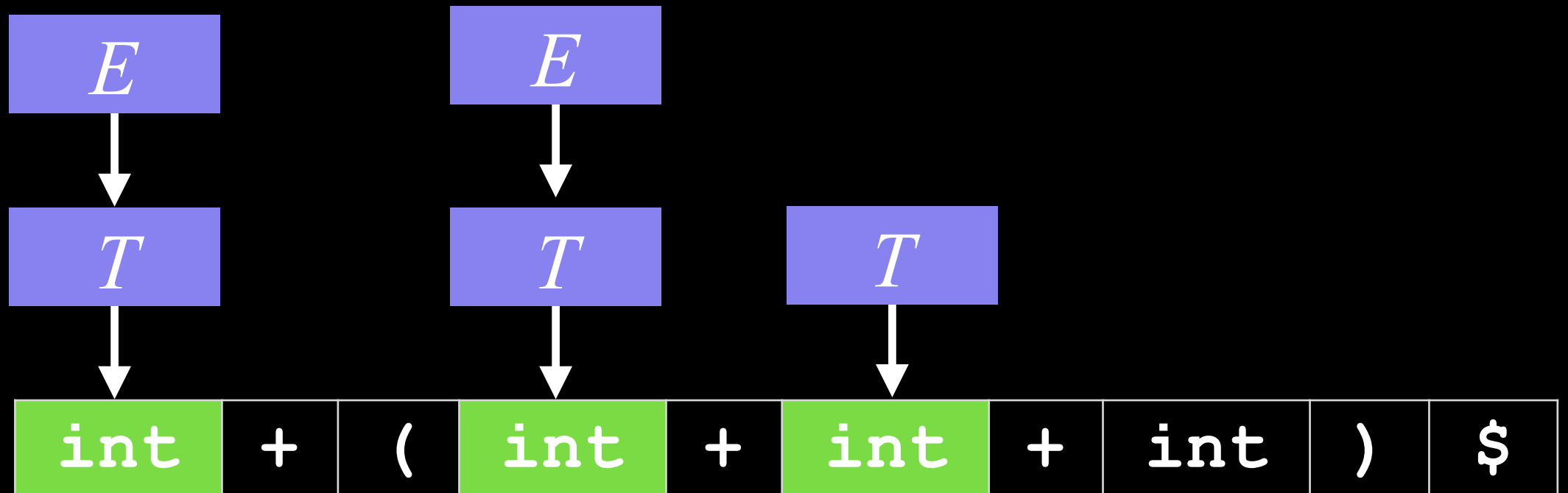
$S \rightarrow E\$$

$E \rightarrow T$

$E \rightarrow E + T$

$T \rightarrow \text{int}$

$T \rightarrow (E)$



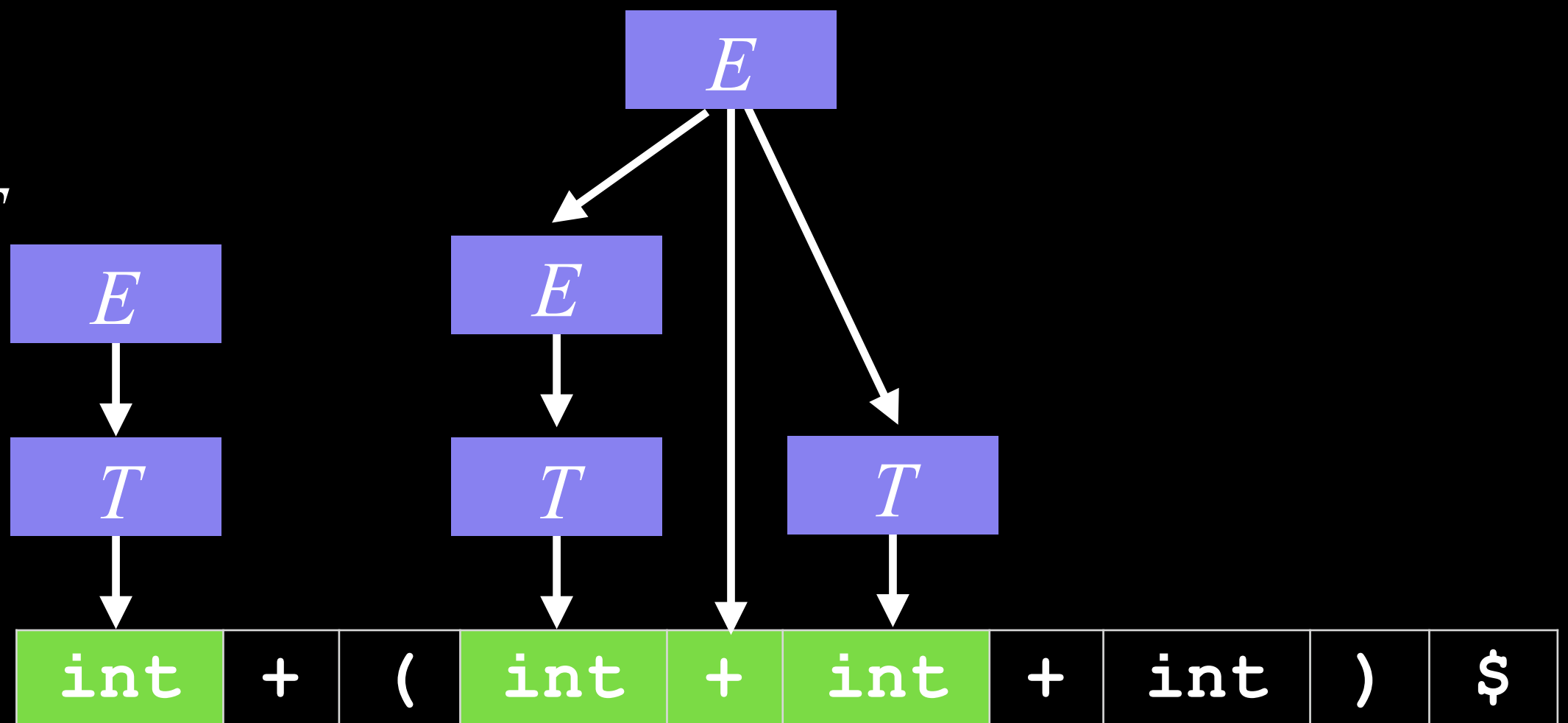
$S \rightarrow E\$$

$E \rightarrow T$

$E \rightarrow E + T$

$T \rightarrow \text{int}$

$T \rightarrow (E)$



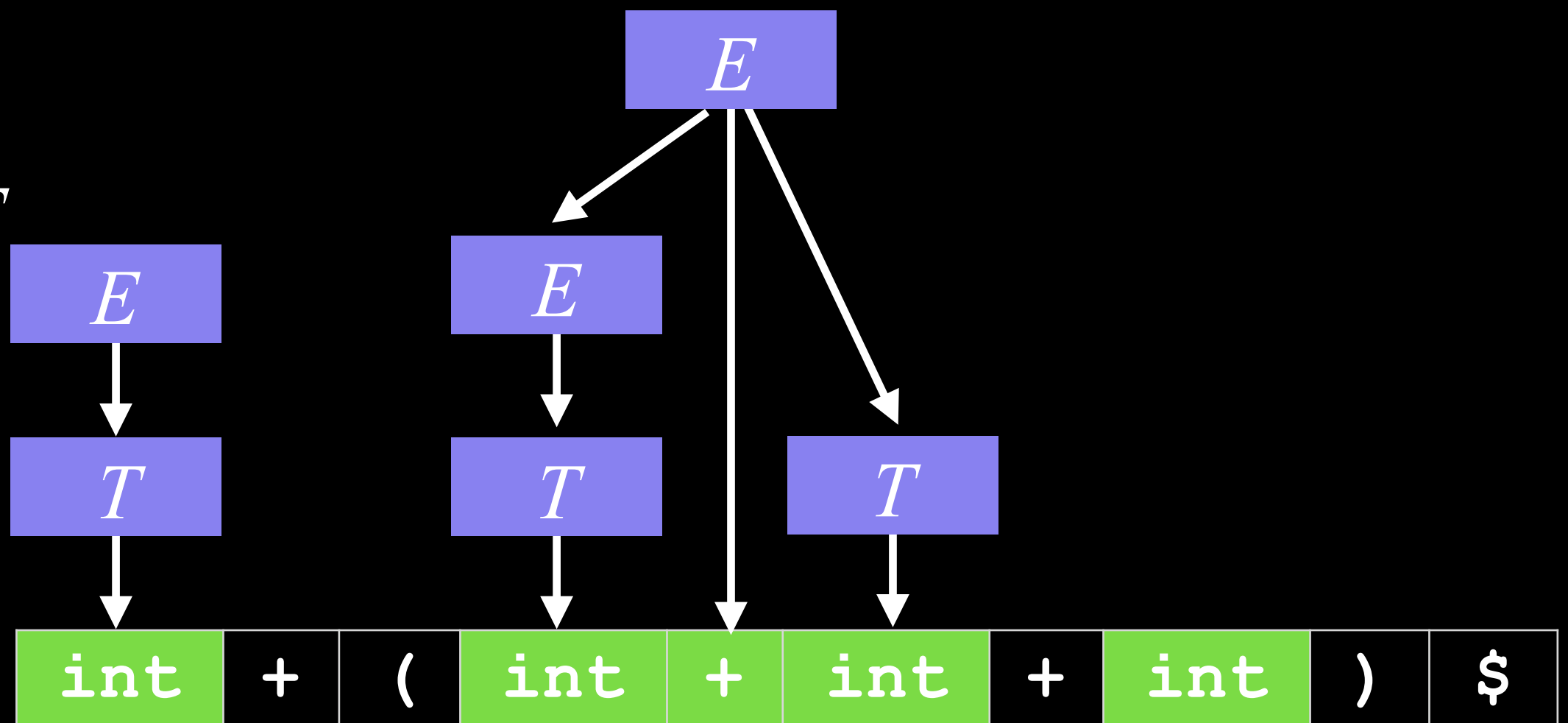
$S \rightarrow E\$$

$E \rightarrow T$

$E \rightarrow E + T$

$T \rightarrow \text{int}$

$T \rightarrow (E)$



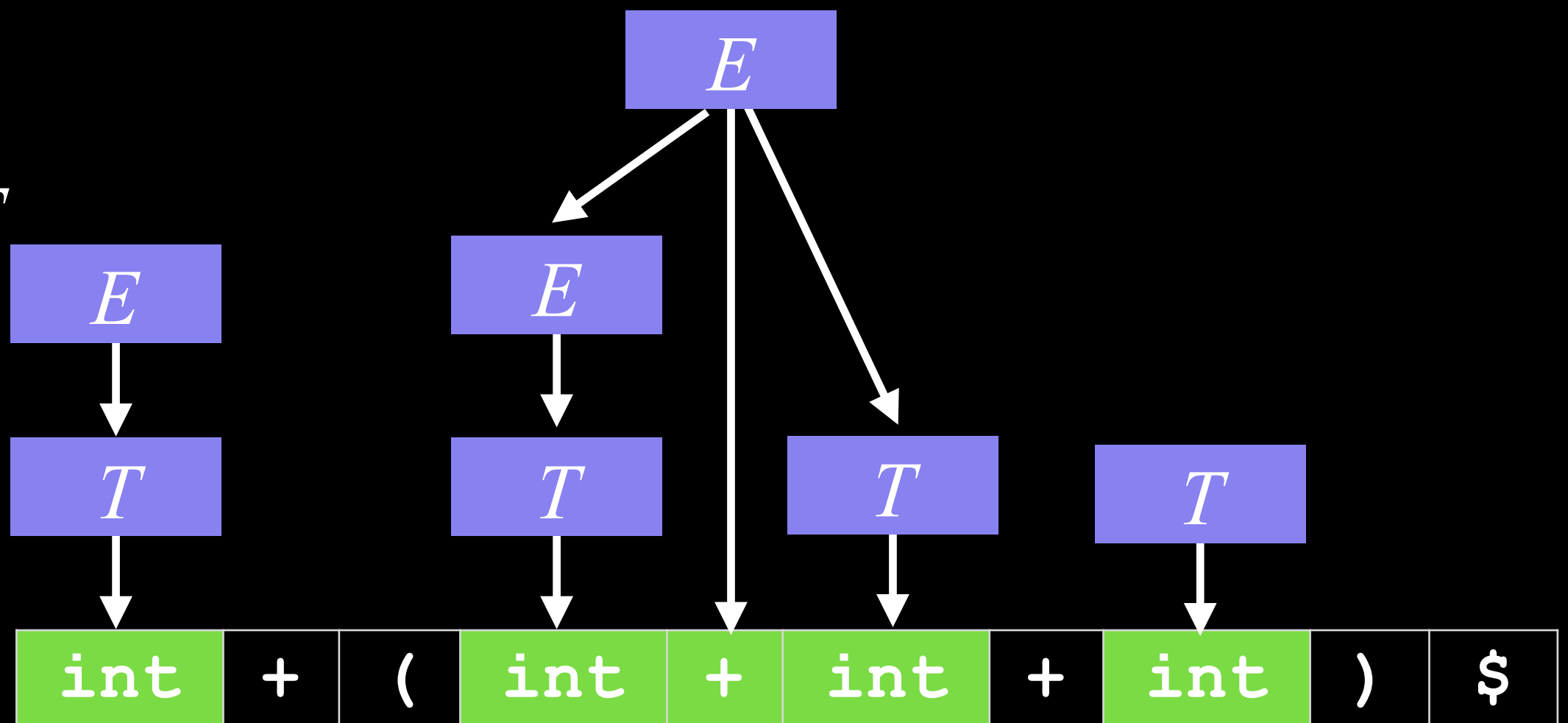
$S \rightarrow E\$$

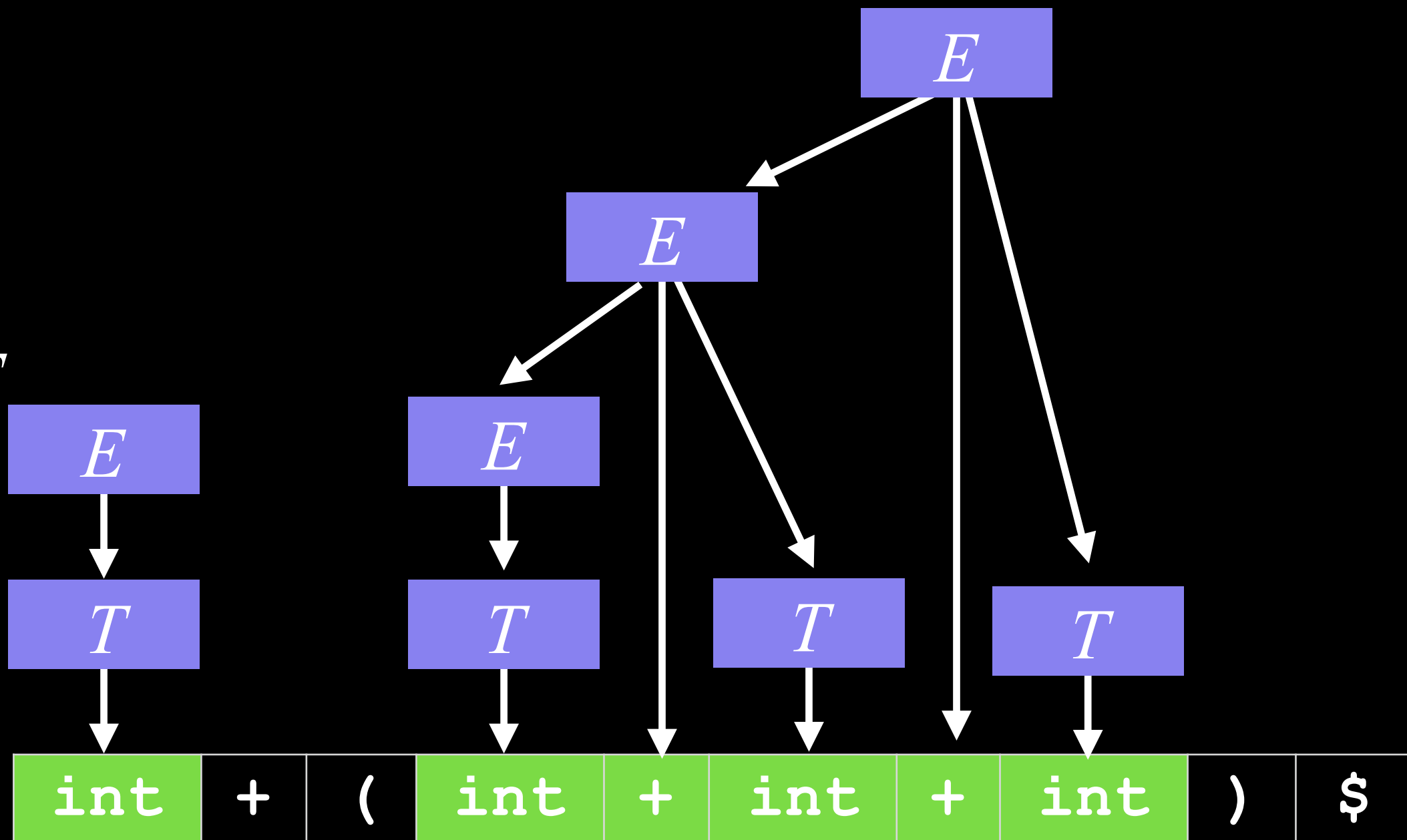
$E \rightarrow T$

$E \rightarrow E + T$

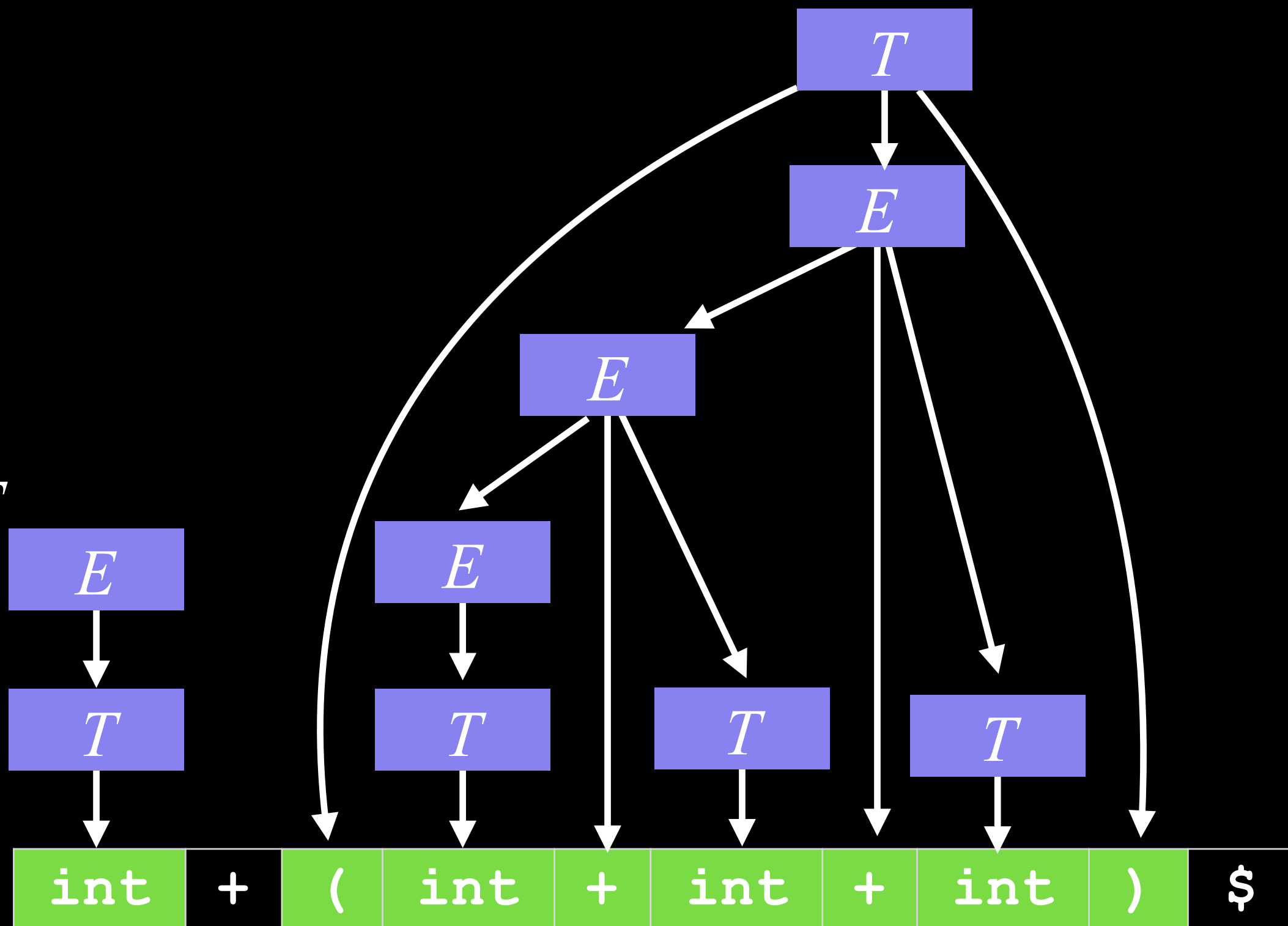
$T \rightarrow \text{int}$

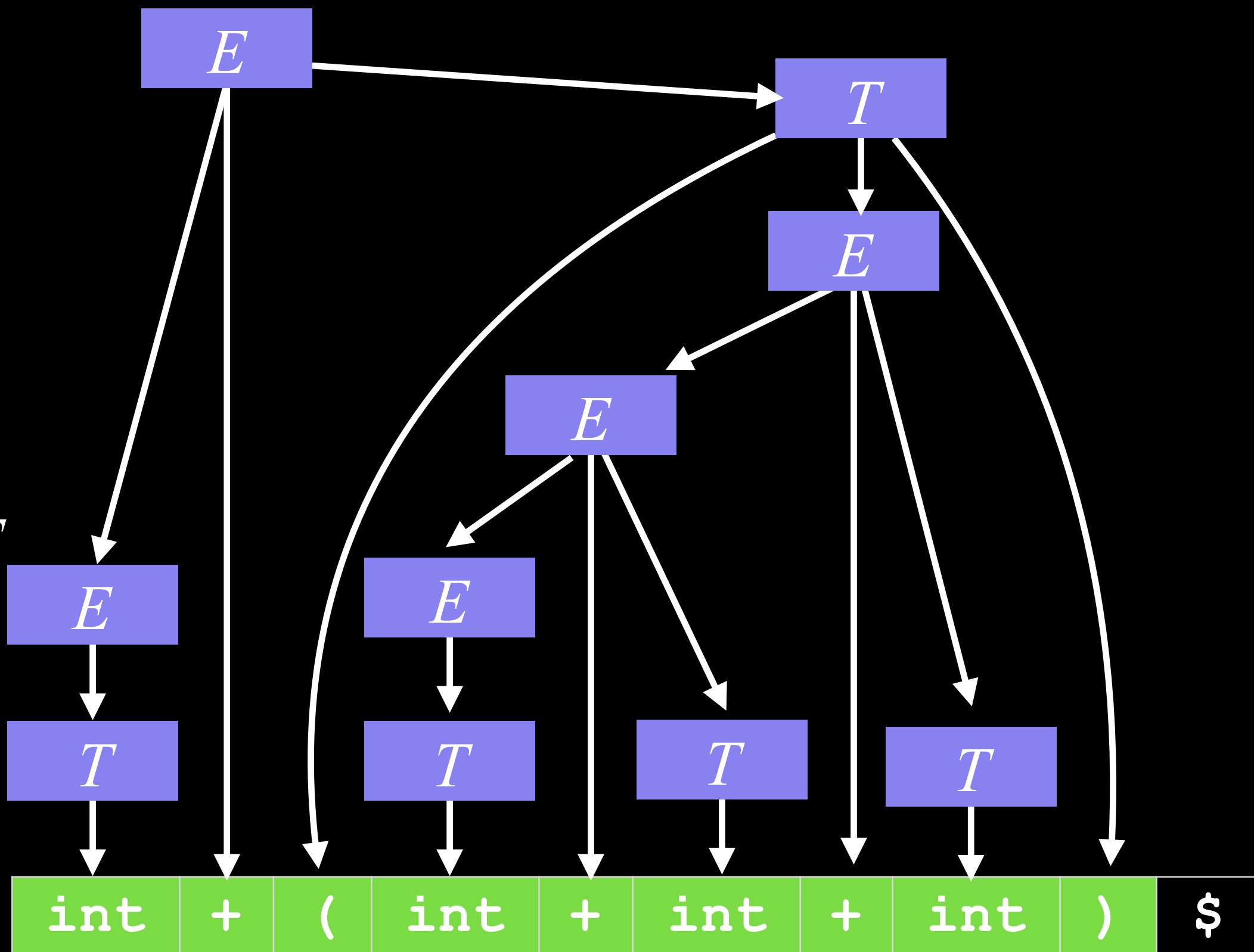
$T \rightarrow (E)$

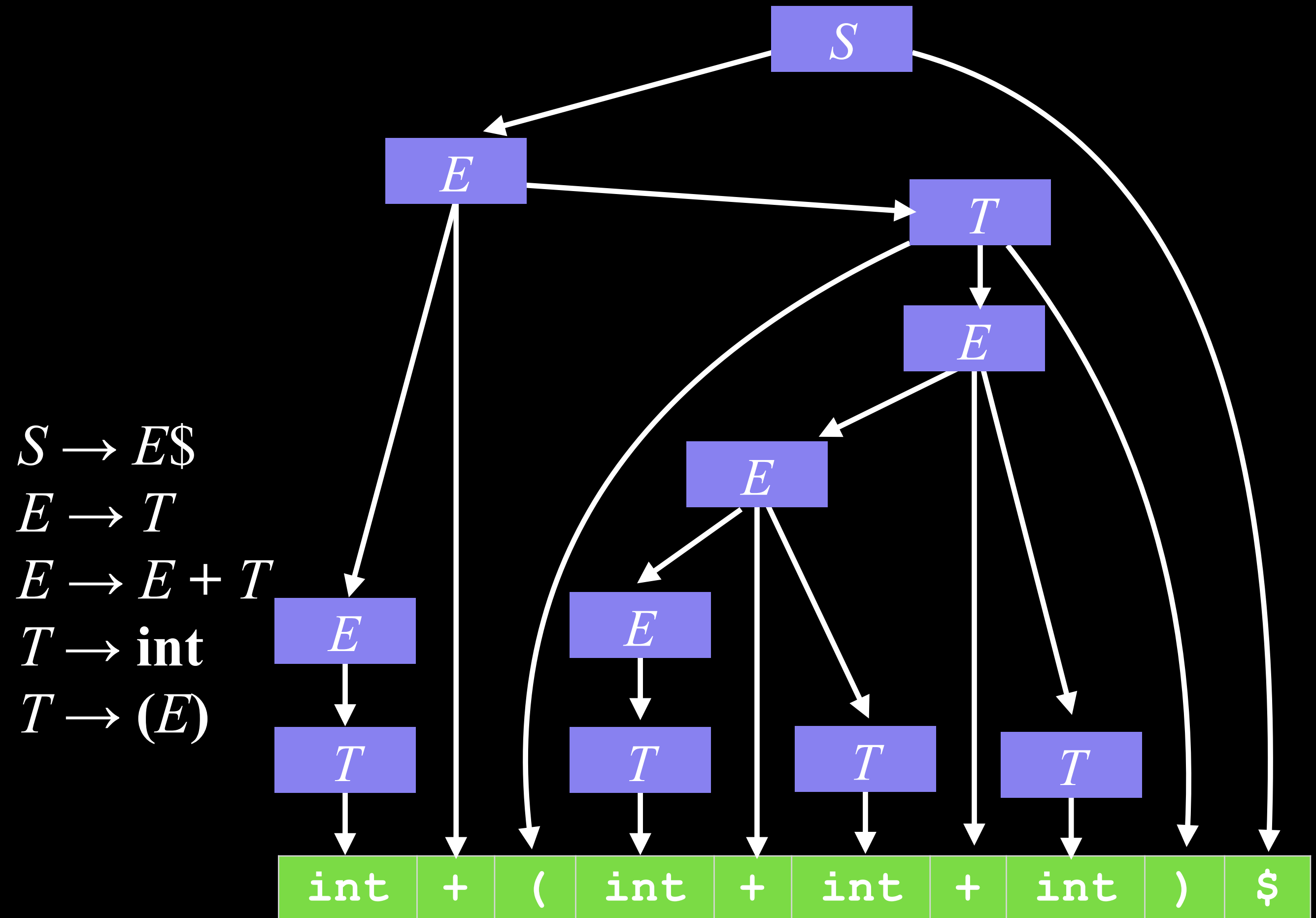


$$S \rightarrow ES$$
$$E \rightarrow T$$
$$E \rightarrow E + T$$
$$T \rightarrow \mathbf{int}$$
$$T \rightarrow (E)$$


$S \rightarrow ES$
 $E \rightarrow T$
 $E \rightarrow E + T$
 $T \rightarrow \text{int}$
 $T \rightarrow (E)$



$$\begin{aligned} S &\rightarrow E\$ \\ E &\rightarrow T \\ E &\rightarrow E + T \\ T &\rightarrow \mathbf{int} \\ T &\rightarrow (E) \end{aligned}$$




Outra visualização...

$S \rightarrow E\$$

$E \rightarrow T$

$E \rightarrow E + T$

int	+	(int	+	int	+	int)	\$
-----	---	---	-----	---	-----	---	-----	---	----

$T \rightarrow \text{int}$

$T \rightarrow (E)$

int	+	(int	+	int	+	int)	\$
-----	---	---	-----	---	-----	---	-----	---	----

$S \rightarrow E\$$

$E \rightarrow T$

$E \rightarrow E + T$

$T \rightarrow \text{int}$

$T \rightarrow (E)$

int	+	(int	+	int	+	int)	\$
<i>T</i>	+	(int	+	int	+	int)	\$

$S \rightarrow E\$$

$E \rightarrow T$

$E \rightarrow E + T$

$T \rightarrow \text{int}$

$T \rightarrow (E)$

int	+	(int	+	int	+	int)	\$
<i>T</i>	+	(int	+	int	+	int)	\$
<i>E</i>	+	(int	+	int	+	int)	\$

$S \rightarrow E\$$

$E \rightarrow T$

$E \rightarrow E + T$

$T \rightarrow \text{int}$

$T \rightarrow (E)$

int	+	(int	+	int	+	int)	\$
<i>T</i>	+	(int	+	int	+	int)	\$
<i>E</i>	+	(int	+	int	+	int)	\$
<i>E</i>	+	(<i>T</i>	+	int	+	int)	\$

$S \rightarrow E\$$

$E \rightarrow T$

$E \rightarrow E + T$

$T \rightarrow \text{int}$

$T \rightarrow (E)$

int	+	(int	+	int	+	int)	\$
<i>T</i>	+	(int	+	int	+	int)	\$
<i>E</i>	+	(int	+	int	+	int)	\$
<i>E</i>	+	(<i>T</i>	+	int	+	int)	\$
<i>E</i>	+	(<i>E</i>	+	int	+	int)	\$

$S \rightarrow E\$$

$E \rightarrow T$

$E \rightarrow E + T$

$T \rightarrow \text{int}$

$T \rightarrow (E)$

$$S \rightarrow E\$$$

$$E \rightarrow T$$

$$E \rightarrow E + T$$

$$T \rightarrow \text{int}$$

$$T \rightarrow (E)$$

int	+	(int	+	int	+	int)	\$
<i>T</i>	+	(int	+	int	+	int)	\$
<i>E</i>	+	(int	+	int	+	int)	\$
<i>E</i>	+	(<i>T</i>	+	int	+	int)	\$
<i>E</i>	+	(<i>E</i>	+	int	+	int)	\$
<i>E</i>	+	(<i>E</i>	+	<i>T</i>	+	int)	\$

$$S \rightarrow E\$$$

$$E \rightarrow T$$

$$E \rightarrow E + T$$

$$T \rightarrow \text{int}$$

$$T \rightarrow (E)$$

int	+	(int	+	int	+	int)	\$
<i>T</i>	+	(int	+	int	+	int)	\$
<i>E</i>	+	(int	+	int	+	int)	\$
<i>E</i>	+	(<i>T</i>	+	int	+	int)	\$
<i>E</i>	+	(<i>E</i>	+	int	+	int)	\$
<i>E</i>	+	(<i>E</i>	+	<i>T</i>	+	int)	\$
<i>E</i>	+	(<i>E</i>			+	int)	\$

$$S \rightarrow E\$$$

$$E \rightarrow T$$

$$E \rightarrow E + T$$

$$T \rightarrow \text{int}$$

$$T \rightarrow (E)$$

int	+	(int	+	int	+	int)	\$
<i>T</i>	+	(int	+	int	+	int)	\$
<i>E</i>	+	(int	+	int	+	int)	\$
<i>E</i>	+	(<i>T</i>	+	int	+	int)	\$
<i>E</i>	+	(<i>E</i>	+	int	+	int)	\$
<i>E</i>	+	(<i>E</i>	+	<i>T</i>	+	int)	\$
<i>E</i>	+	(<i>E</i>			+	int)	\$
<i>E</i>	+	(<i>E</i>			+	<i>T</i>)	\$

$$S \rightarrow E\$$$

$$E \rightarrow T$$

$$E \rightarrow E + T$$

$$T \rightarrow \text{int}$$

$$T \rightarrow (E)$$

int	+	(int	+	int	+	int)	\$
<i>T</i>	+	(int	+	int	+	int)	\$
<i>E</i>	+	(int	+	int	+	int)	\$
<i>E</i>	+	(<i>T</i>	+	int	+	int)	\$
<i>E</i>	+	(<i>E</i>	+	int	+	int)	\$
<i>E</i>	+	(<i>E</i>	+	<i>T</i>	+	int)	\$
<i>E</i>	+	(<i>E</i>			+	int)	\$
<i>E</i>	+	(<i>E</i>			+	<i>T</i>)	\$
<i>E</i>	+	(<i>E</i>)	\$

$$S \rightarrow E\$$$
$$E \rightarrow T$$
$$E \rightarrow E + T$$
$$T \rightarrow \mathbf{int}$$
$$T \rightarrow (E)$$

int	+	(int	+	int	+	int)	\$
<i>T</i>	+	(int	+	int	+	int)	\$
<i>E</i>	+	(int	+	int	+	int)	\$
<i>E</i>	+	(<i>T</i>	+	int	+	int)	\$
<i>E</i>	+	(<i>E</i>	+	int	+	int)	\$
<i>E</i>	+	(<i>E</i>	+	<i>T</i>	+	int)	\$
<i>E</i>	+	(<i>E</i>			+	int)	\$
<i>E</i>	+	(<i>E</i>			+	<i>T</i>)	\$
<i>E</i>	+	(<i>E</i>)	\$
<i>E</i>	+	<i>T</i>							\$

$$S \rightarrow E\$$$
$$E \rightarrow T$$
$$E \rightarrow E + T$$
$$T \rightarrow \mathbf{int}$$
$$T \rightarrow (E)$$

int	+	(int	+	int	+	int)	\$
T	+	(int	+	int	+	int)	\$
E	+	(int	+	int	+	int)	\$
E	+	(T	+	int	+	int)	\$
E	+	(E	+	int	+	int)	\$
E	+	(E	+	T	+	int)	\$
E	+	(E			+	int)	\$
E	+	(E			+	T)	\$
E	+	(E)	\$
E	+	T							\$
E									\$

$$S \rightarrow E\$$$
$$E \rightarrow T$$
$$E \rightarrow E + T$$
$$T \rightarrow \mathbf{int}$$
$$T \rightarrow (E)$$

int	+	(int	+	int	+	int)	\$
<i>T</i>	+	(int	+	int	+	int)	\$
<i>E</i>	+	(int	+	int	+	int)	\$
<i>E</i>	+	(<i>T</i>	+	int	+	int)	\$
<i>E</i>	+	(<i>E</i>	+	int	+	int)	\$
<i>E</i>	+	(<i>E</i>	+	<i>T</i>	+	int)	\$
<i>E</i>	+	(<i>E</i>			+	int)	\$
<i>E</i>	+	(<i>E</i>			+	<i>T</i>)	\$
<i>E</i>	+	(<i>E</i>)	\$
<i>E</i>	+	<i>T</i>							\$
<i>E</i>									\$
<i>S</i>									

Reduções e Derivações

- a análise ascendente lê a entrada da esquerda pra direita, 'construindo a árvore de baixo pra cima'
- aplica sequência de reduções de produções
 - corresponde a uma sequência de derivações mais à direita, lida de trás pra frente
- para uma gramática não ambígua, cada entrada só pode ter uma única derivação à direita
 - portanto, sequência de reduções também é única

Derivações

$$S \rightarrow E\$$$

$$E \rightarrow T$$

$$E \rightarrow E + T$$

$$T \rightarrow \text{int}$$

$$T \rightarrow (E)$$

int	+	(int	+	int	+	int)	\$
<i>T</i>	+	(int	+	int	+	int)	\$
<i>E</i>	+	(int	+	int	+	int)	\$
<i>E</i>	+	(<i>T</i>	+	int	+	int)	\$
<i>E</i>	+	(<i>E</i>	+	int	+	int)	\$
<i>E</i>	+	(<i>E</i>	+	<i>T</i>	+	int)	\$
<i>E</i>	+	(<i>E</i>			+	int)	\$
<i>E</i>	+	(<i>E</i>			+	<i>T</i>)	\$
<i>E</i>	+	(<i>E</i>)	\$
<i>E</i>	+	<i>T</i>							\$
<i>E</i>									\$
<i>S</i>									

S

$$\rightarrow E\$$$

$$\rightarrow E + T\$$$

$$\rightarrow E + (E) \$$$

$$\rightarrow E + (E + T) \$$$

$$\rightarrow E + (E + \text{int}) \$$$

$$\rightarrow E + (E + T + \text{int}) \$$$

$$\rightarrow E + (E + \text{int} + \text{int}) \$$$

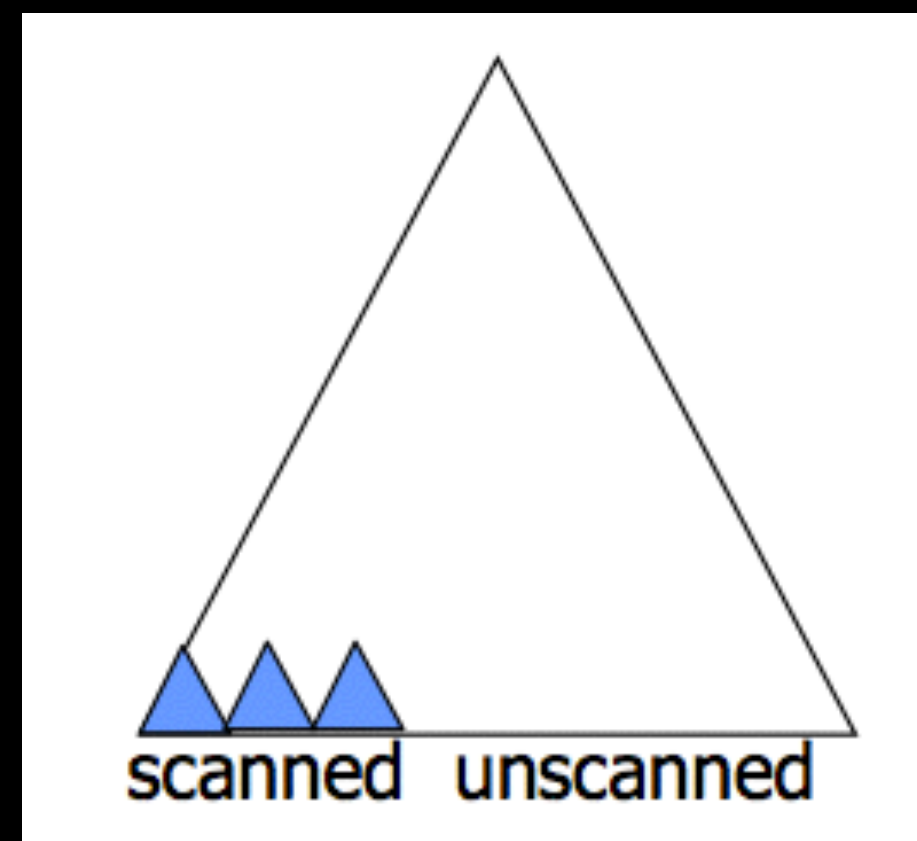
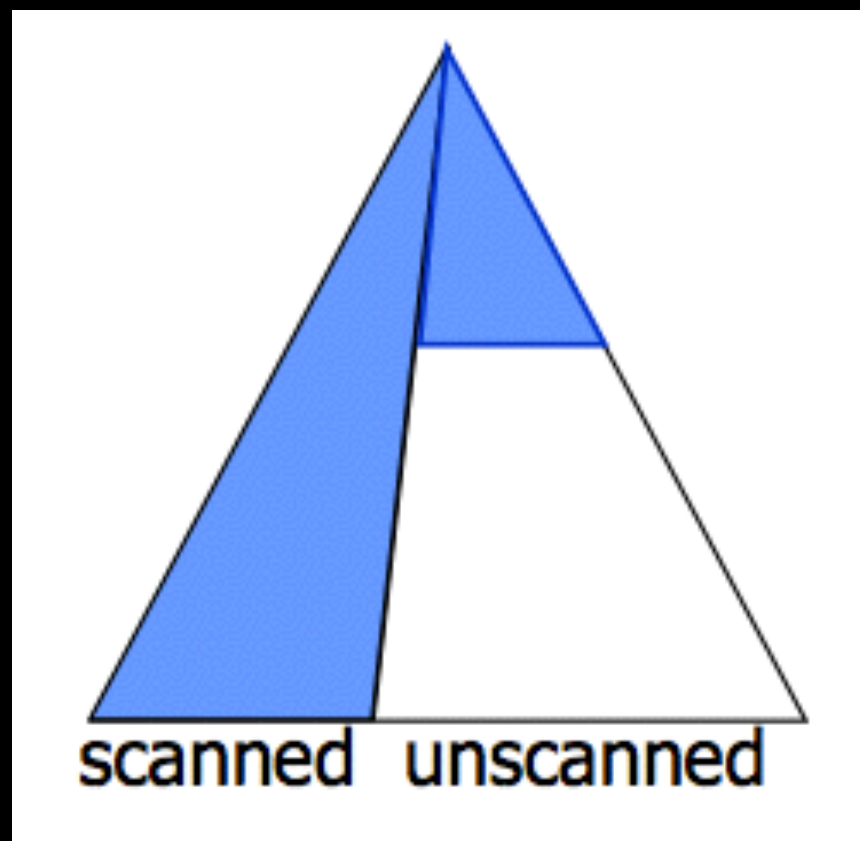
$$\rightarrow E + (T + \text{int} + \text{int}) \$$$

$$\rightarrow E + (\text{int} + \text{int} + \text{int}) \$$$

$$\rightarrow T + (\text{int} + \text{int} + \text{int}) \$$$

$$\rightarrow \text{int} + (\text{int} + \text{int} + \text{int}) \$$$

Top-Down vs. Bottom-UP



Handles e reduções

- Uma redução transforma a string $\mathbf{u}\mathbf{w}\mathbf{v}$ em $\mathbf{u}\mathbf{A}\mathbf{v}$ se $\mathbf{A} \rightarrow \mathbf{w}$ é uma produção da gramática
- Um handle é uma substring \mathbf{w} e uma produção $\mathbf{A} \rightarrow \mathbf{w}$ tal que, reduzindo $\mathbf{u}\mathbf{w}\mathbf{v} \rightarrow \mathbf{u}\mathbf{A}\mathbf{v}$ permite que o símbolo inicial seja alcançado de $\mathbf{u}\mathbf{A}\mathbf{v}$
- Informalmente, uma produção que podemos reduzir sem que seja gerado um problema

Cuidado com Handles

$$S \rightarrow E\$$$

$$E \rightarrow F$$

$$E \rightarrow E + F$$

$$F \rightarrow F * T$$

$$F \rightarrow T$$

$$T \rightarrow \text{int}$$

$$T \rightarrow (E)$$

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

$S \rightarrow E\$$

$E \rightarrow F$

$E \rightarrow E + F$

$F \rightarrow F * T$

$F \rightarrow T$

$T \rightarrow \text{int}$

$T \rightarrow (E)$



$S \rightarrow E\$$

$E \rightarrow F$

$E \rightarrow E + F$

$F \rightarrow F * T$

$F \rightarrow T$

$T \rightarrow \text{int}$

$T \rightarrow (E)$



$S \rightarrow E\$$

$E \rightarrow F$

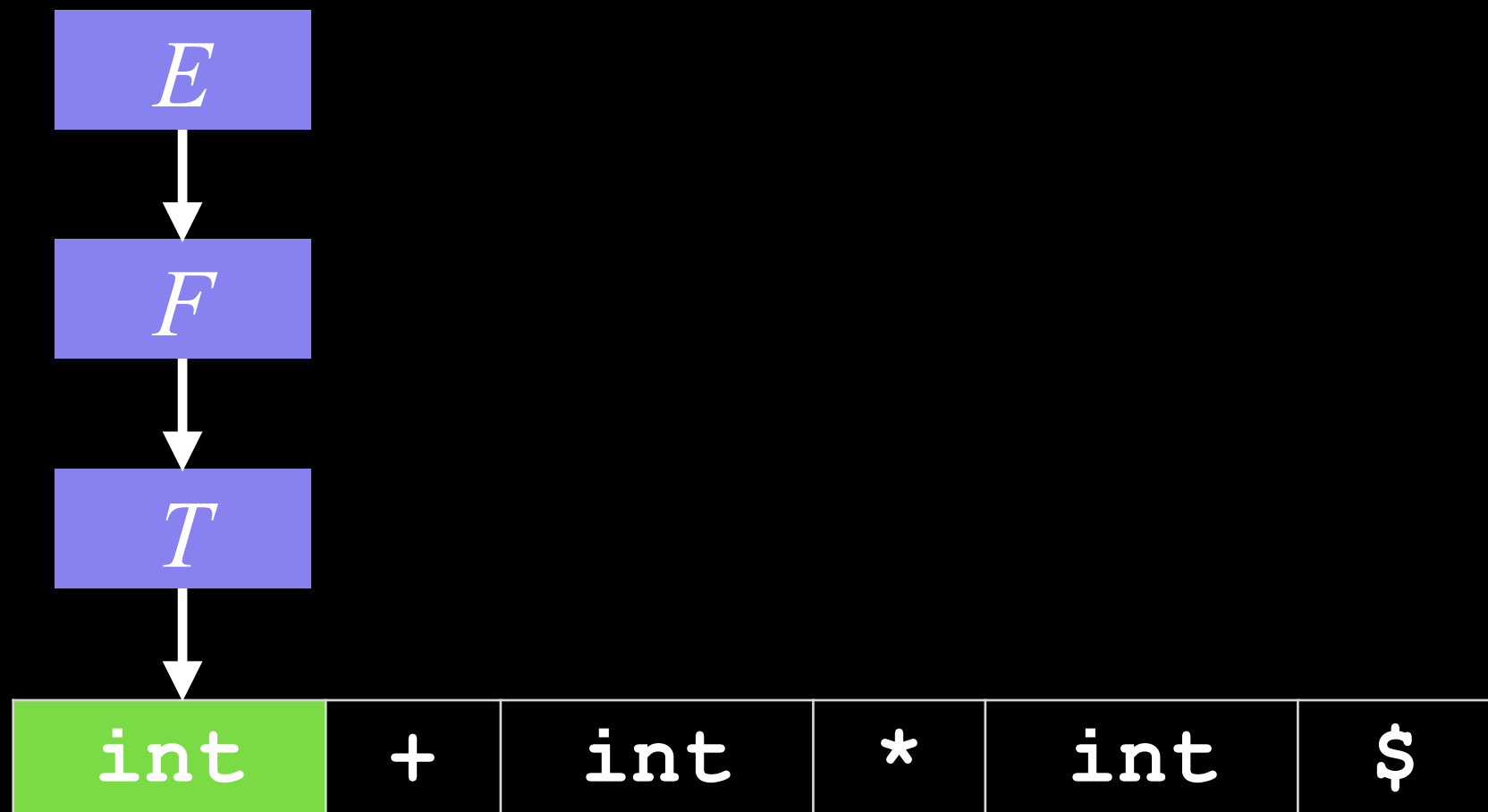
$E \rightarrow E + F$

$F \rightarrow F * T$

$F \rightarrow T$

$T \rightarrow \text{int}$

$T \rightarrow (E)$



$S \rightarrow E\$$

$E \rightarrow F$

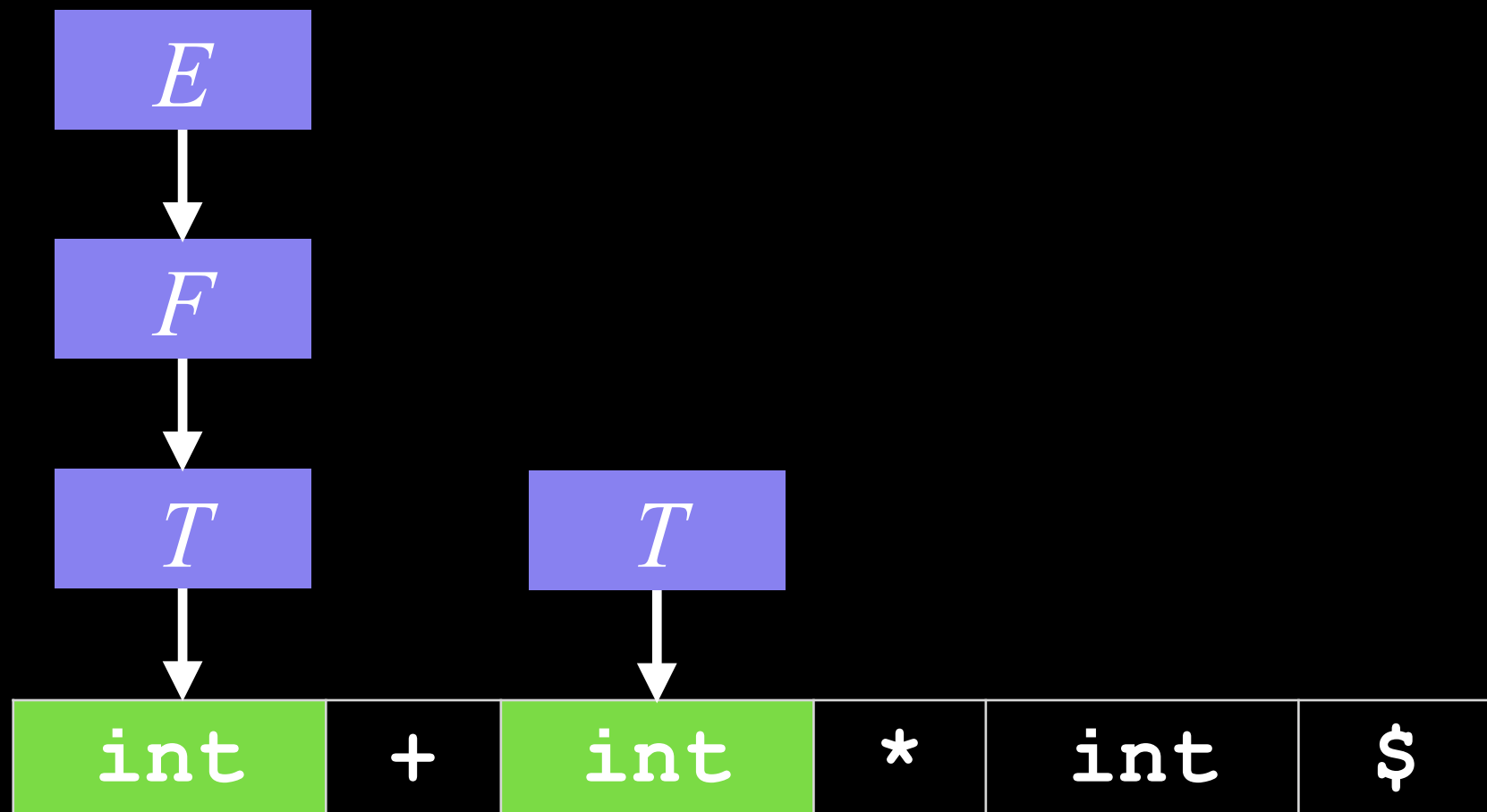
$E \rightarrow E + F$

$F \rightarrow F * T$

$F \rightarrow T$

$T \rightarrow \text{int}$

$T \rightarrow (E)$



$S \rightarrow E\$$

$E \rightarrow F$

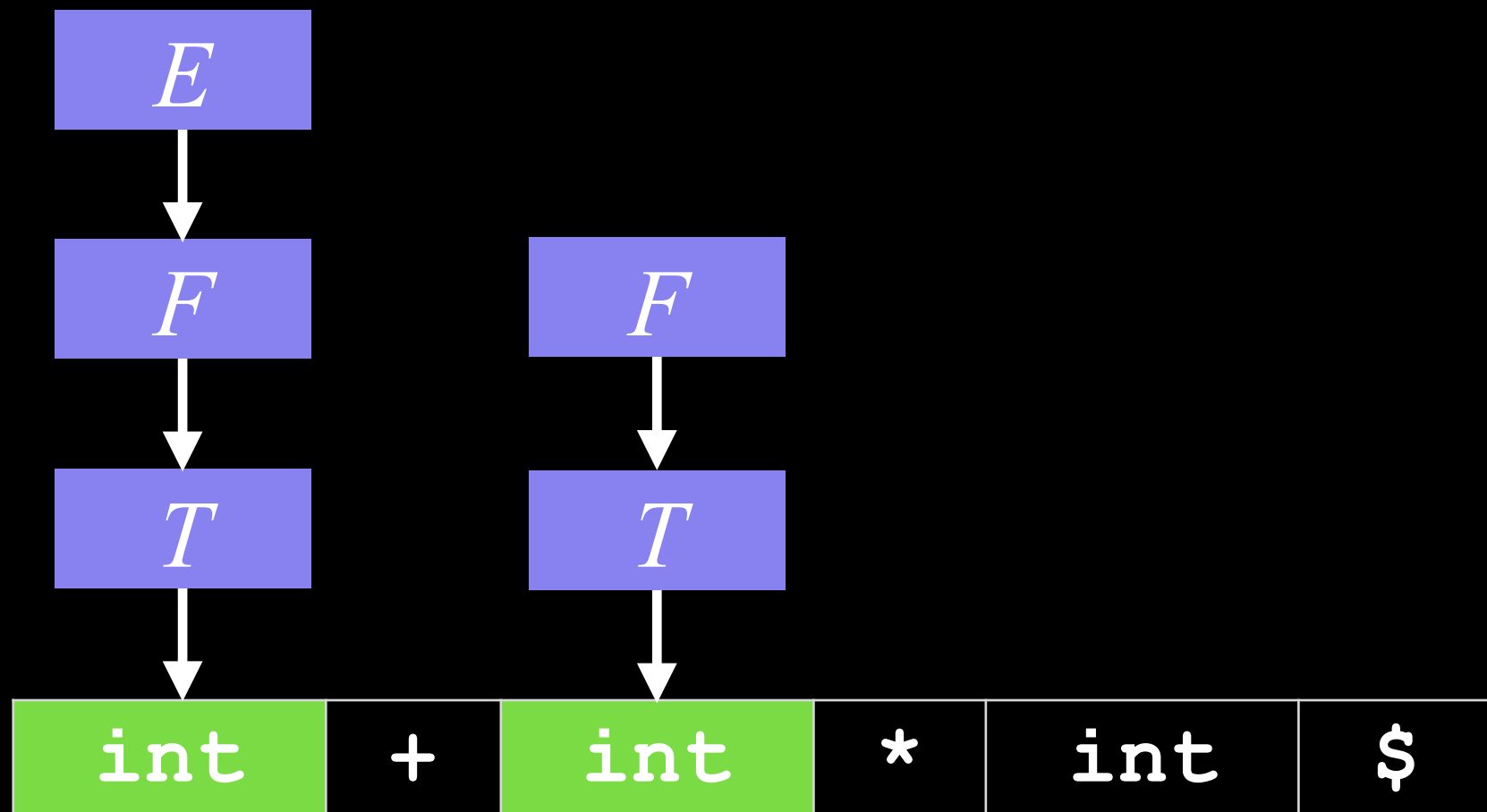
$E \rightarrow E + F$

$F \rightarrow F * T$

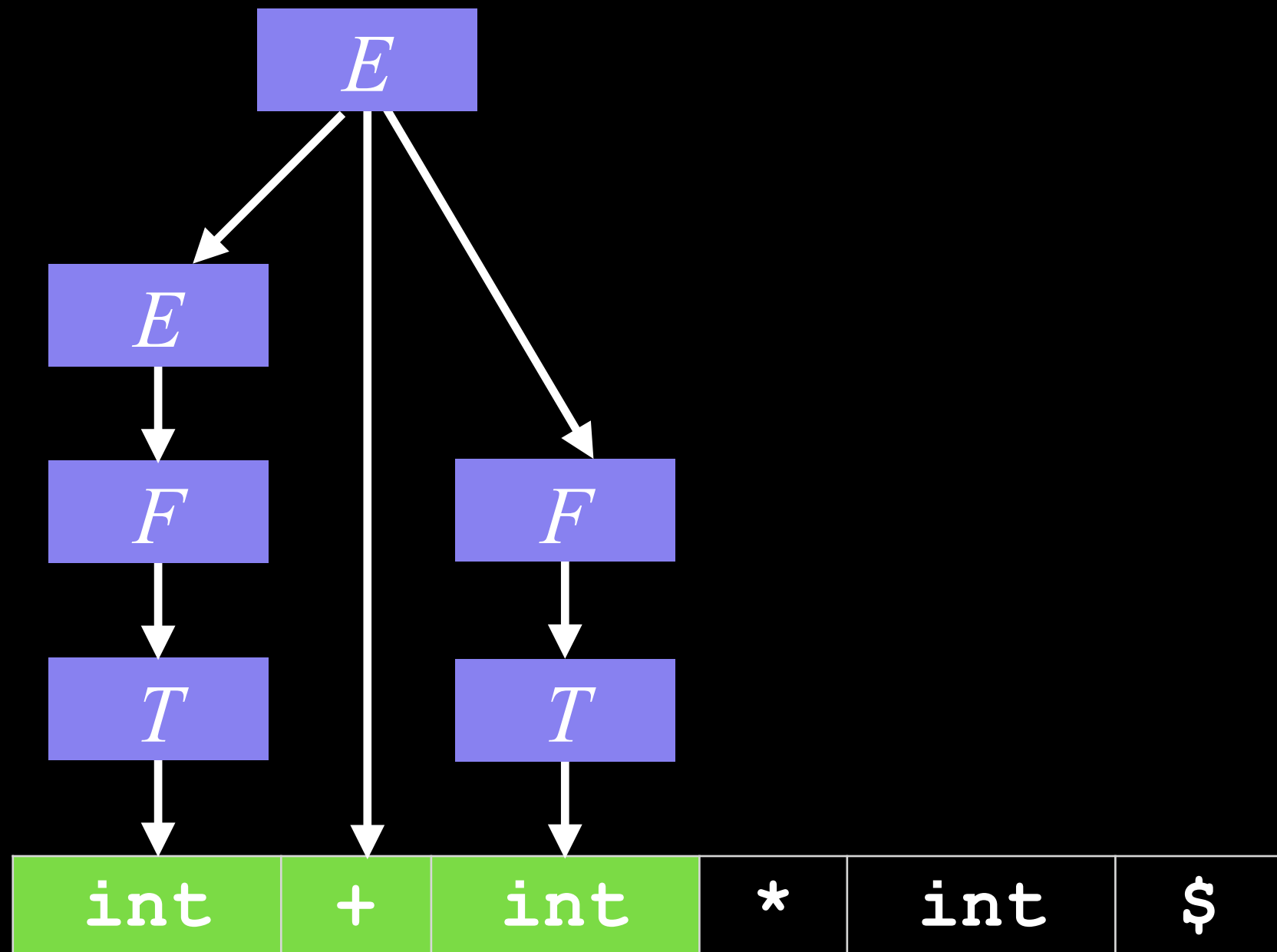
$F \rightarrow T$

$T \rightarrow \text{int}$

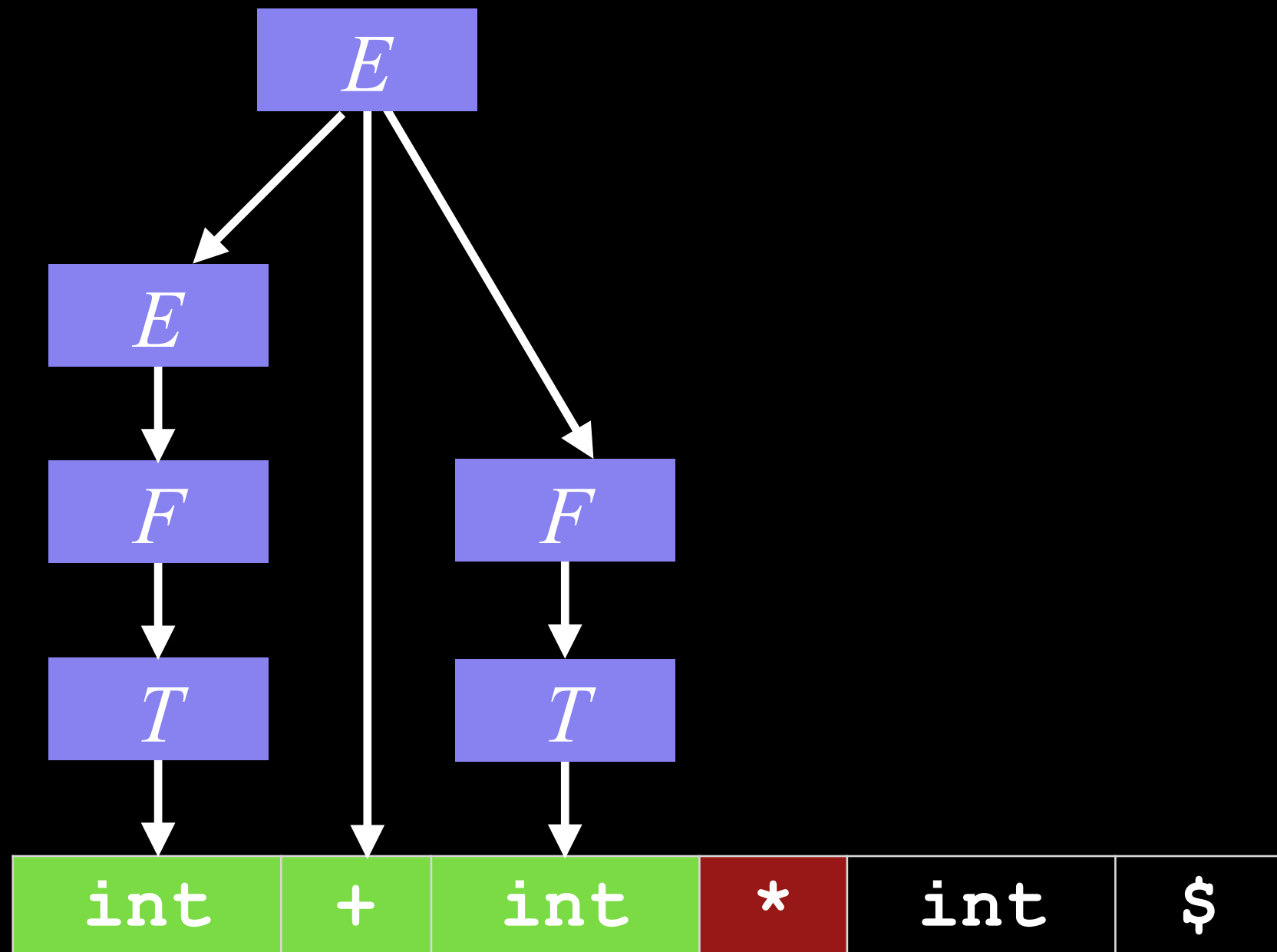
$T \rightarrow (E)$



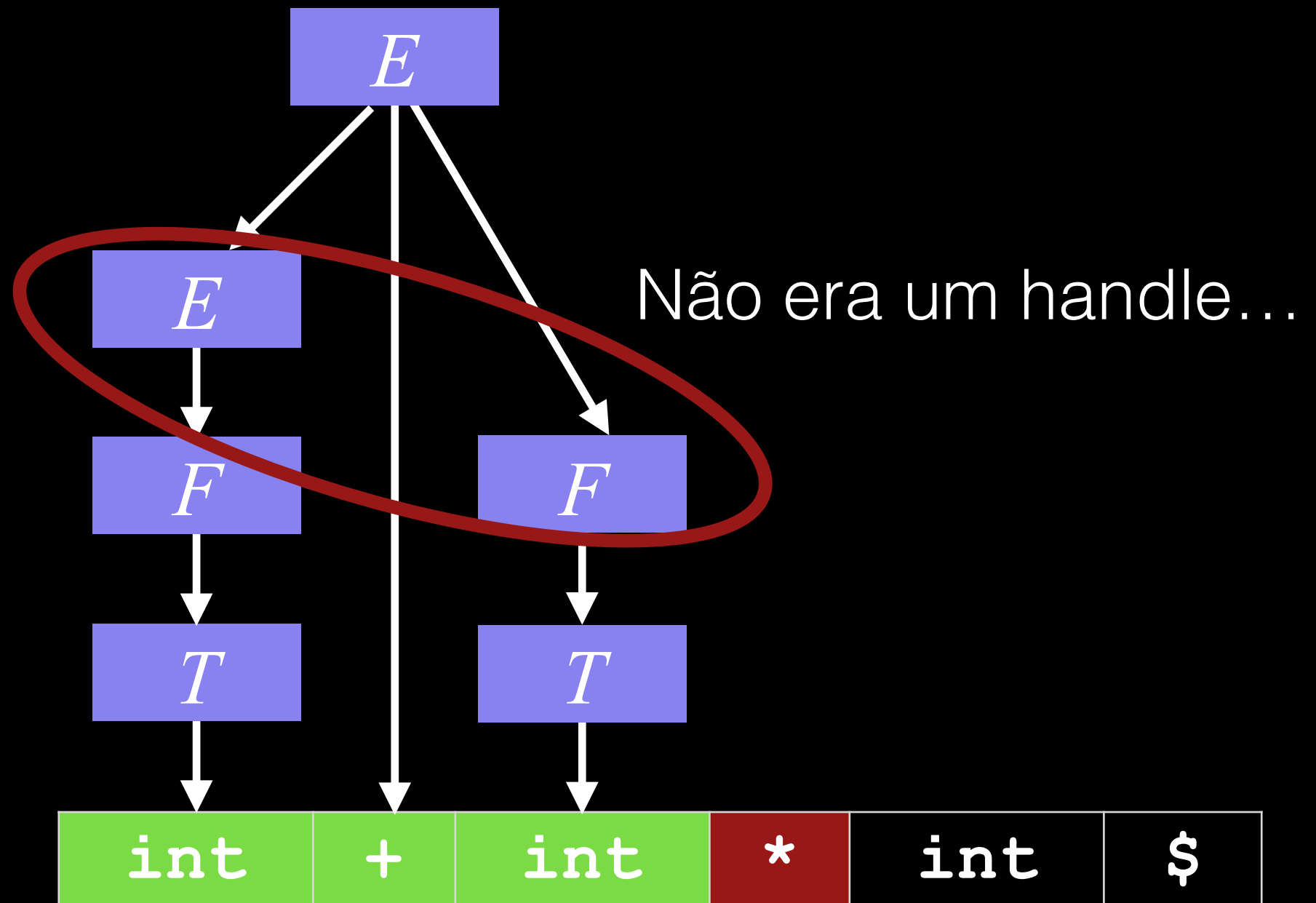
$S \rightarrow E\$$
 $E \rightarrow F$
 $E \rightarrow E + F$
 $F \rightarrow F * T$
 $F \rightarrow T$
 $T \rightarrow \text{int}$
 $T \rightarrow (E)$



$S \rightarrow E\$$
 $E \rightarrow F$
 $E \rightarrow E + F$
 $F \rightarrow F * T$
 $F \rightarrow T$
 $T \rightarrow \text{int}$
 $T \rightarrow (E)$



$S \rightarrow E\$$
 $E \rightarrow F$
 $E \rightarrow E + F$
 $F \rightarrow F * T$
 $F \rightarrow T$
 $T \rightarrow \text{int}$
 $T \rightarrow (E)$



Como identificar os handles?

Como saber quando reduzir?

Análise shift-reduce

- Considere o passo da análise que transforma a string $\mathbf{u}w\mathbf{v}$ em $\mathbf{u}A\mathbf{v}$ usando a regra $A \rightarrow w$
- A substring \mathbf{v} consiste apenas de terminais, pois a redução corresponde ao passo $\mathbf{u}A\mathbf{v} \rightarrow \mathbf{u}w\mathbf{v}$ de uma derivação mais à direita
- Então, a cada passo da análise, temos um sufixo que corresponde ao restante da entrada que ainda não foi reduzida

Análise shift-reduce

- Ideia geral: dividir a entrada em duas partes
 - Substring da esquerda e direita
- Divisão geralmente ilustrada com um marcador: | ou •
 - À direita do foco existem terminais ainda não reduzidos
 - À esquerda existem terminais e não-terminais
 - A parte mais à direita (ou imediatamente à esquerda) contém um potencial candidato a handle (redução)

Análise shift-reduce

- Handles e reduções só ocorrem na substring da esquerda
- Substring da direita consiste apenas de terminais
- Então a análise consiste de, em cada passo decidir entre:
 - shift - deslocar o foco à direita
(jogando um terminal para a substring da esquerda)
 - reduce - aplicar uma redução a um handle

Análise shift-reduce

- Shift - mover o foco à direita
 - $\mathbf{A B C} \mid \mathbf{x y z} \rightarrow \mathbf{A B C x} \mid \mathbf{y z}$
- Reduce: reduz o que está imediatamente à esquerda do foco usando uma produção
 - Se $\mathbf{A} \rightarrow \mathbf{x y}$ é uma produção, então $\mathbf{C b x y} \mid \mathbf{l i j k} \rightarrow \mathbf{C b A} \mid \mathbf{l i j k}$ é uma ação reduce $\mathbf{A} \rightarrow \mathbf{x y}$
- Erro sintático ocorre quando não se pode tomar nenhuma das duas ações e a entrada é aceita quando chegamos a $\mathbf{S|}$, onde \mathbf{S} é o símbolo inicial

Shift-reduce parsing

$$S \rightarrow E\$$$

$$E \rightarrow F$$

$$E \rightarrow E + F$$

$$F \rightarrow F * T$$

$$F \rightarrow T$$

$$T \rightarrow \text{int}$$

$$T \rightarrow (E)$$

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

$S \rightarrow E\$$

$E \rightarrow F$


$E \rightarrow E + F$

$F \rightarrow F * T$

$F \rightarrow T$

$T \rightarrow \text{int}$

$T \rightarrow (E)$



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

$S \rightarrow E\$$

$E \rightarrow F$

$E \rightarrow E + F$

$F \rightarrow F * T$

$F \rightarrow T$

$T \rightarrow \text{int}$

$T \rightarrow (E)$

int

int

+

int

*

int

+

int

\$

$S \rightarrow E\$$

$E \rightarrow F$

$E \rightarrow E + F$

$F \rightarrow F * T$

$F \rightarrow T$

$T \rightarrow \text{int}$

$T \rightarrow (E)$

T



int

T



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

$S \rightarrow E\$$

$E \rightarrow F$

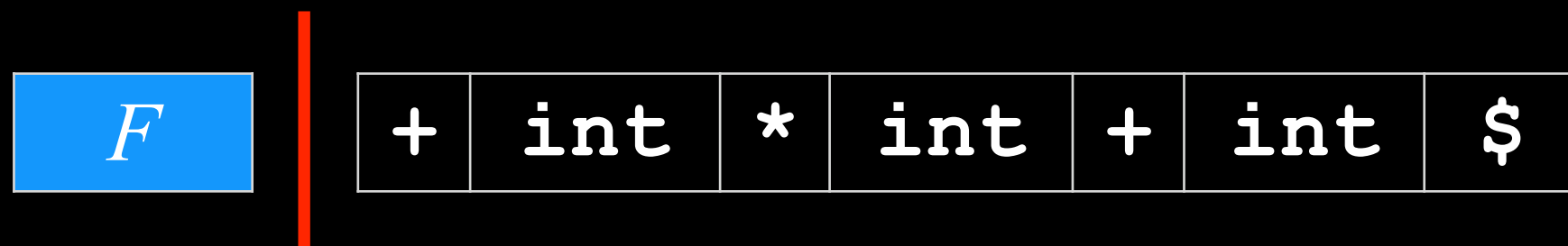
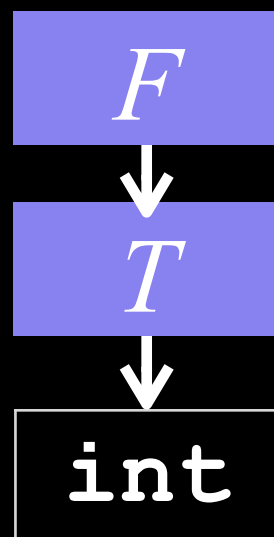
$E \rightarrow E + F$

$F \rightarrow F * T$

$F \rightarrow T$

$T \rightarrow \text{int}$

$T \rightarrow (E)$



$S \rightarrow E\$$

$E \rightarrow F$

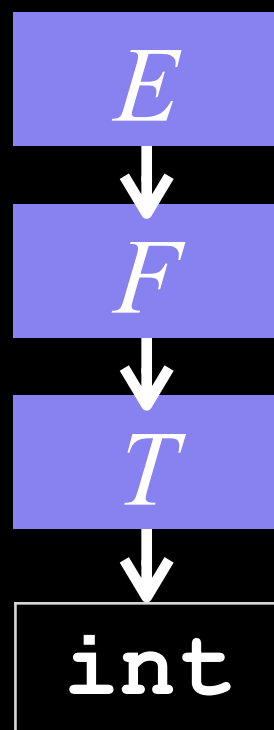
$E \rightarrow E + F$

$F \rightarrow F * T$

$F \rightarrow T$

$T \rightarrow \text{int}$

$T \rightarrow (E)$



E

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

$S \rightarrow E\$$

$E \rightarrow F$

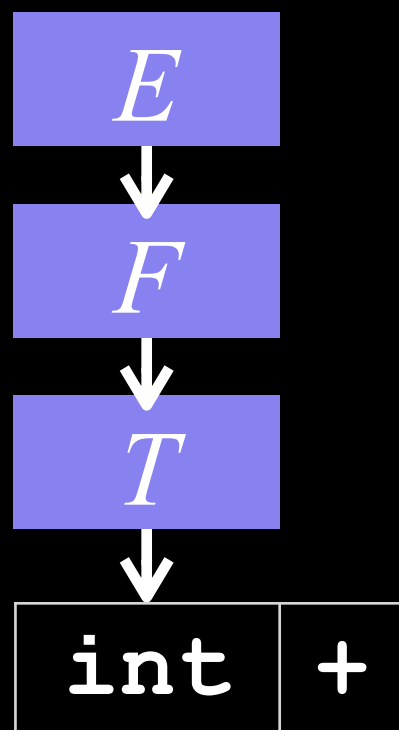
$E \rightarrow E + F$

$F \rightarrow F * T$

$F \rightarrow T$

$T \rightarrow \text{int}$

$T \rightarrow (E)$



$S \rightarrow E\$$

$E \rightarrow F$

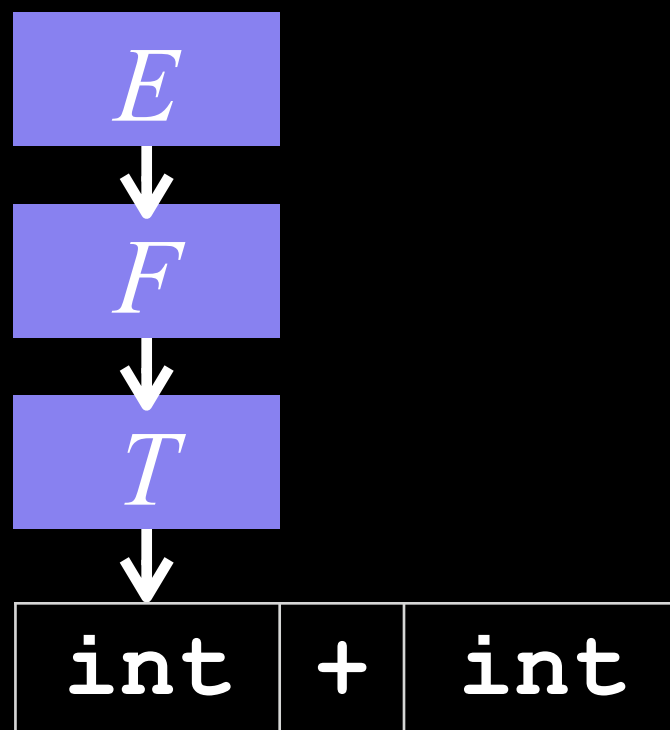
$E \rightarrow E + F$

$F \rightarrow F * T$

$F \rightarrow T$

$T \rightarrow \text{int}$

$T \rightarrow (E)$



E	<code>+</code>	<code>int</code>
-----	----------------	------------------

<code>*</code>	<code>int</code>	<code>+</code>	<code>int</code>	<code>\$</code>
----------------	------------------	----------------	------------------	-----------------

$S \rightarrow E\$$

$E \rightarrow F$

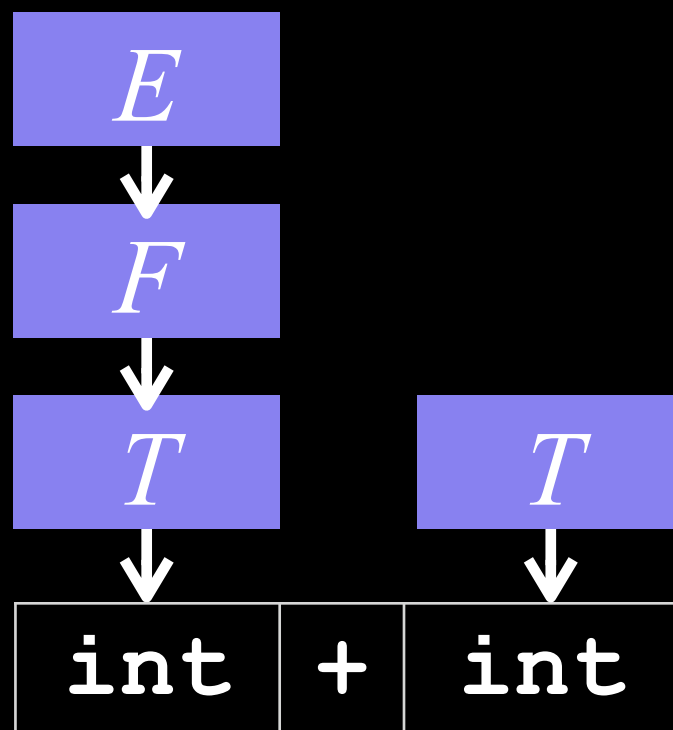
$E \rightarrow E + F$

$F \rightarrow F * T$

$F \rightarrow T$

$T \rightarrow \text{int}$

$T \rightarrow (E)$



E	+	T
-----	---	-----

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

$S \rightarrow E\$$

$E \rightarrow F$

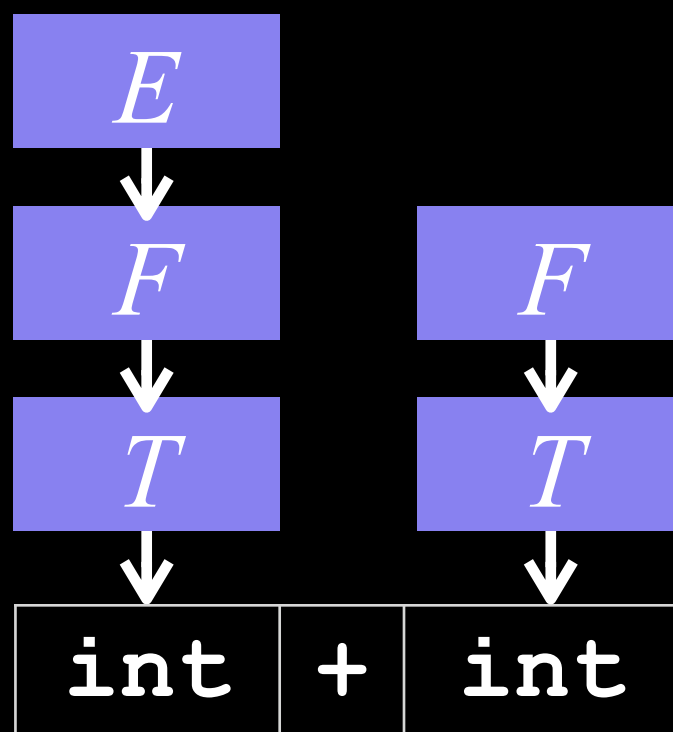
$E \rightarrow E + F$

$F \rightarrow F * T$

$F \rightarrow T$

$T \rightarrow \text{int}$

$T \rightarrow (E)$



E	+	F
-----	---	-----

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

$$S \rightarrow E\$$$

$$E \rightarrow F$$

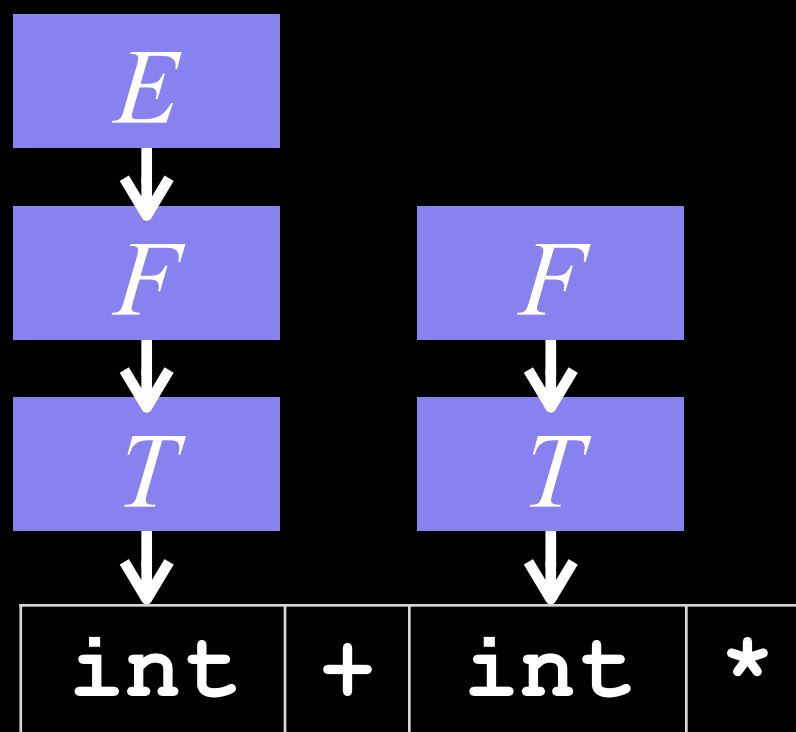
$$E \rightarrow E + F$$

$$F \rightarrow F * T$$

$$F \rightarrow T$$

$$T \rightarrow \text{int}$$

$$T \rightarrow (E)$$



E	+	F	*
-----	---	-----	---

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

$S \rightarrow E\$$

$E \rightarrow F$

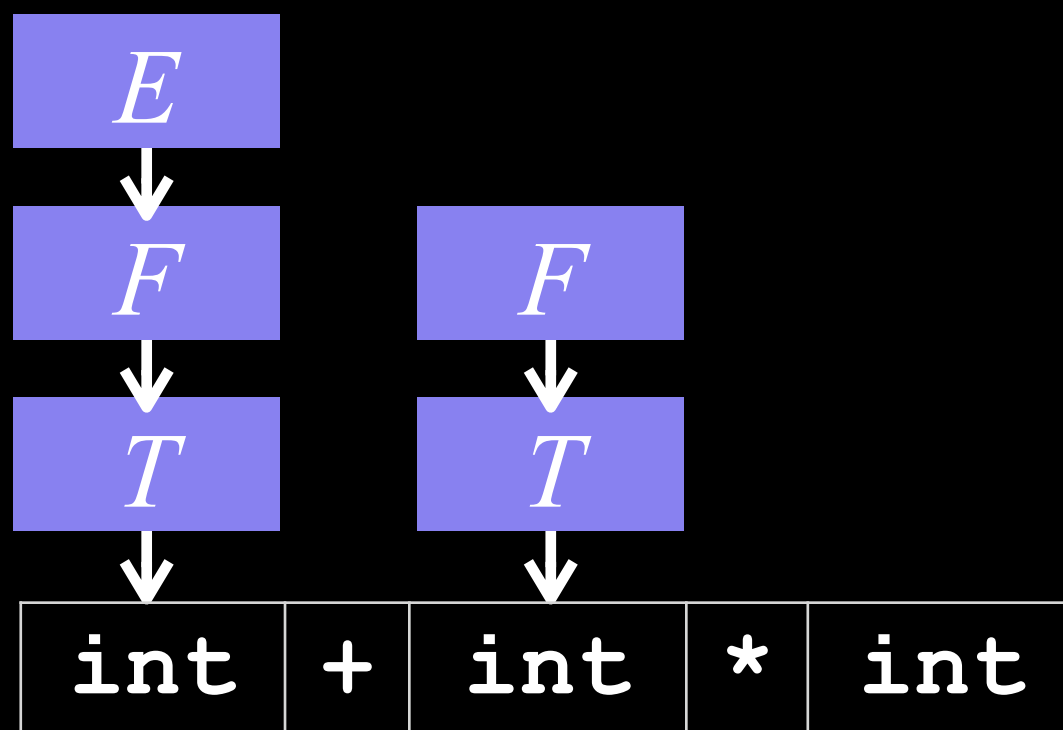
$E \rightarrow E + F$

$F \rightarrow F * T$

$F \rightarrow T$

$T \rightarrow \text{int}$

$T \rightarrow (E)$



E	+	F	*	int
-----	---	-----	---	-----

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

$$S \rightarrow E\$$$

$$E \rightarrow F$$

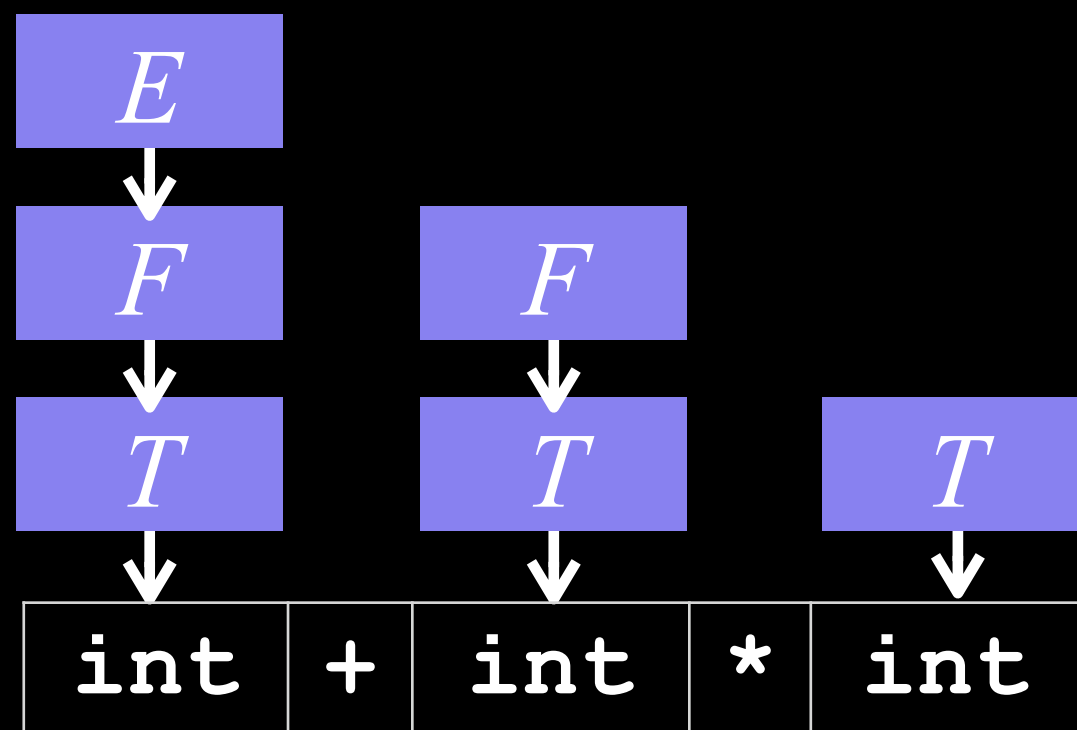
$$E \rightarrow E + F$$

$$F \rightarrow F * T$$

$$F \rightarrow T$$

$$T \rightarrow \text{int}$$

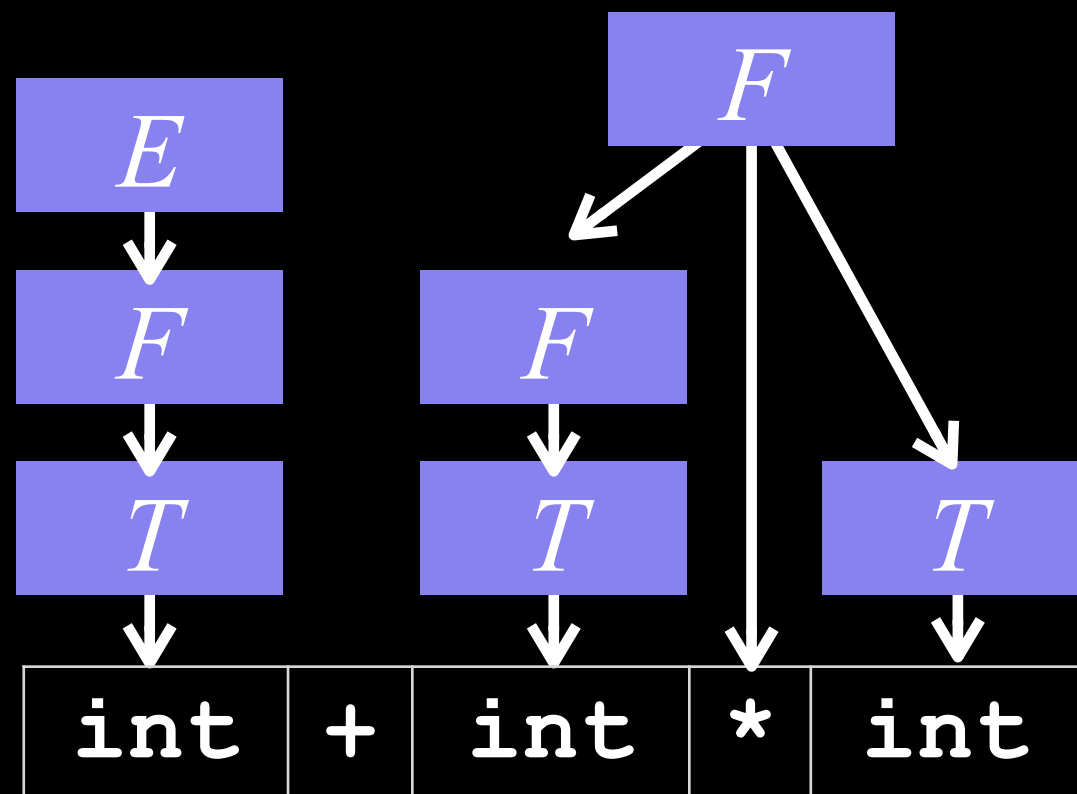
$$T \rightarrow (E)$$



E	$+$	F	$*$	T
-----	-----	-----	-----	-----

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

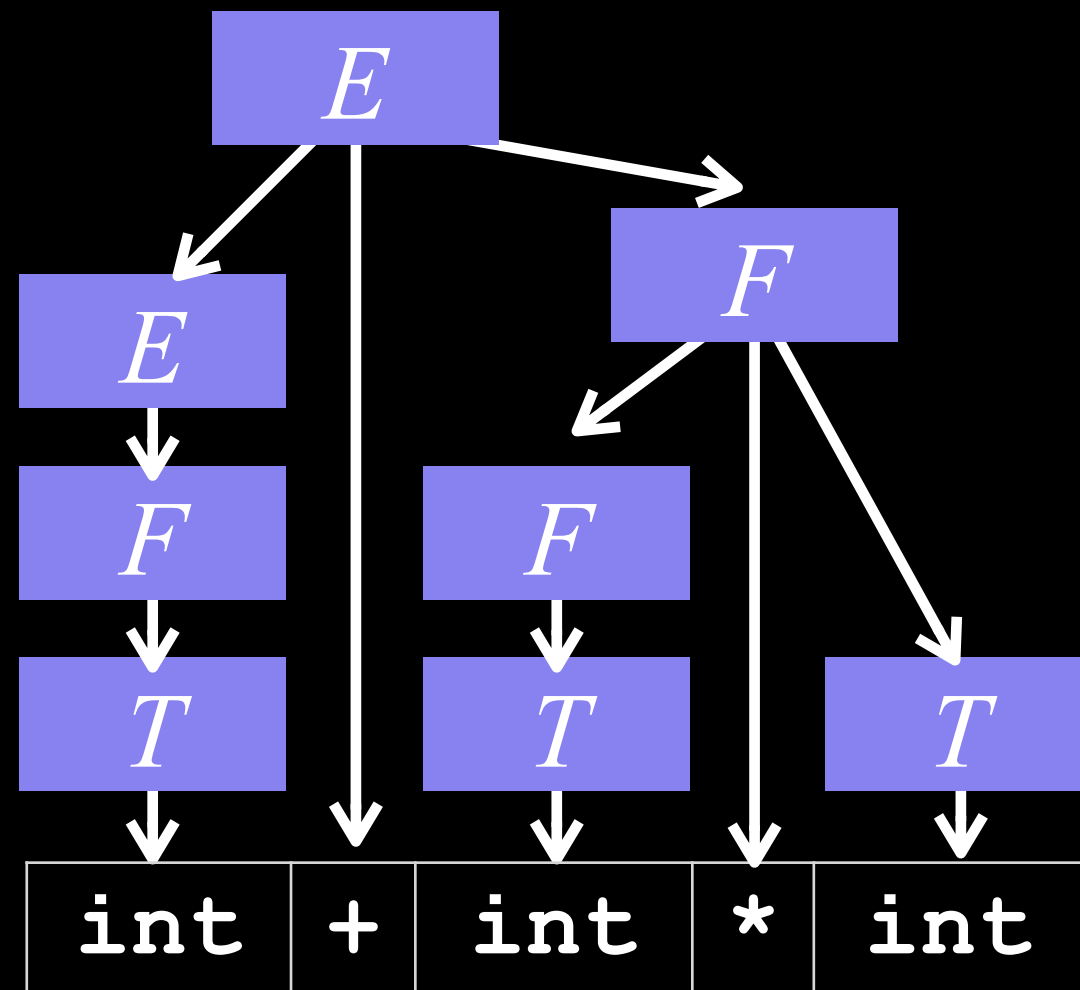
$S \rightarrow E\$$
 $E \rightarrow F$
 $E \rightarrow E + F$
 $F \rightarrow F * T$
 $F \rightarrow T$
 $T \rightarrow \text{int}$
 $T \rightarrow (E)$



E	+	F
-----	---	-----

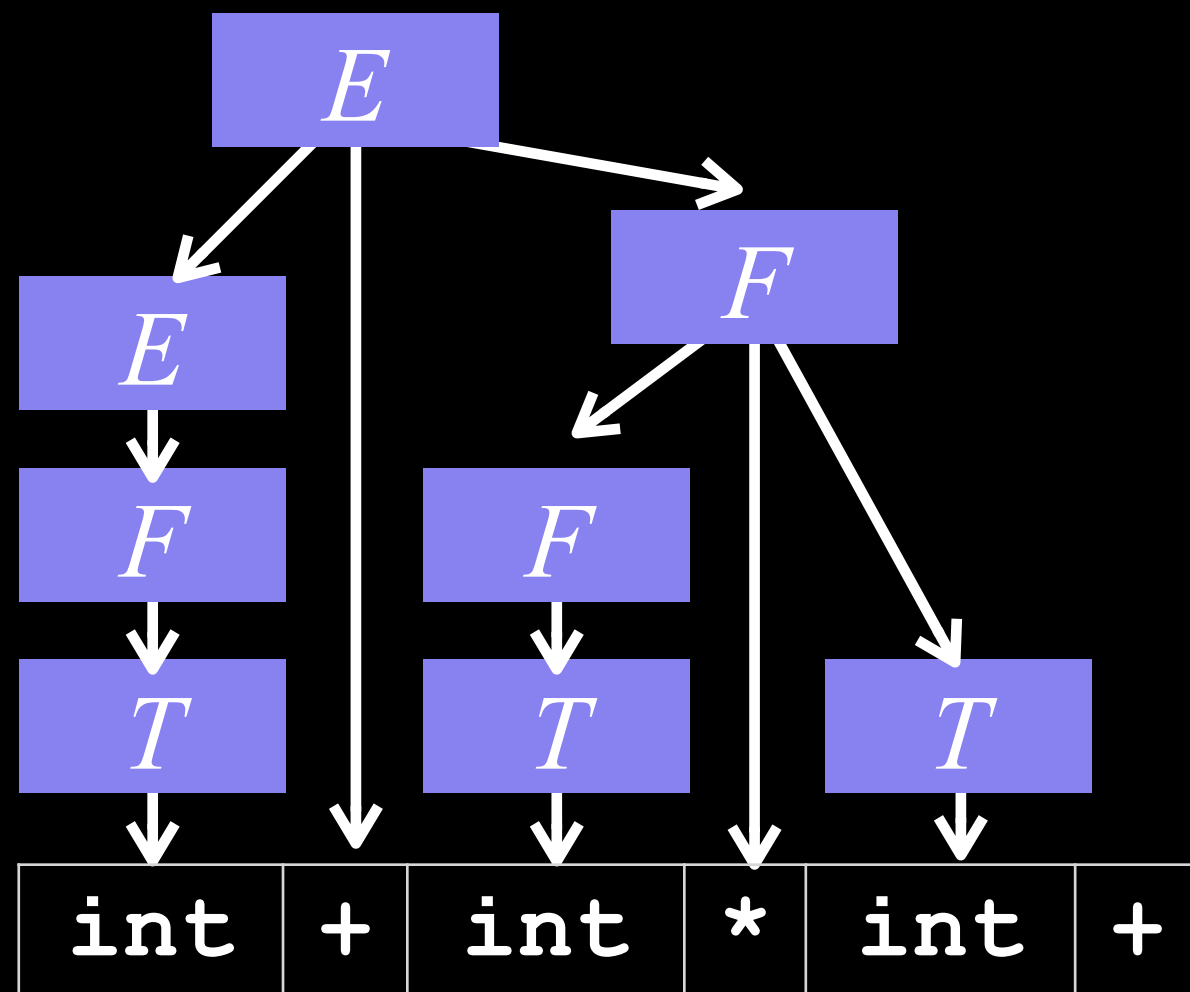
+	int	\$
---	-----	----

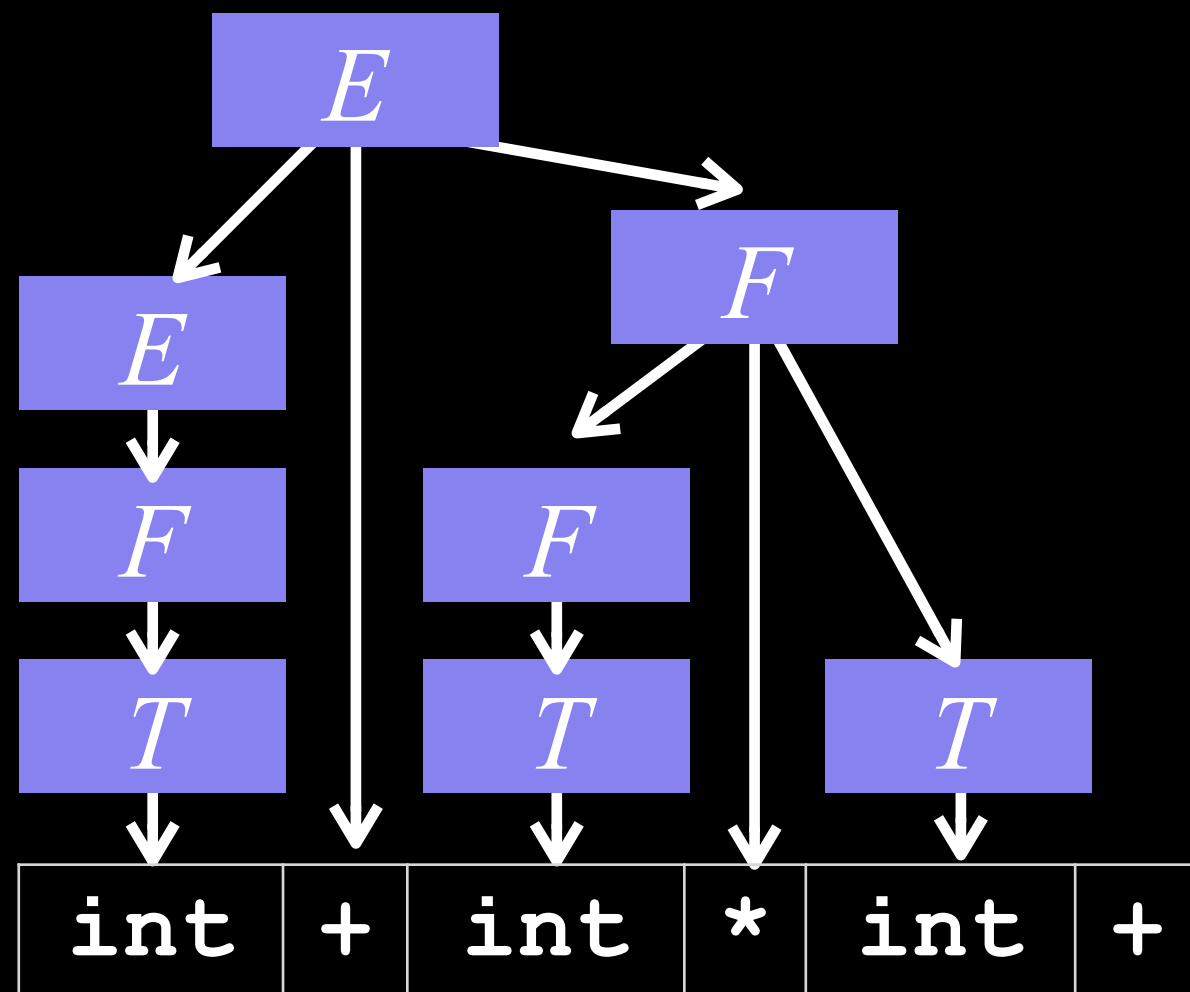
$S \rightarrow E\$$
 $E \rightarrow F$
 $E \rightarrow E + F$
 $F \rightarrow F * T$
 $F \rightarrow T$
 $T \rightarrow \text{int}$
 $T \rightarrow (E)$

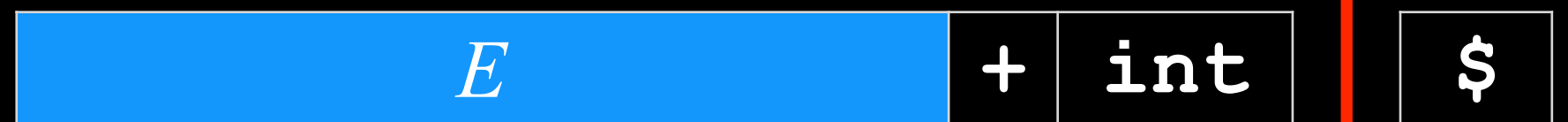
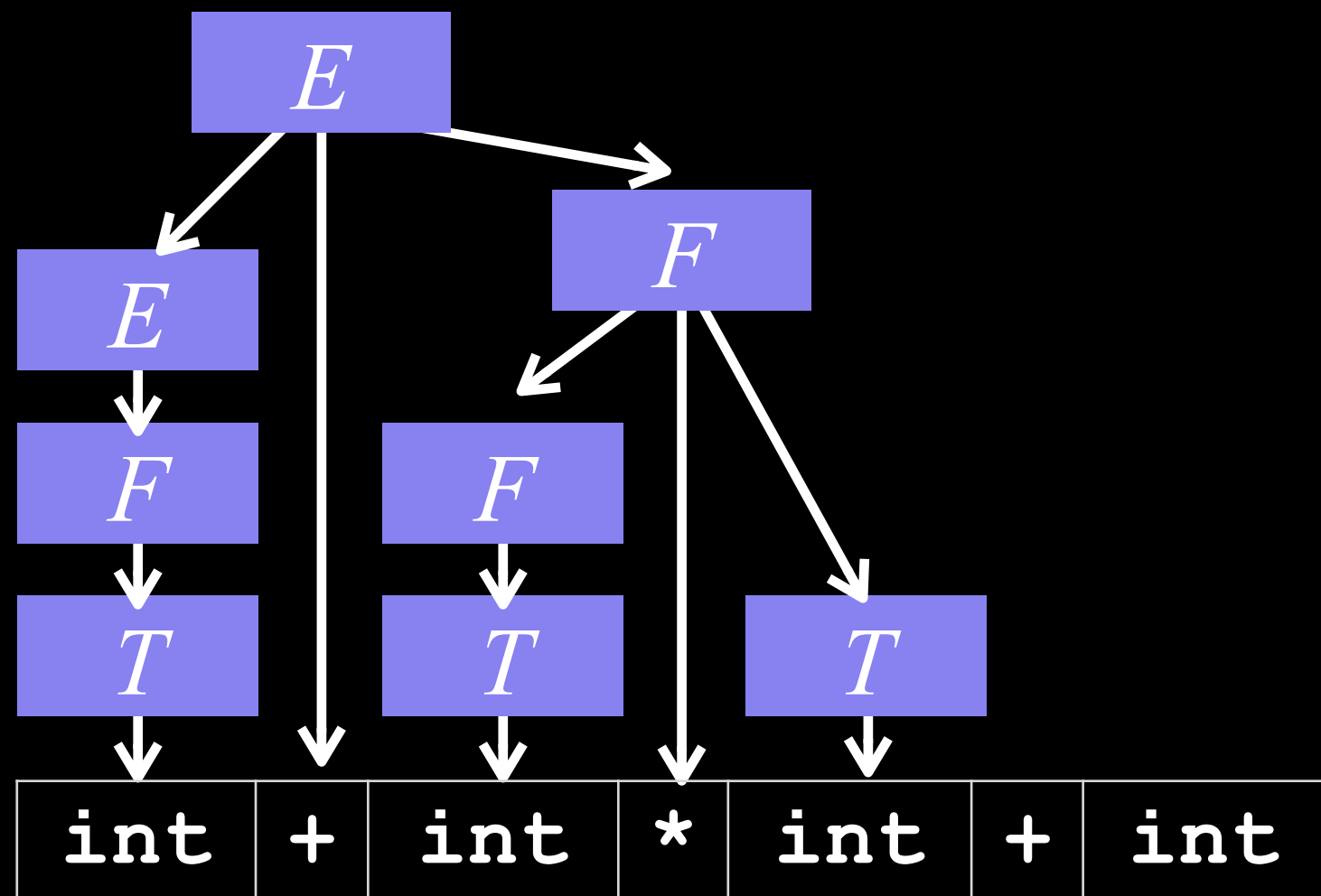


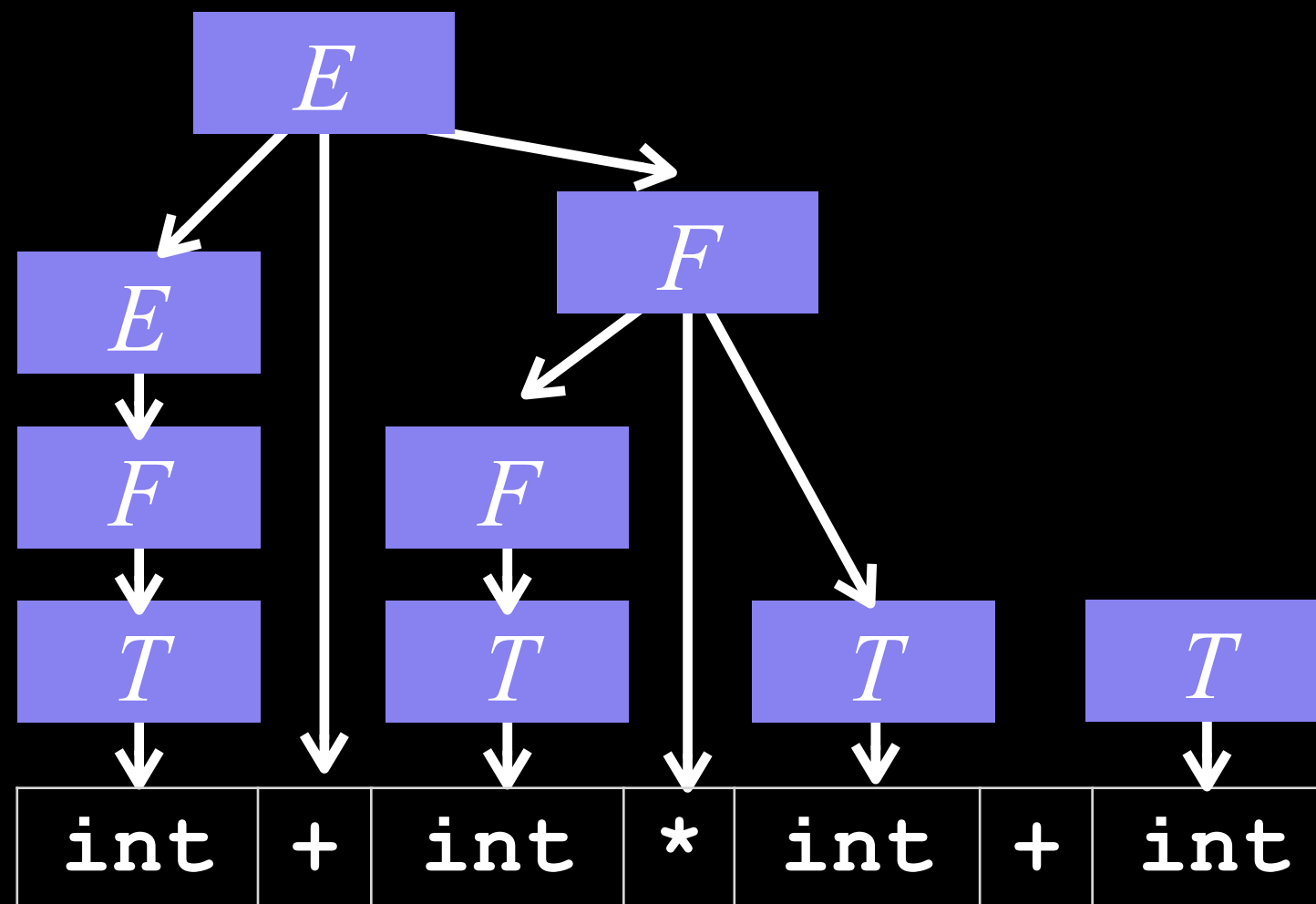
E

$+$ int $\$$

$$\begin{aligned}
 S &\rightarrow E\$ \\
 E &\rightarrow F \\
 E &\rightarrow E + F \\
 F &\rightarrow F * T \\
 F &\rightarrow T \\
 T &\rightarrow \text{int} \\
 T &\rightarrow (E)
 \end{aligned}$$


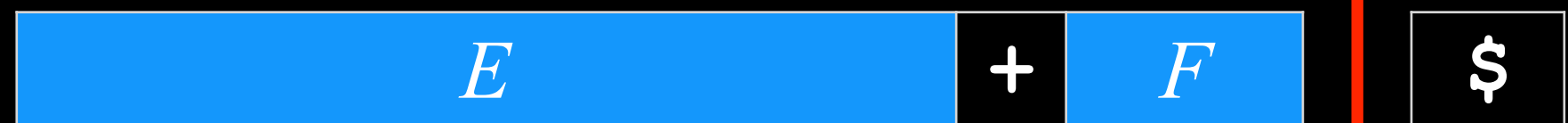
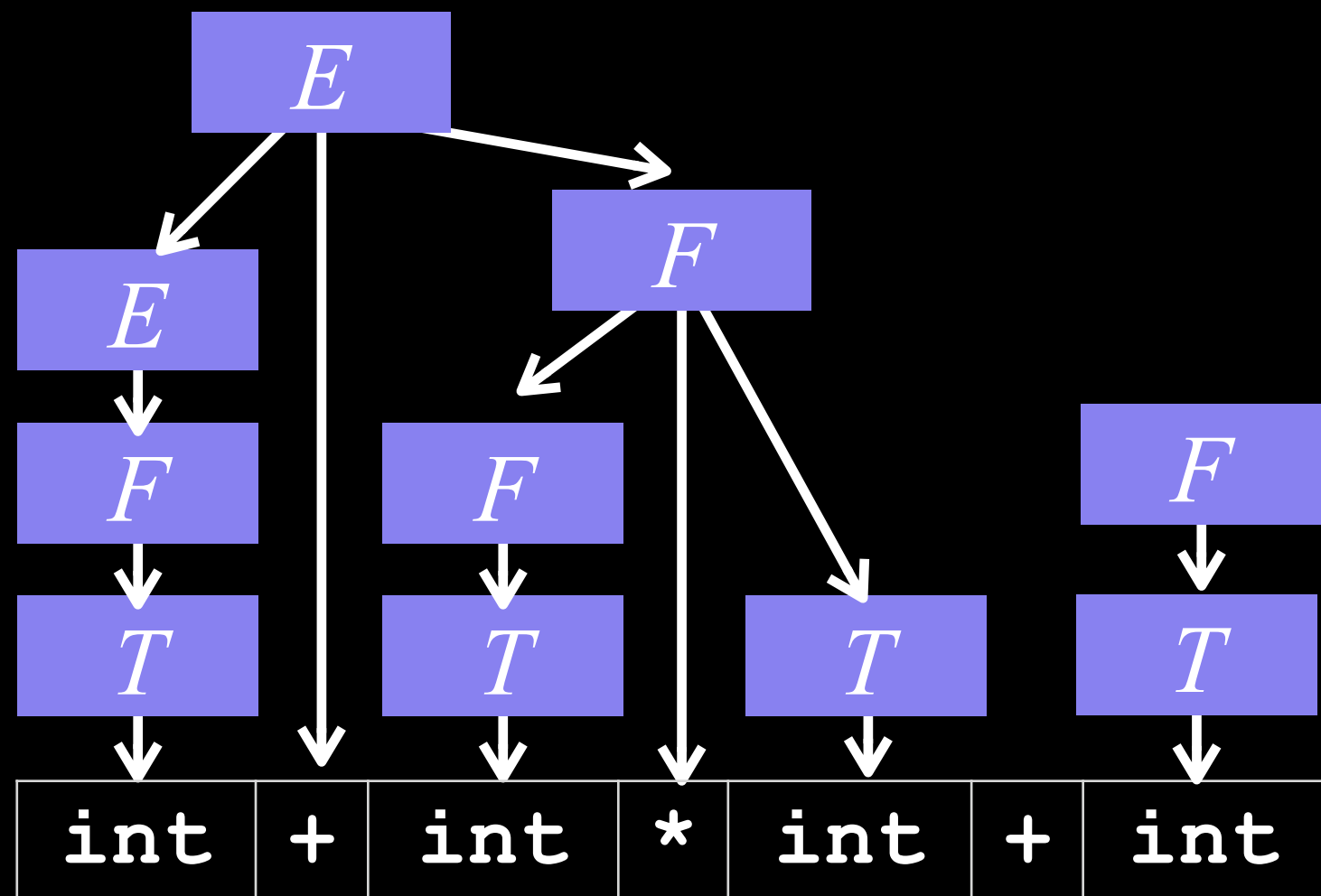
$$\begin{aligned}
 S &\rightarrow E\$ \\
 E &\rightarrow F \\
 E &\rightarrow E + F \\
 F &\rightarrow F * T \\
 F &\rightarrow T \\
 T &\rightarrow \text{int} \\
 T &\rightarrow (E)
 \end{aligned}$$


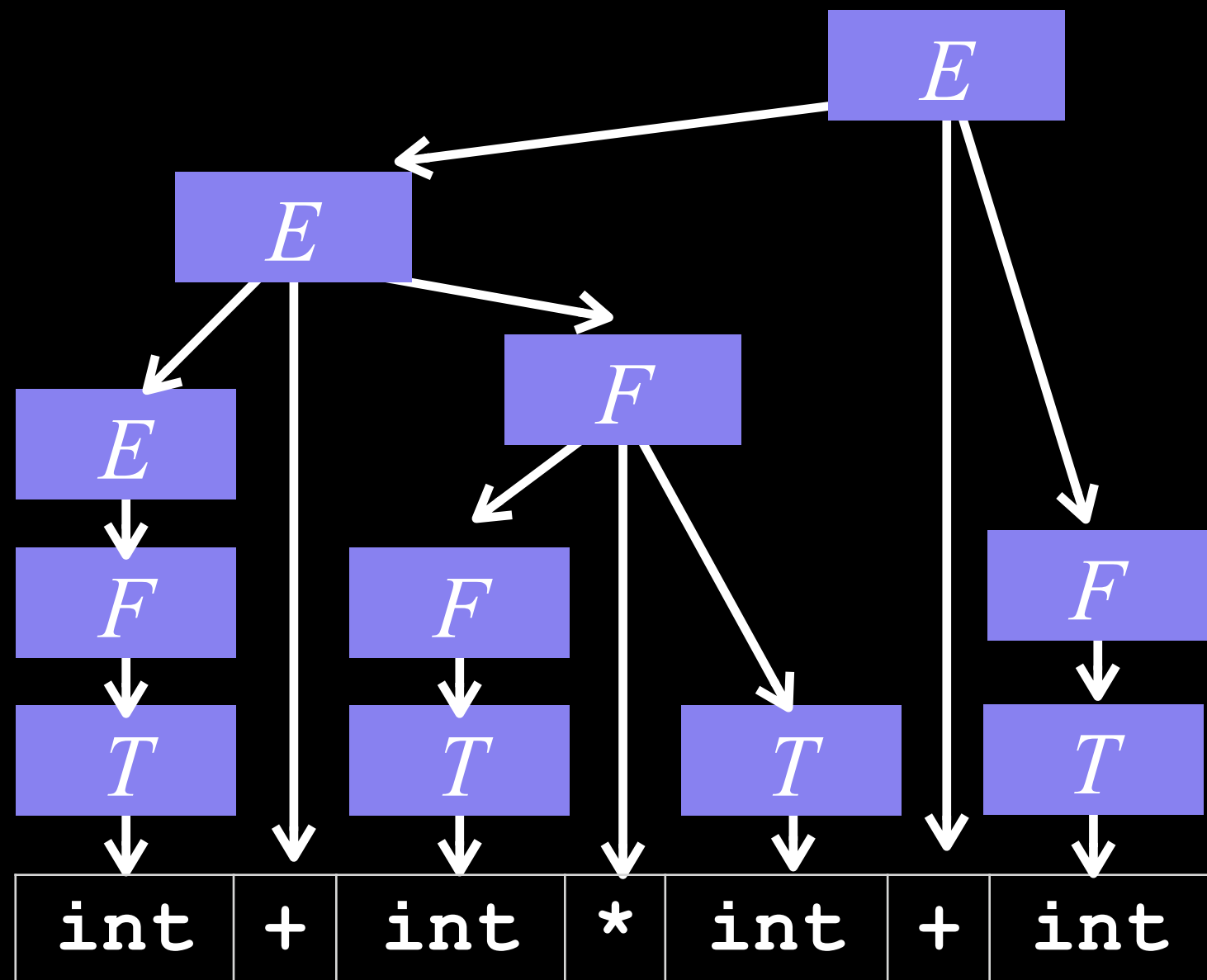
$$\begin{aligned}
 S &\rightarrow E\$ \\
 E &\rightarrow F \\
 E &\rightarrow E + F \\
 F &\rightarrow F * T \\
 F &\rightarrow T \\
 T &\rightarrow \text{int} \\
 T &\rightarrow (E)
 \end{aligned}$$


$$\begin{aligned}
 S &\rightarrow E\$ \\
 E &\rightarrow F \\
 E &\rightarrow E + F \\
 F &\rightarrow F * T \\
 F &\rightarrow T \\
 T &\rightarrow \text{int} \\
 T &\rightarrow (E)
 \end{aligned}$$


E	$+$	T
-----	-----	-----

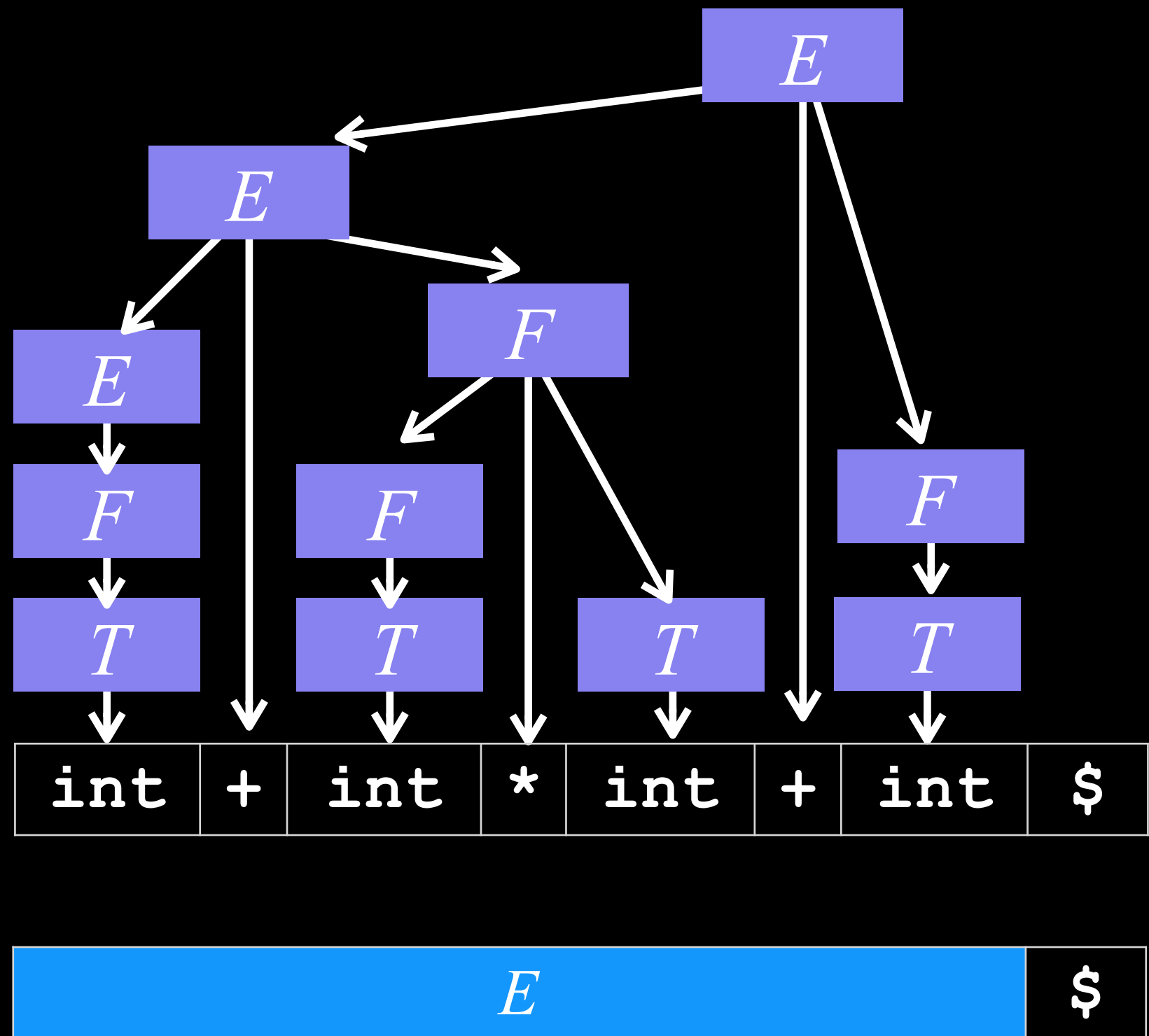
$\$$

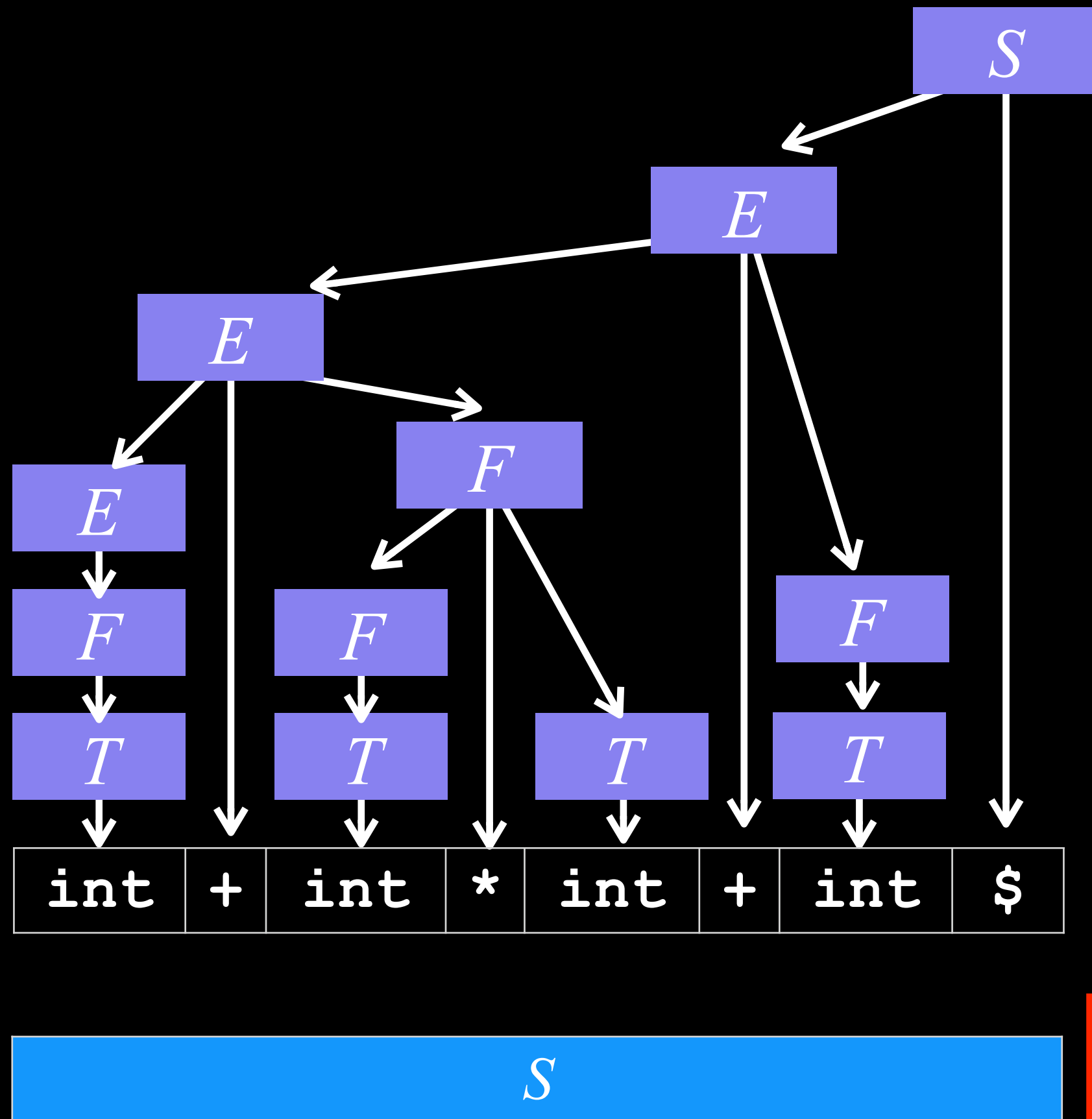
$$\begin{aligned}
 S &\rightarrow E\$ \\
 E &\rightarrow F \\
 E &\rightarrow E + F \\
 F &\rightarrow F * T \\
 F &\rightarrow T \\
 T &\rightarrow \text{int} \\
 T &\rightarrow (E)
 \end{aligned}$$


$$\begin{aligned}
 S &\rightarrow E\$ \\
 E &\rightarrow F \\
 E &\rightarrow E + F \\
 F &\rightarrow F * T \\
 F &\rightarrow T \\
 T &\rightarrow \text{int} \\
 T &\rightarrow (E)
 \end{aligned}$$


E

\$

$$\begin{aligned}
 S &\rightarrow E\$ \\
 E &\rightarrow F \\
 E &\rightarrow E + F \\
 F &\rightarrow F * T \\
 F &\rightarrow T \\
 T &\rightarrow \text{int} \\
 T &\rightarrow (E)
 \end{aligned}$$


$$\begin{aligned}
 S &\rightarrow E\$ \\
 E &\rightarrow F \\
 E &\rightarrow E + F \\
 F &\rightarrow F * T \\
 F &\rightarrow T \\
 T &\rightarrow \text{int} \\
 T &\rightarrow (E)
 \end{aligned}$$


Observe que todas as reduções
foram na área mais à direita da
substring da esquerda

Como implementar
shift-reduce parsing?

Implementando

- Toda a atividade de redução é na parte mais à direita da substring da esquerda
- Intuição: representar esta substring como uma pilha
- Shift: empilhar um token na pilha
- Reduce: desempilhar símbolos da pilha e empilhar o não-terminal apropriado
 - Para $A \rightarrow w$, então desempilha $|w|$ símbolos e empilha A

Handles

- De que forma o analisador escolhe qual ação deve tomar?
- Escolha errada pode levar a problemas, como visto, mesmo em entradas válidas
- Em um passo de derivação mais à direita $\mathbf{uAv} \rightarrow \mathbf{uwv}$, a cadeia \mathbf{uw} é um handle de \mathbf{uwv}
- Reduzimos quando o conteúdo da pilha for um handle

Identificando Handles

- Não existe um algoritmo geral para reconhecimento de handles no topo da pilha
- Podemos utilizar heurísticas que funcionam para classes de gramáticas
- De forma geral, reconhece handles olhando pra pilha e o próximo token de entrada (*lookahead*)

Prefixos Viáveis

- Existe um algoritmo simples para reconhecer prefixos de *handles* (prefixos viáveis)
- Enquanto o analisador tem um prefixo viável na pilha, está ok
- Isto leva a um algoritmo de parsing básico:
 - **shift**, se após a ação, a pilha continuar tendo um prefixo viável
 - **reduce**, se encontrou um handle
 - **error**, se entrada é mal-formada

Prefixos Viáveis

- O conjunto de prefixos viáveis de uma gramática é uma ***linguagem regular***
- Isto quer dizer que podemos utilizar autômatos finitos (determinísticos) para determinar se o conteúdo da pilha corresponde a um prefixo viável (ou não)

Itens LR(0)

- Para construir um autômato que reconhece prefixos viáveis, é utilizado o conceito de itens
- Um item LR(0) de uma gramática é uma produção da gramática com uma marca no lado direito (similar ao conceito de foco)

Exemplo

$$T \rightarrow (E)$$

$$T \rightarrow \bullet(E)$$

$$T \rightarrow (\bullet E)$$

$$T \rightarrow (E\bullet)$$

$$T \rightarrow (E)\bullet$$

itens com a marca no final são ***itens de redução***

Rastreando posição nas produções com itens

$S \rightarrow E\$$

$E \rightarrow E + T$

$E \rightarrow T$

$T \rightarrow \text{int}$

$T \rightarrow (E)$



int	+	(int	+	int)	\$
-----	---	---	-----	---	-----	---	----

$S \rightarrow \bullet E\$$

$S \rightarrow E\$$

$E \rightarrow E + T$

$E \rightarrow T$

$T \rightarrow \text{int}$

$T \rightarrow (E)$

int

+

(

int

+

int

)

\$

$S \rightarrow \bullet E\$$
$E \rightarrow \bullet E+T$

$S \rightarrow E\$$

$E \rightarrow E + T$

$E \rightarrow T$

$T \rightarrow \text{int}$

$T \rightarrow (E)$

int	+	(int	+	int)	\$
-----	---	---	-----	---	-----	---	----


$S \rightarrow \bullet E \$$
$E \rightarrow \bullet E + T$
$E \rightarrow \bullet T$

$S \rightarrow E \$$
 $E \rightarrow E + T$
 $E \rightarrow T$
 $T \rightarrow \text{int}$
 $T \rightarrow (E)$

int	+	(int	+	int)	\$
-----	---	---	-----	---	-----	---	----

$S \rightarrow \bullet E\$$
$E \rightarrow \bullet E + T$
$E \rightarrow \bullet T$
$T \rightarrow \bullet \text{int}$

$S \rightarrow E\$$
 $E \rightarrow E + T$
 $E \rightarrow T$
 $T \rightarrow \text{int}$
 $T \rightarrow (E)$



int	+	(int	+	int)	\$
-----	---	---	-----	---	-----	---	----

$S \rightarrow \bullet E \$$
$E \rightarrow \bullet E + T$
$E \rightarrow \bullet T$
$T \rightarrow \bullet \text{int}$

$S \rightarrow E \$$
 $E \rightarrow E + T$
 $E \rightarrow T$
 $T \rightarrow \text{int}$
 $T \rightarrow (E)$

int

+ (int + int) \$

$S \rightarrow \bullet E\$$
$E \rightarrow \bullet E + T$
$E \rightarrow \bullet T$
$T \rightarrow \text{int} \bullet$

$S \rightarrow E\$$

$E \rightarrow E + T$

$E \rightarrow T$

$T \rightarrow \text{int}$

$T \rightarrow (E)$

int

+ (int + int) \$

$S \rightarrow \bullet E \$$
$E \rightarrow \bullet E + T$
$E \rightarrow \bullet T$
$T \rightarrow \text{int} \bullet$

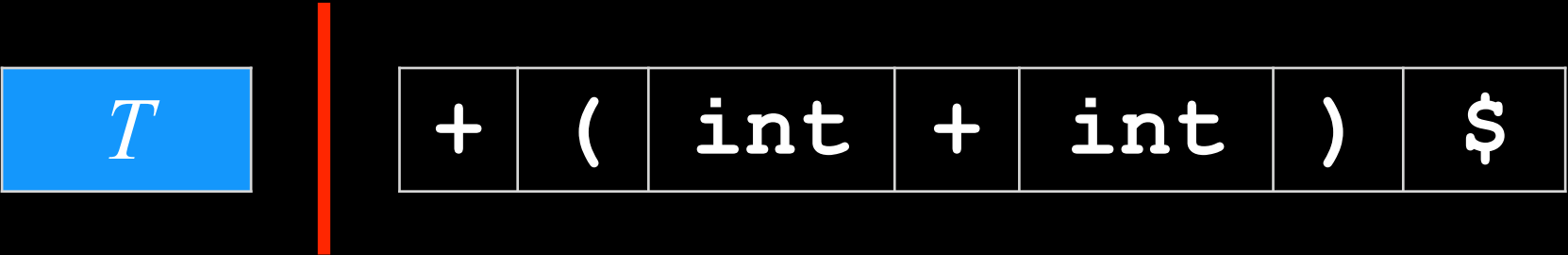
$$S \rightarrow E \$$$

$$E \rightarrow E + T$$

$$E \rightarrow T$$

$$T \rightarrow \text{int}$$

$$T \rightarrow (E)$$



$S \rightarrow \bullet E \$$
$E \rightarrow \bullet E + T$
$E \rightarrow T \bullet$

$S \rightarrow E \$$
 $E \rightarrow E + T$
 $E \rightarrow T$
 $T \rightarrow \text{int}$
 $T \rightarrow (E)$

T

+	(int	+	int)	\$
---	---	-----	---	-----	---	----

$S \rightarrow \bullet E \$$
$E \rightarrow \bullet E + T$
$E \rightarrow T \bullet$

$S \rightarrow E \$$
 $E \rightarrow E + T$
 $E \rightarrow T$
 $T \rightarrow \text{int}$
 $T \rightarrow (E)$

E

+	(int	+	int)	\$
---	---	-----	---	-----	---	----

$S \rightarrow \bullet E \$$
$E \rightarrow E \bullet + T$

...

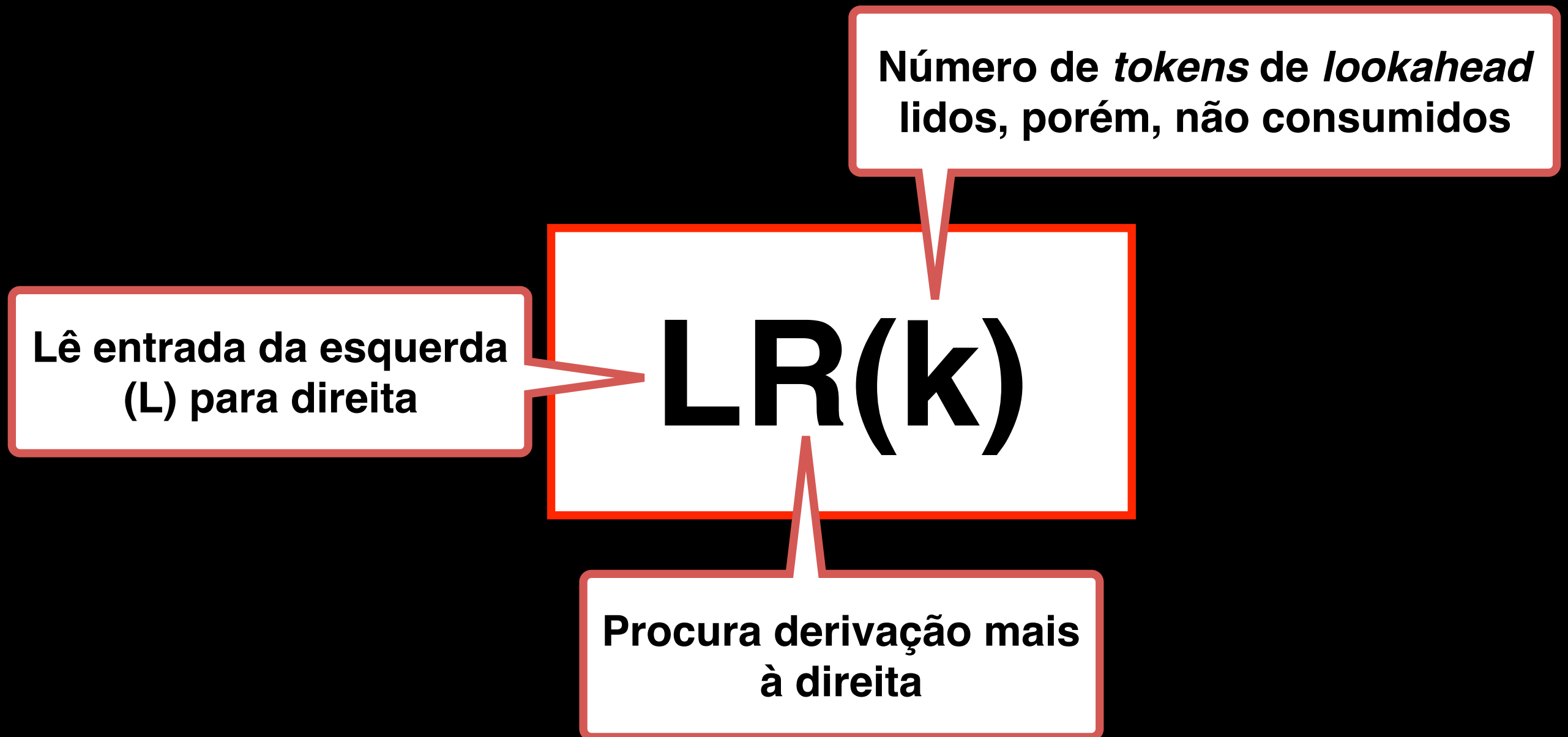
continuado no quadro...

$S \rightarrow E \$$
 $E \rightarrow E + T$
 $E \rightarrow T$
 $T \rightarrow \text{int}$
 $T \rightarrow (E)$

E

+	(int	+	int)	\$
---	---	-----	---	-----	---	----

Classificação de Parsers



Gramáticas LR(0)

- O parsing pode ser feito olhando apenas o conteúdo da pilha
- Não é necessário lookahead para decidir entre ações de shift e reduce
- Classe razoavelmente fraca de gramáticas
- Algoritmo de construção de tabelas é útil como introdução a algoritmos LR(1)...

Exemplo

$$_0 S' \rightarrow S\$$$

$$_1 S \rightarrow (L)$$

$$_2 S \rightarrow \mathbf{x}$$

$$_3 L \rightarrow S$$

$$_4 L \rightarrow L, S$$

$$_0 S' \rightarrow SS$$

$$_1 S \rightarrow (L)$$

$$_3 L \rightarrow S$$

$$_2 S \rightarrow \mathbf{x}$$

$$_4 L \rightarrow L, S$$

1

$$S' \rightarrow \bullet SS$$

$$_0 S' \longrightarrow S\$$$

$$_1 S \longrightarrow (L)$$

$$_3 L \longrightarrow S$$

$$_2 S \longrightarrow \mathbf{x}$$

$$_4 L \longrightarrow L, S$$

1

$$\begin{array}{l} S' \longrightarrow \bullet S\$ \\ S \longrightarrow \bullet (L) \\ S \longrightarrow \bullet \mathbf{x} \end{array}$$

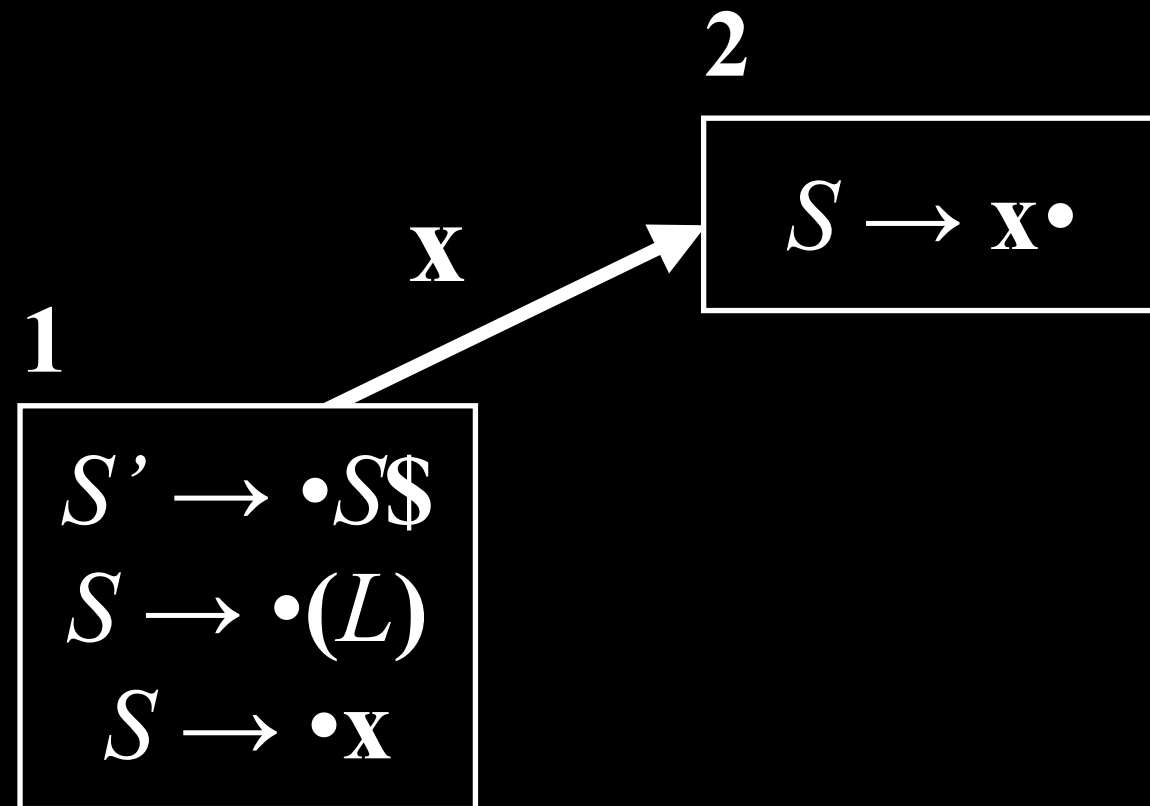
$$_0 S' \rightarrow S\$$$

$$_1 S \rightarrow (L)$$

$$_3 L \rightarrow S$$

$$_2 S \rightarrow \mathbf{x}$$

$$_4 L \rightarrow L, S$$



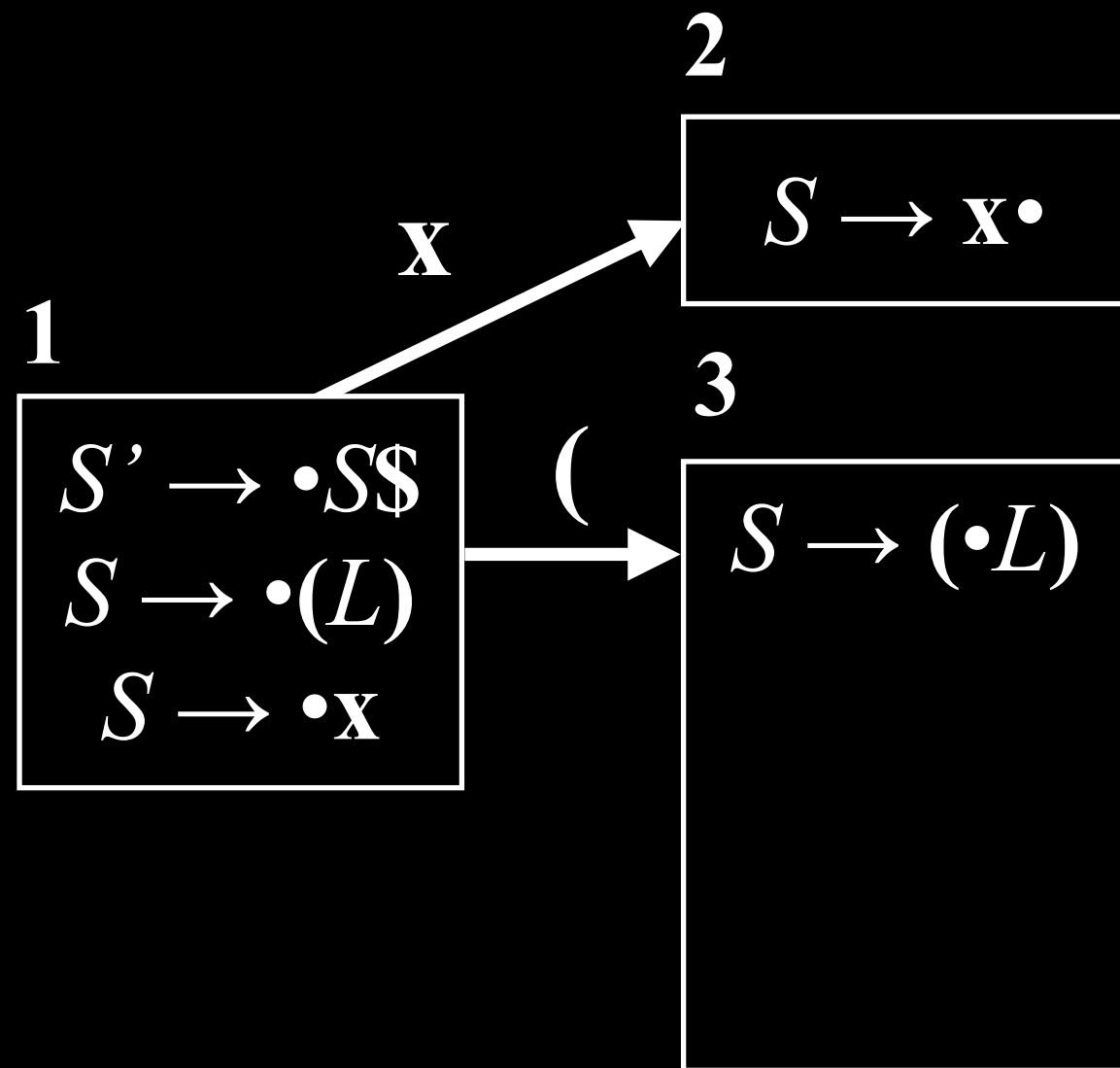
$$_0 S' \rightarrow SS$$

$$_1 S \rightarrow (L)$$

$$_3 L \rightarrow S$$

$$_2 S \rightarrow \mathbf{x}$$

$$_4 L \rightarrow L, S$$



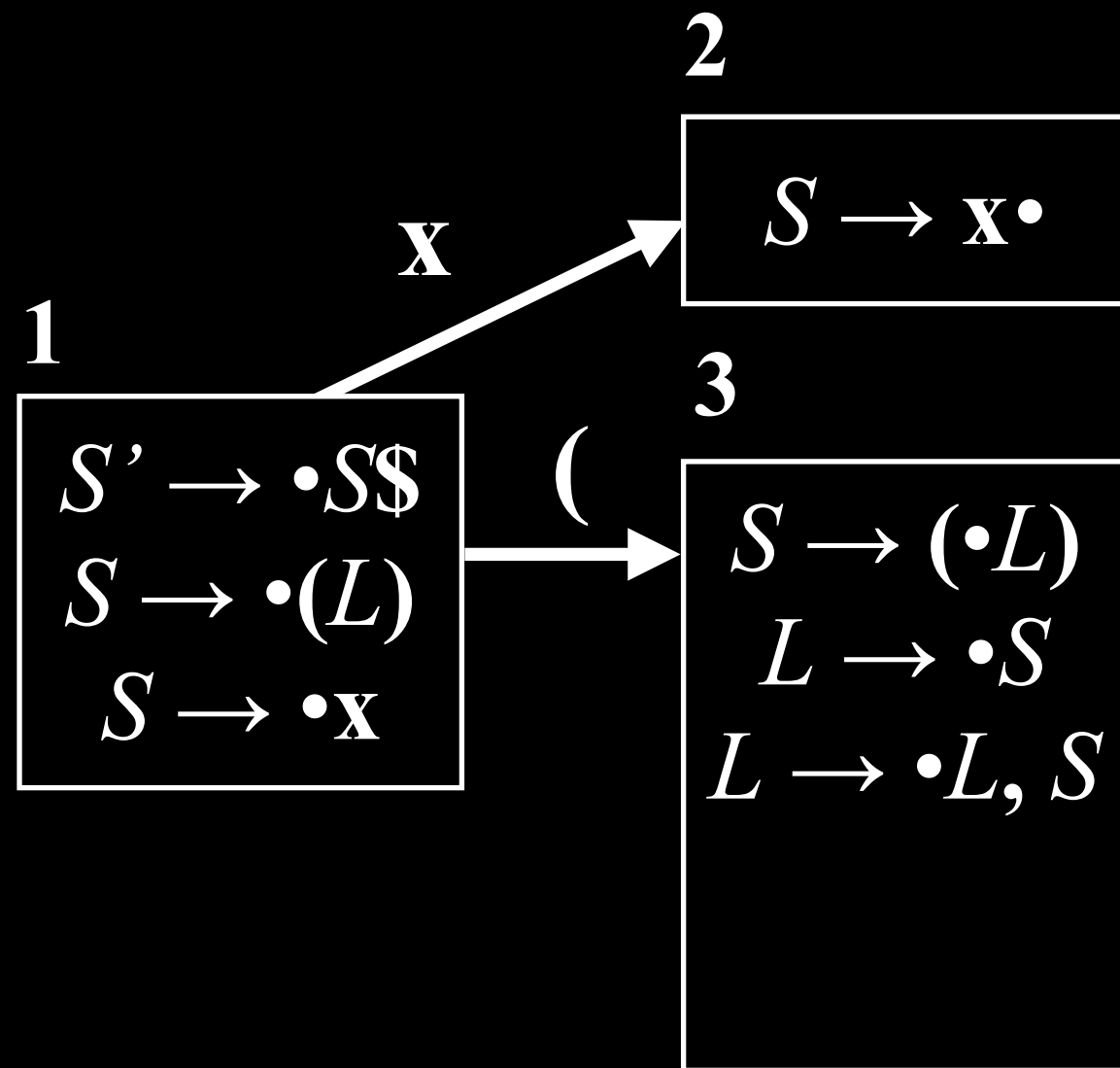
$$_0 S' \rightarrow SS$$

$$_1 S \rightarrow (L)$$

$$_3 L \rightarrow S$$

$$_2 S \rightarrow \mathbf{x}$$

$$_4 L \rightarrow L, S$$



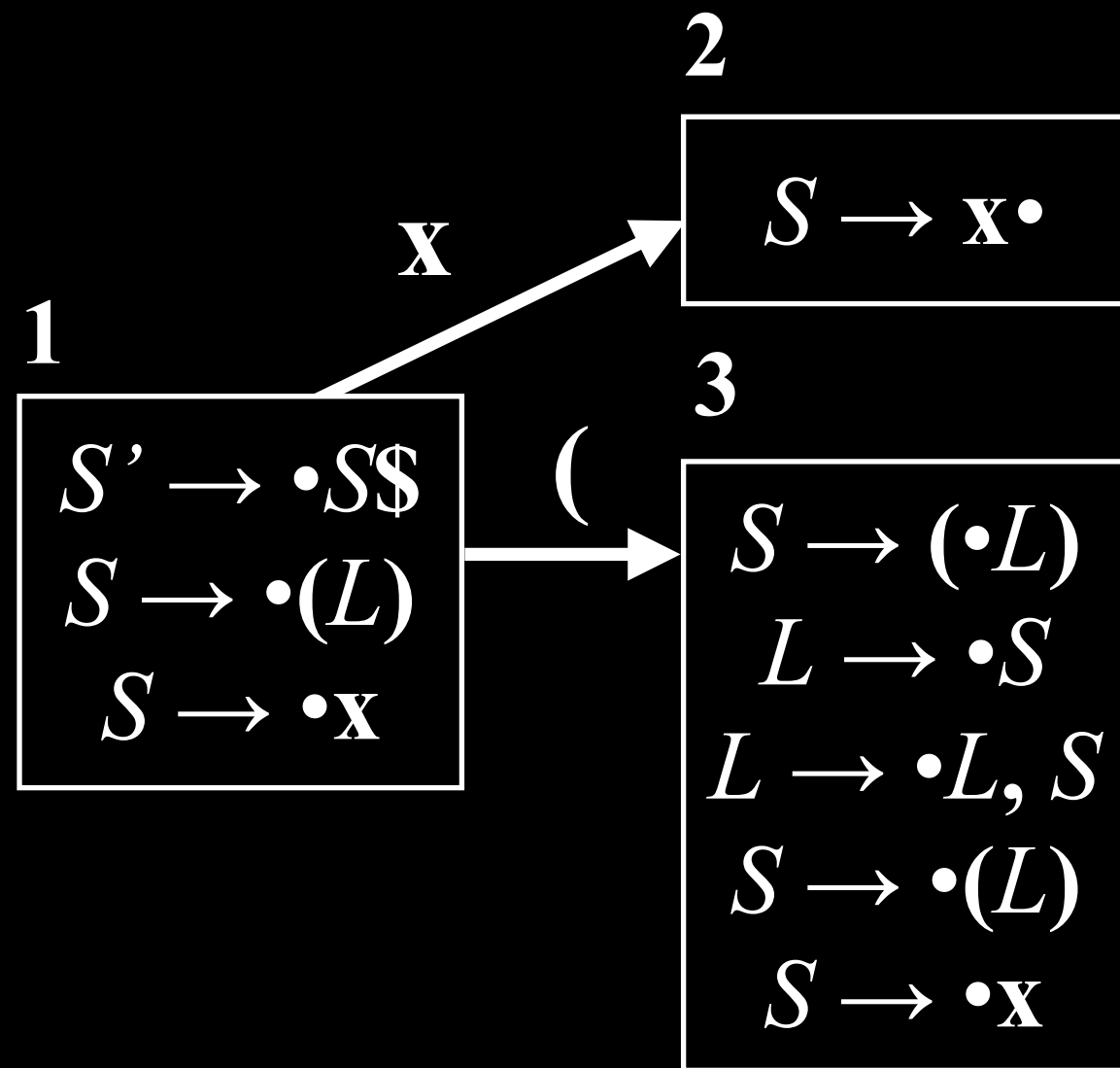
$$_0 S' \rightarrow SS$$

$$_1 S \rightarrow (L)$$

$$_3 L \rightarrow S$$

$$_2 S \rightarrow \mathbf{x}$$

$$_4 L \rightarrow L, S$$



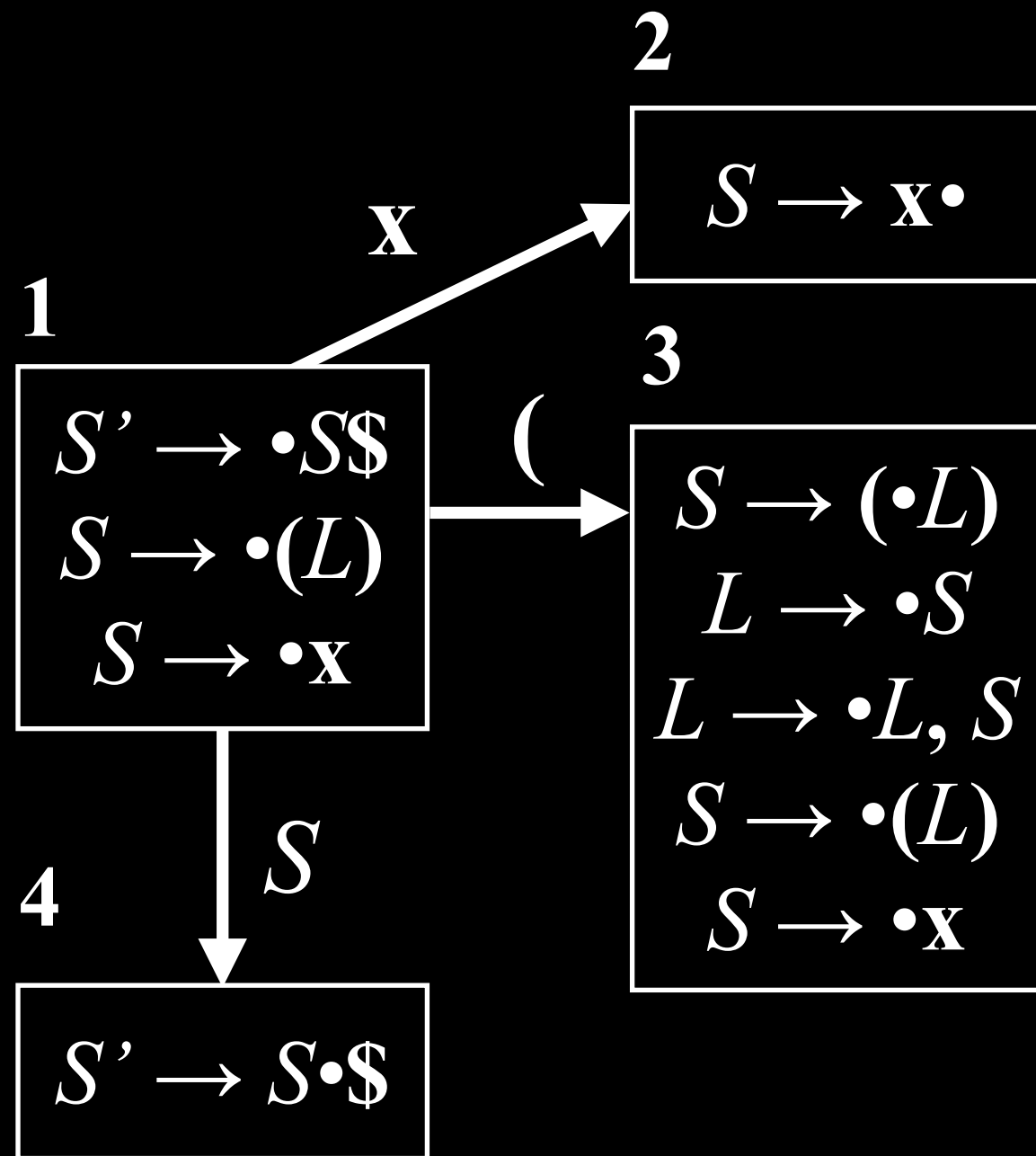
$$_0 S' \rightarrow SS$$

$$_1 S \rightarrow (L)$$

$$_3 L \rightarrow S$$

$$_2 S \rightarrow \mathbf{x}$$

$$_4 L \rightarrow L, S$$



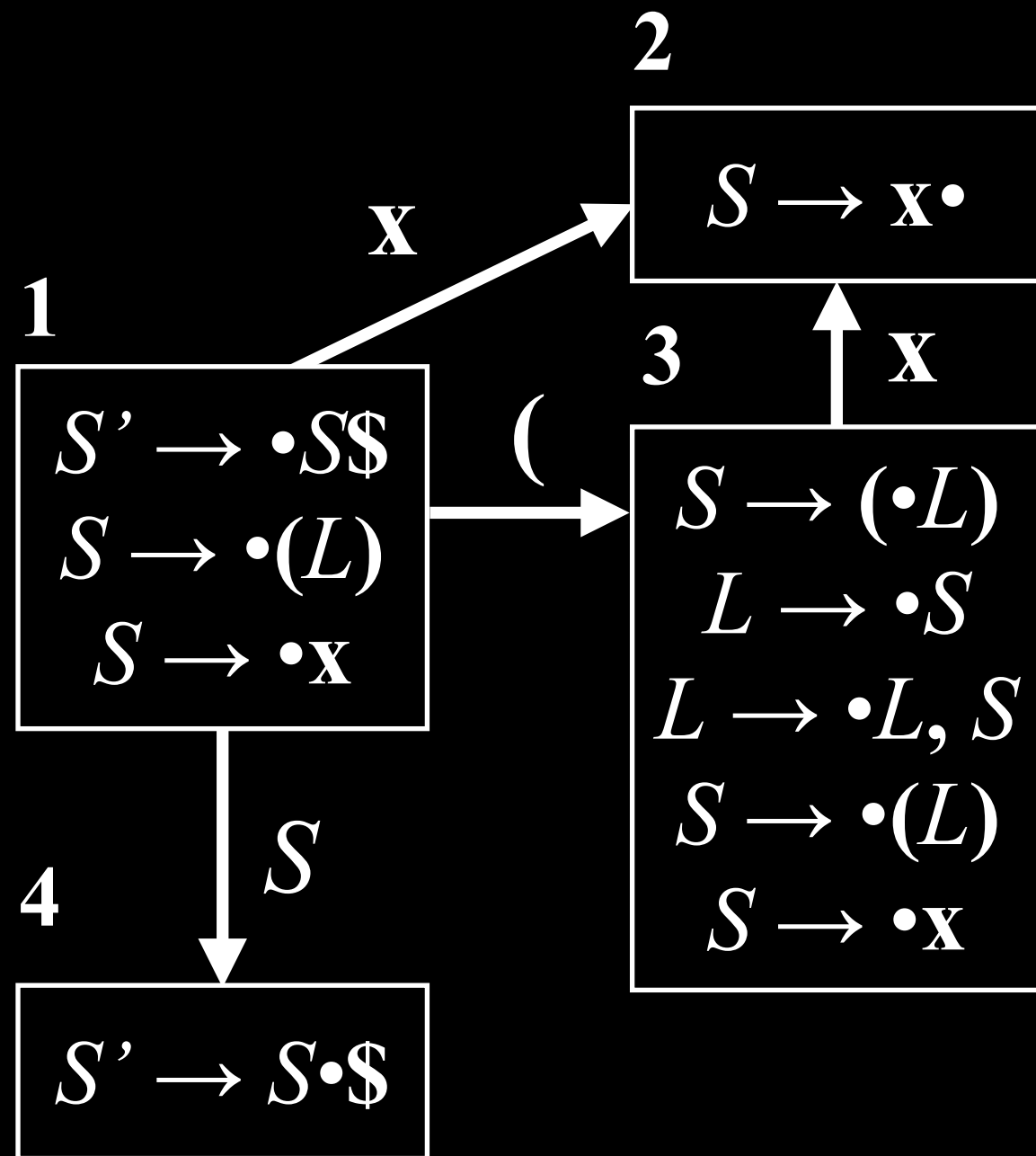
$$_0 S' \rightarrow SS$$

$$_1 S \rightarrow (L)$$

$$_3 L \rightarrow S$$

$$_2 S \rightarrow x$$

$$_4 L \rightarrow L, S$$



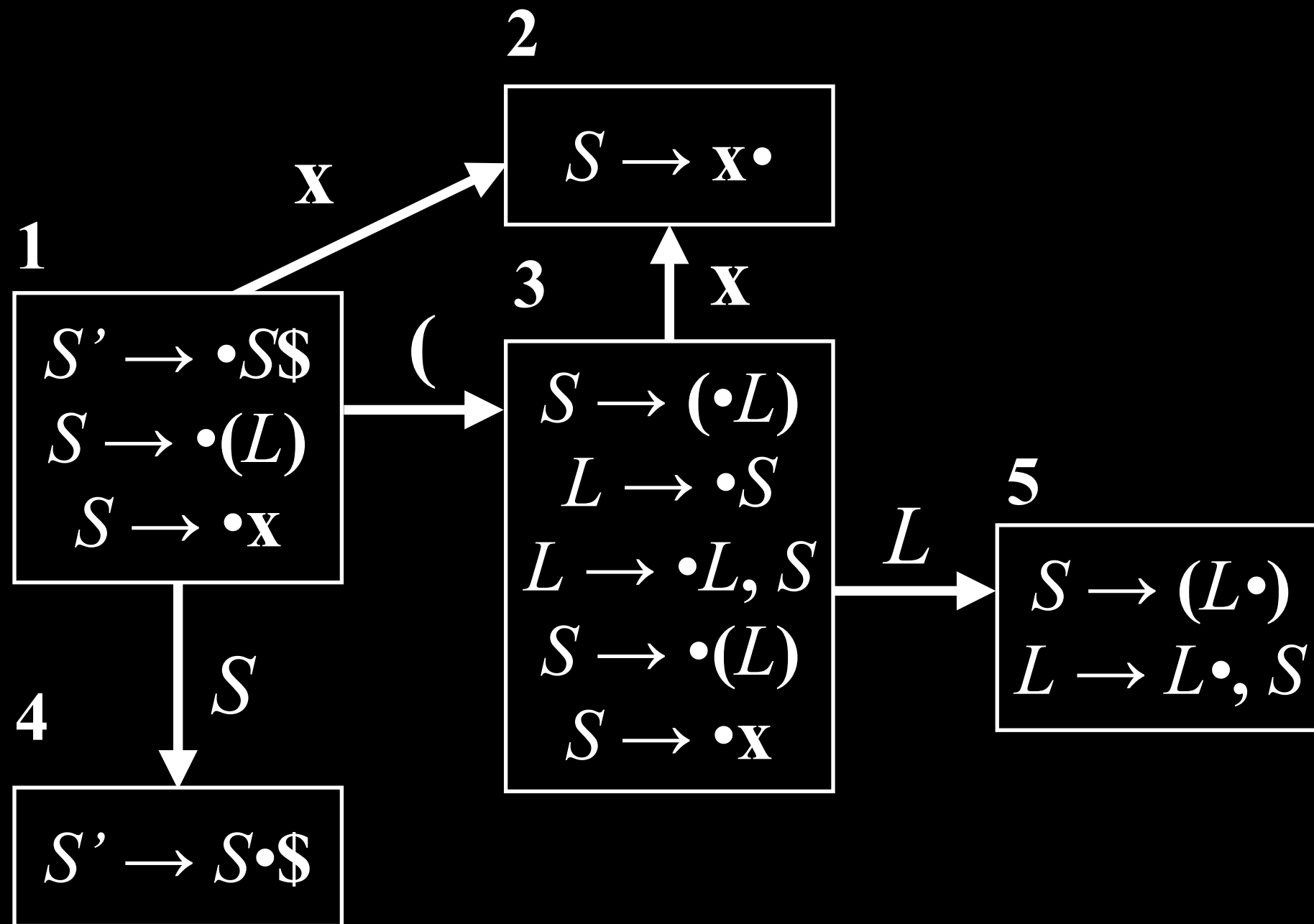
$$_0 S' \rightarrow SS$$

$$_1 S \rightarrow (L)$$

$$_3 L \rightarrow S$$

$$_2 S \rightarrow \mathbf{x}$$

$$_4 L \rightarrow L, S$$



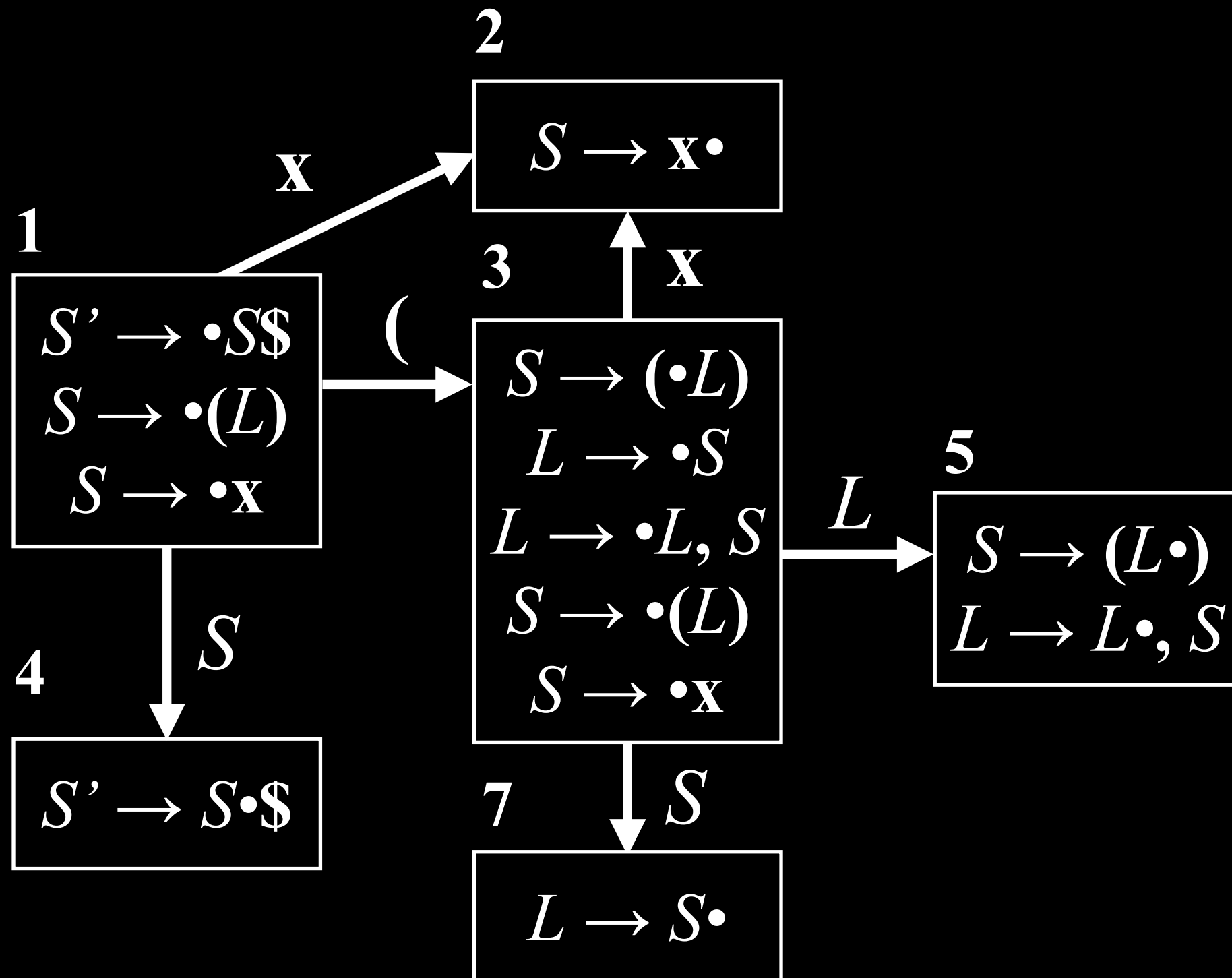
$$_0 S' \rightarrow SS$$

$$_1 S \rightarrow (L)$$

$$_3 L \rightarrow S$$

$$_2 S \rightarrow x$$

$$_4 L \rightarrow L, S$$



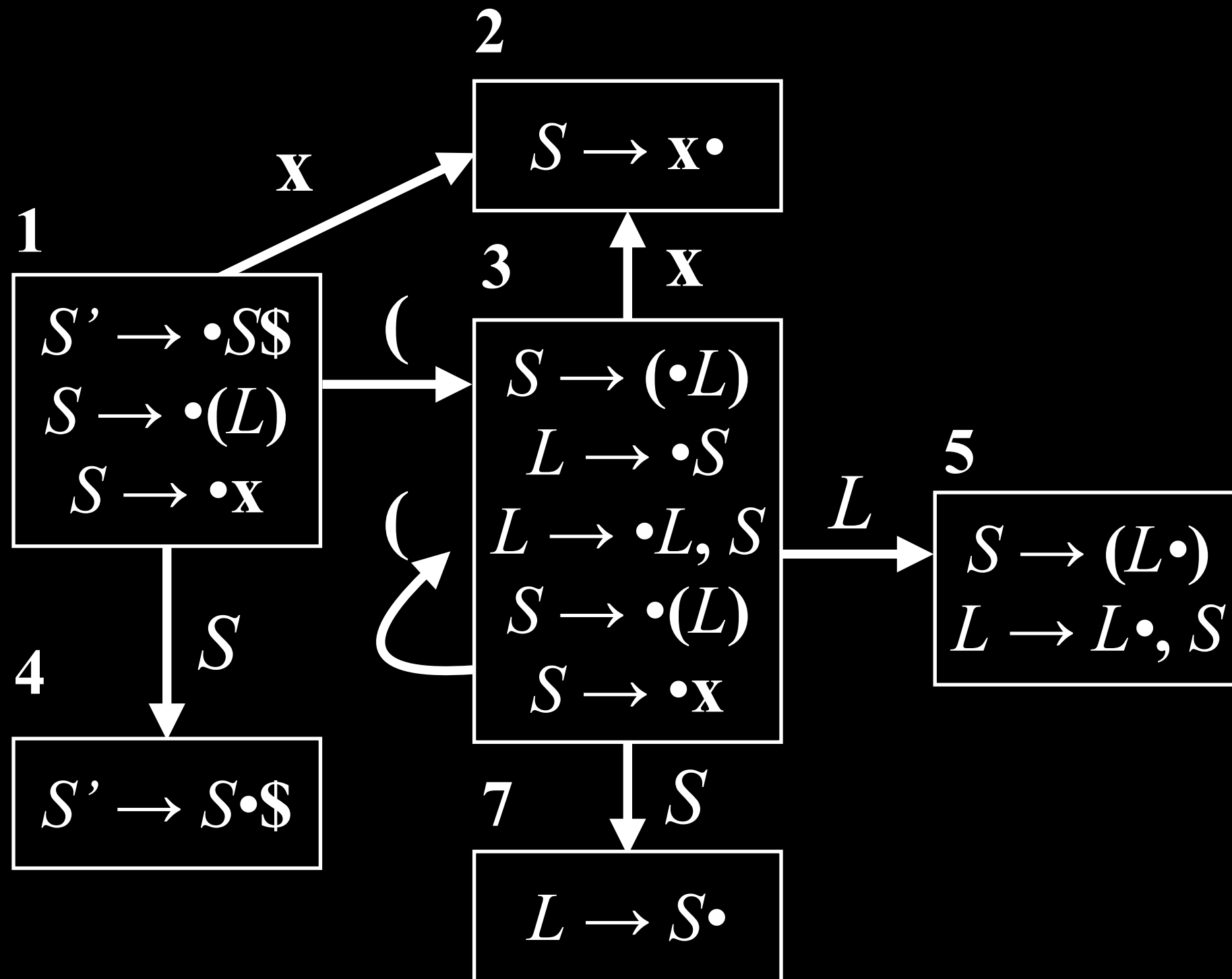
$$_0 S' \rightarrow SS$$

$$_1 S \rightarrow (L)$$

$$_3 L \rightarrow S$$

$$_2 S \rightarrow \mathbf{x}$$

$$_4 L \rightarrow L, S$$



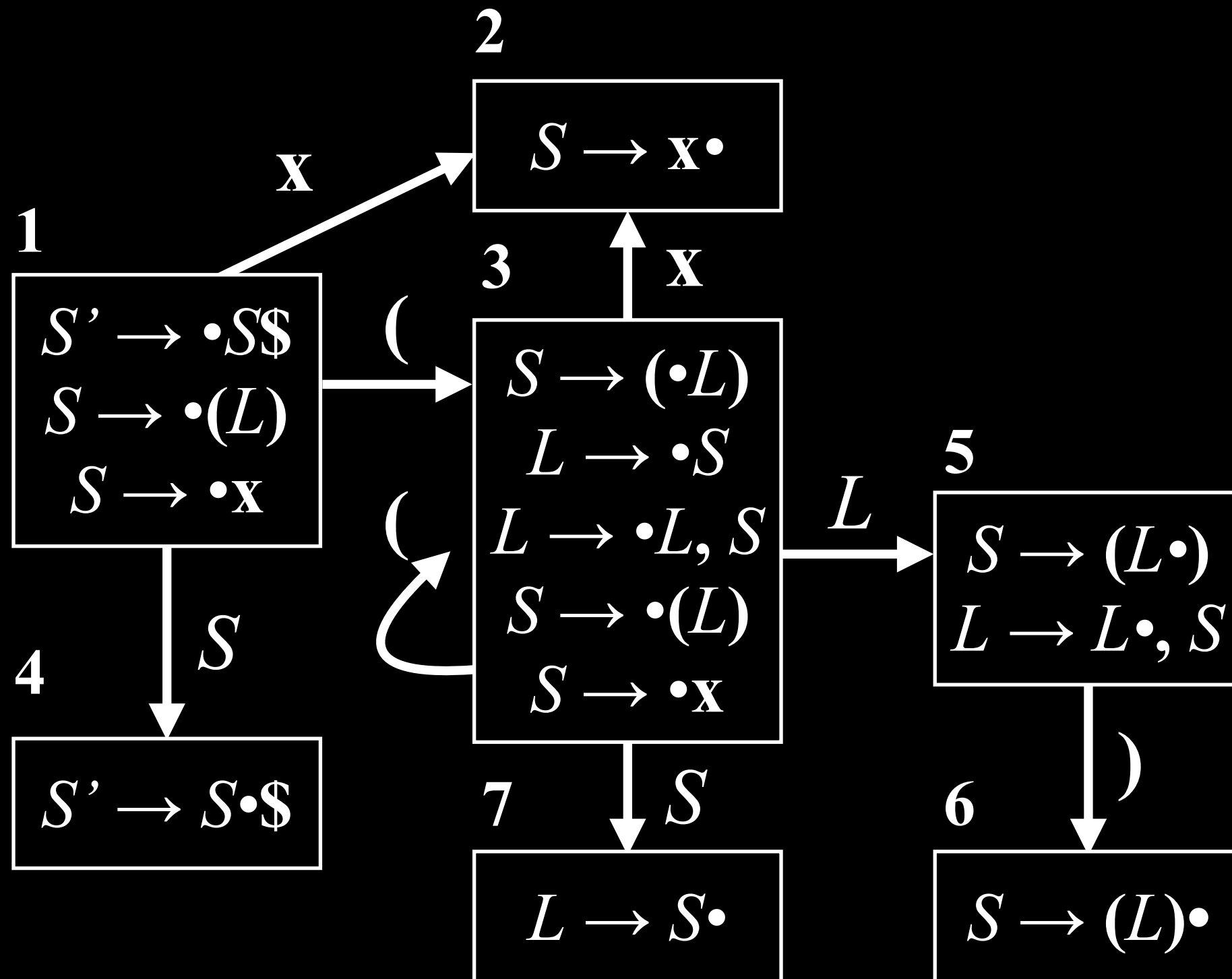
$$_0 S' \rightarrow SS$$

$$_1 S \rightarrow (L)$$

$$_3 L \rightarrow S$$

$$_2 S \rightarrow x$$

$$_4 L \rightarrow L, S$$



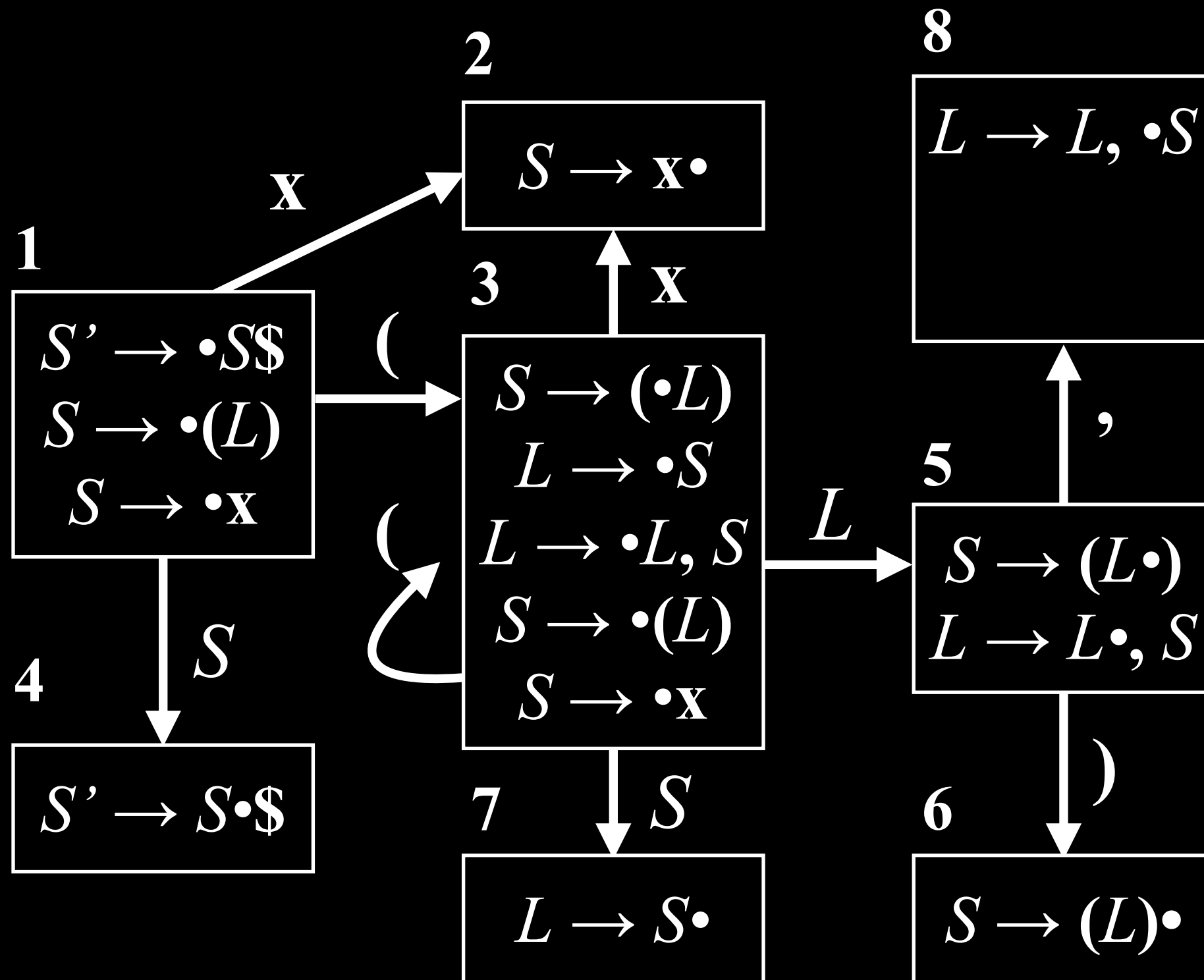
$$_0 S' \rightarrow SS$$

$$_1 S \rightarrow (L)$$

$$_3 L \rightarrow S$$

$$_2 S \rightarrow \mathbf{x}$$

$$_4 L \rightarrow L, S$$



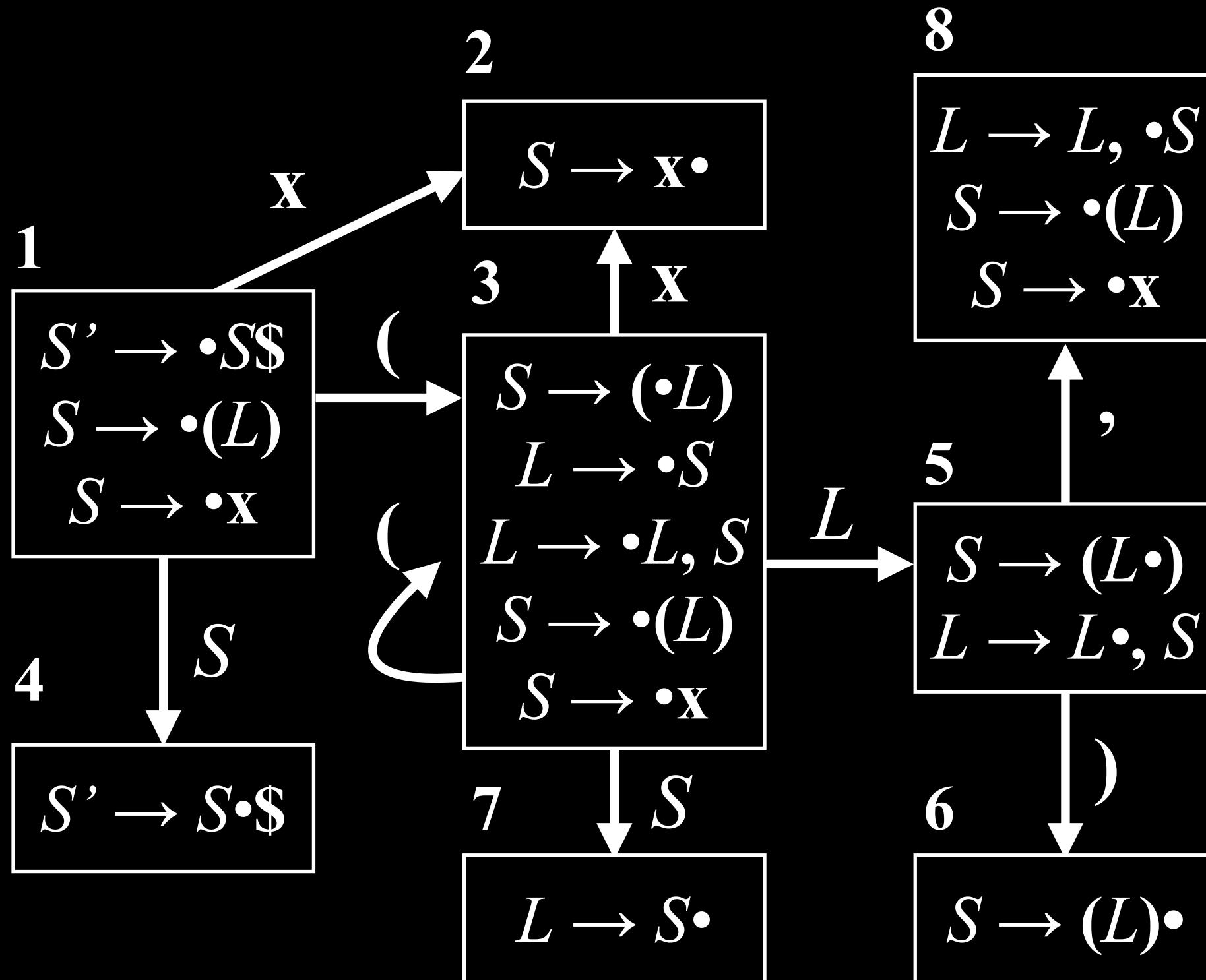
$$_0 S' \rightarrow SS$$

$$_1 S \rightarrow (L)$$

$$_3 L \rightarrow S$$

$$_2 S \rightarrow x$$

$$_4 L \rightarrow L, S$$



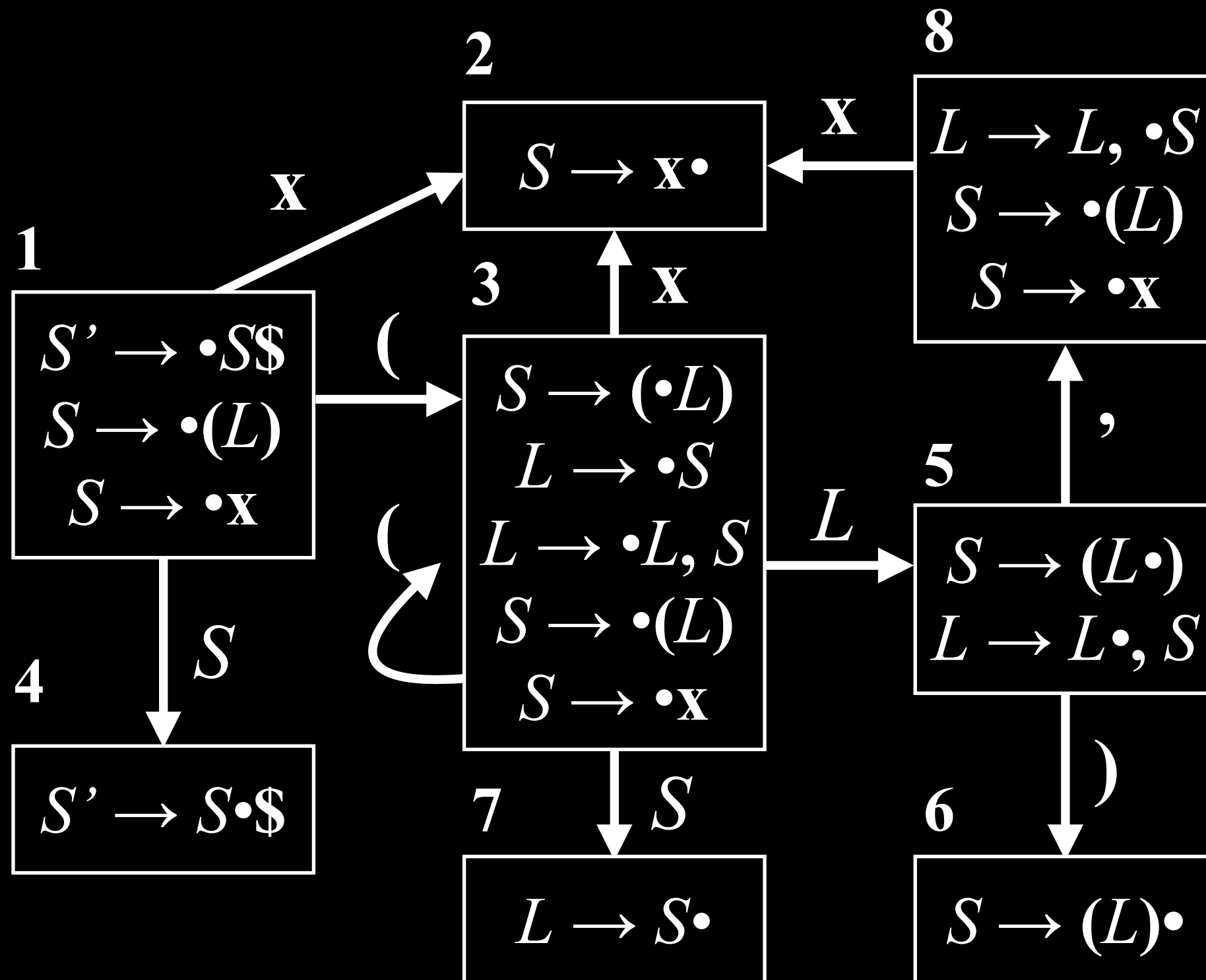
$$_0 S' \rightarrow SS$$

$$_1 S \rightarrow (L)$$

$$_3 L \rightarrow S$$

$$_2 S \rightarrow x$$

$$_4 L \rightarrow L, S$$



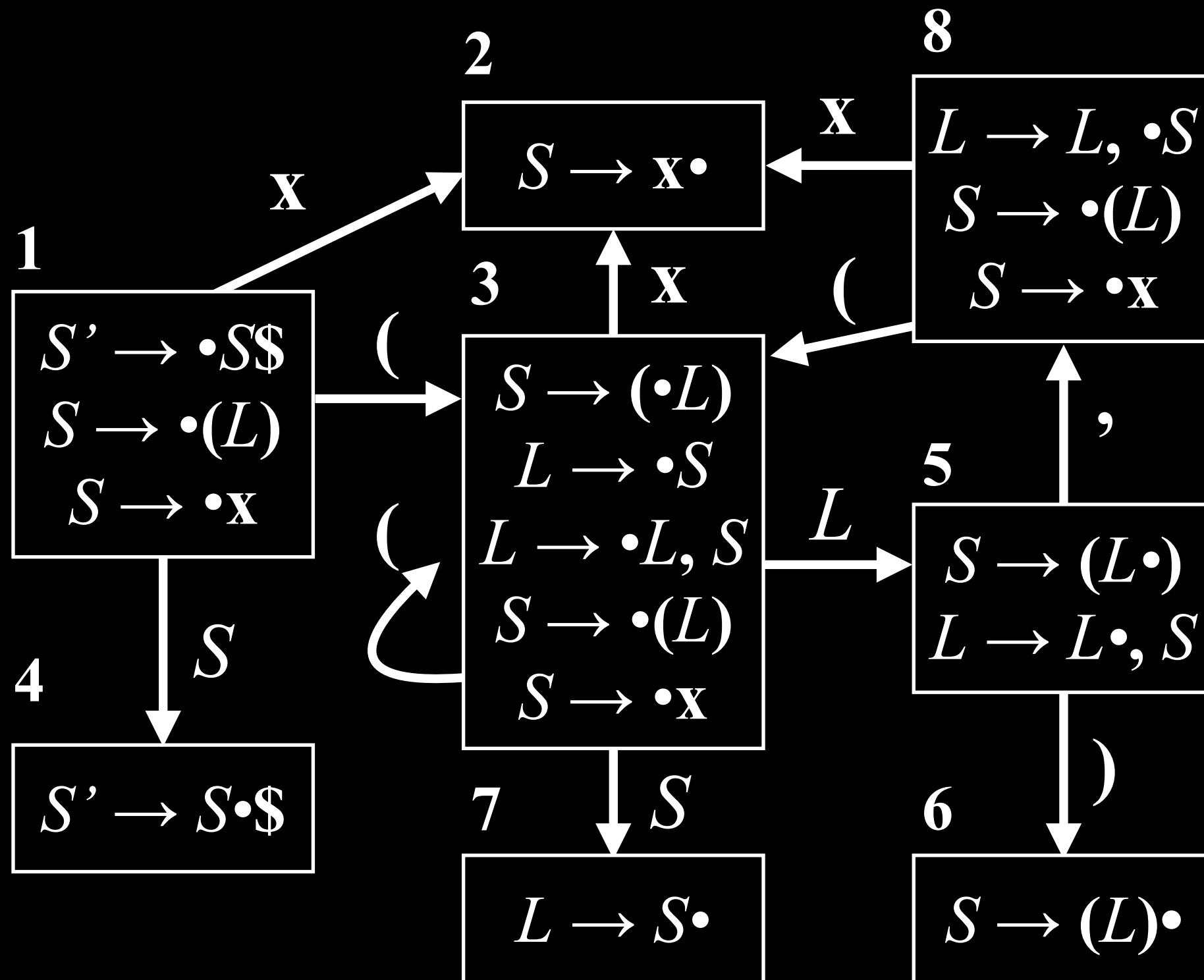
$$_0 S' \rightarrow SS$$

$$_1 S \rightarrow (L)$$

$$_3 L \rightarrow S$$

$$_2 S \rightarrow x$$

$$_4 L \rightarrow L, S$$



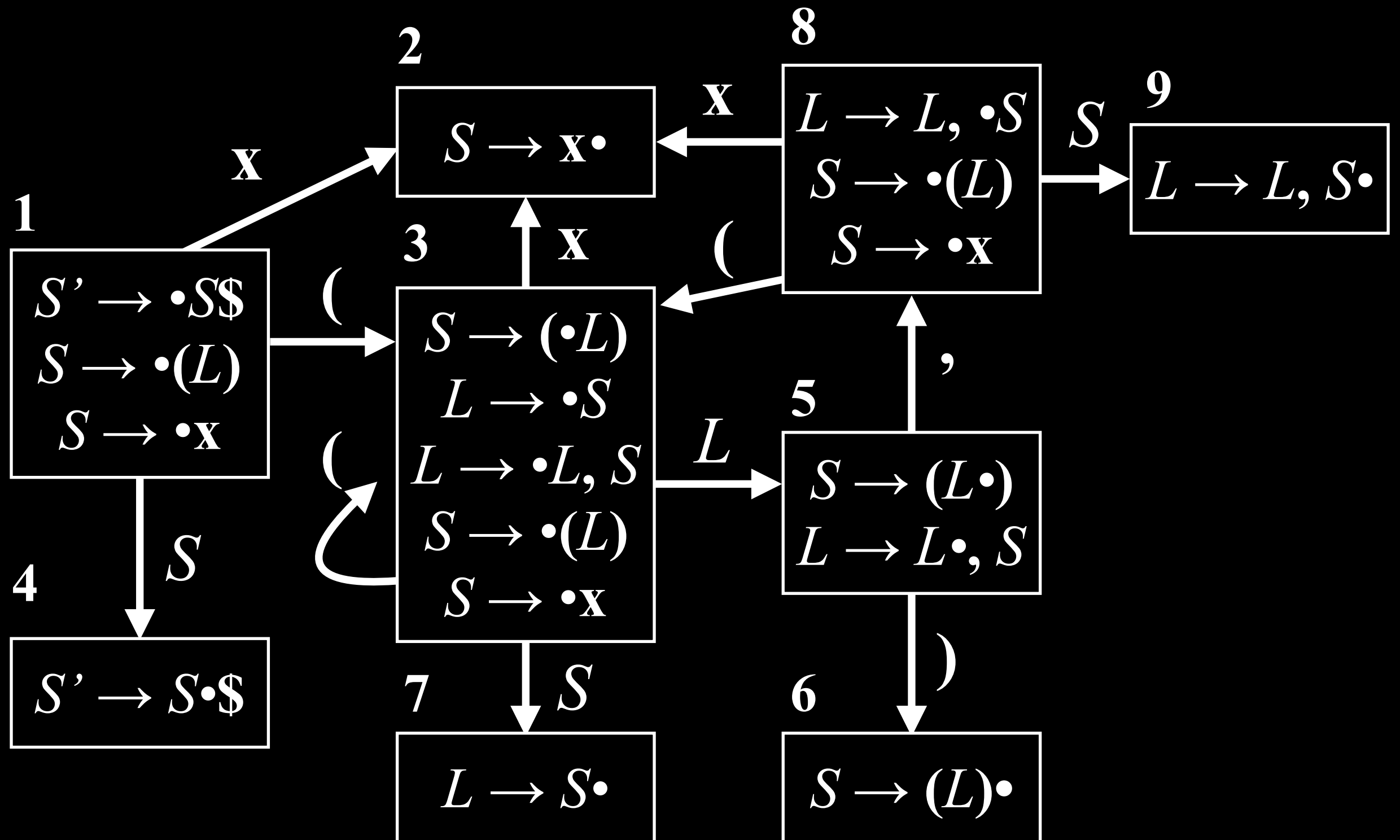
$$_0 S' \rightarrow SS$$

$$_1 S \rightarrow (L)$$

$$_3 L \rightarrow S$$

$$_2 S \rightarrow \mathbf{x}$$

$$_4 L \rightarrow L, S$$



Parsing Table

- ***sn***: shift para estado *n*
- ***gn***: goto para estado *n*
- ***rk***: reduce pela regra *k*
- ***a***: aceite a entrada
- ***erro***: entradas em branco

Construindo parsing table M

- Para cada transição entre estados I e J , disparadas por um símbolo X
 - se X for terminal $M[I,X] = \text{shift } J$
 - se X for não terminal $M[I,X] = \text{goto } J$
- Em estados com item $S \rightarrow \alpha \bullet$ (onde esta é a n -ésima produção), em todos os tokens Y : $M[I,Y] = \text{reduce } n$

$$_0 S' \rightarrow S\$$$

$$_1 S \rightarrow (L)$$

$$_3 L \rightarrow S$$

$$_2 S \rightarrow \mathbf{x}$$

$$_4 L \rightarrow L, S$$

	()	x	,	\$	S	L
1							
2							
3							
4							
5							
6							
7							
8							
9							

$$_0 S' \rightarrow S\$$$

$$_1 S \rightarrow (L)$$

$$_3 L \rightarrow S$$

$$_2 S \rightarrow \mathbf{x}$$

$$_4 L \rightarrow L, S$$

	()	x	,	\$	S	L
1	s3		s2			g4	
2	r2	r2	r2	r2	r2		
3	s3		s2			g7	g5
4					a		
5		s6		s8			
6	r1	r1	r1	r1	r1		
7	r3	r3	r3	r3	r3		
8	s3		s2			g9	
9	r4	r4	r4	r4	r4		

Parsing

- Ao invés de reescanear a pilha para cada token, pode lembrar o estado alcançado para cada elemento da pilha
- Algoritmo consiste em olhar o estado no topo da pilha e o símbolo de entrada para definir a ação

Parsing

- `shift(n)`: avance a entrada em um token, empilhe o estado `n`
- `reduce(k)`: desempilhe a quantidade de símbolos do lado direito da regra `k`; Seja `X` o não-terminal do lado esquerdo da regra `k`; No estado agora no topo da pilha, observe a entrada `X` para pegar o valor de `goto(n)`; Empilhe o estado `n`
- `accept`: encerra reportando sucesso
- `error`: encerra reportando falha

Construa a parsing table

$$_0 S' \rightarrow E\$$$

$$_1 E \rightarrow T + E$$

$$_2 E \rightarrow T$$

$$_3 T \rightarrow \mathbf{x}$$

$$_0 S' \longrightarrow ES\$$$

$$_1 E \longrightarrow T + E$$

$$_2 E \longrightarrow T$$

$$_3 T \longrightarrow \mathbf{x}$$

1

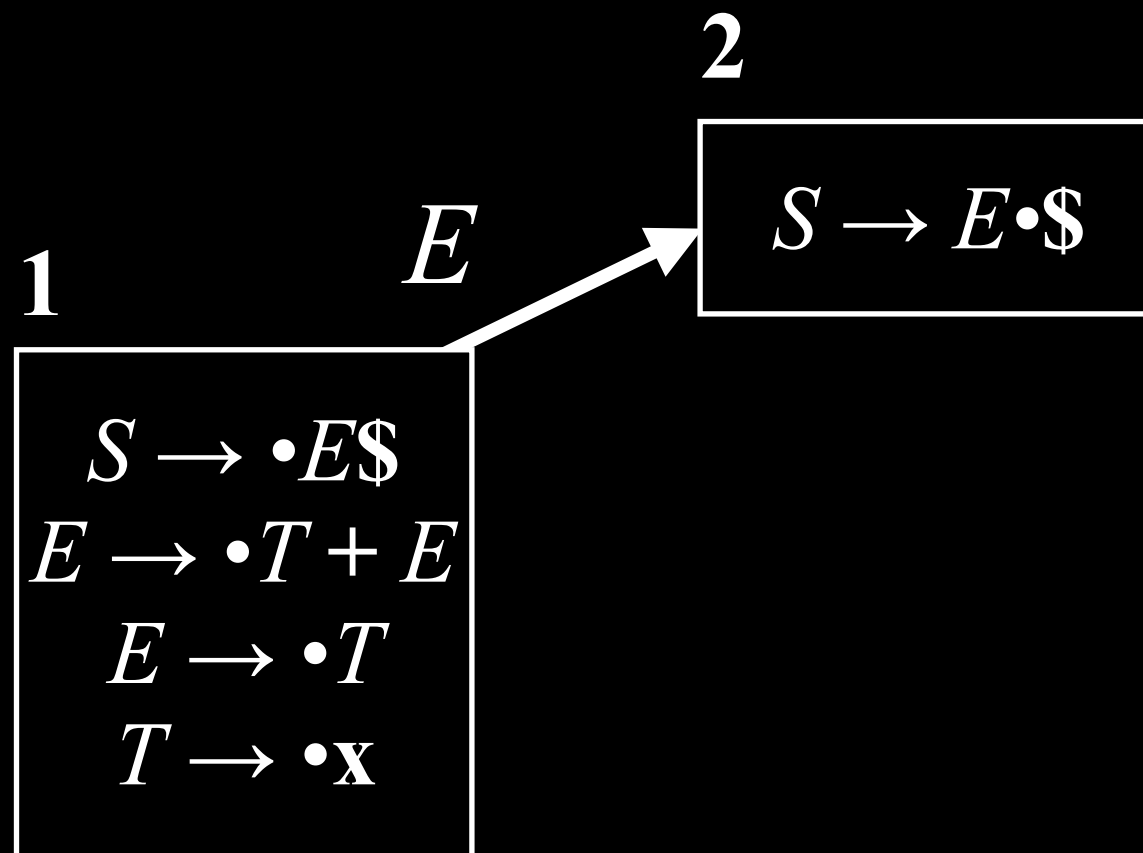
$$\begin{array}{l} S \longrightarrow \bullet ES \\ E \longrightarrow \bullet T + E \\ E \longrightarrow \bullet T \\ T \longrightarrow \bullet \mathbf{x} \end{array}$$

$$_0 S' \rightarrow ES$$

$$_1 E \rightarrow T + E$$

$$_2 E \rightarrow T$$

$$_3 T \rightarrow \mathbf{x}$$

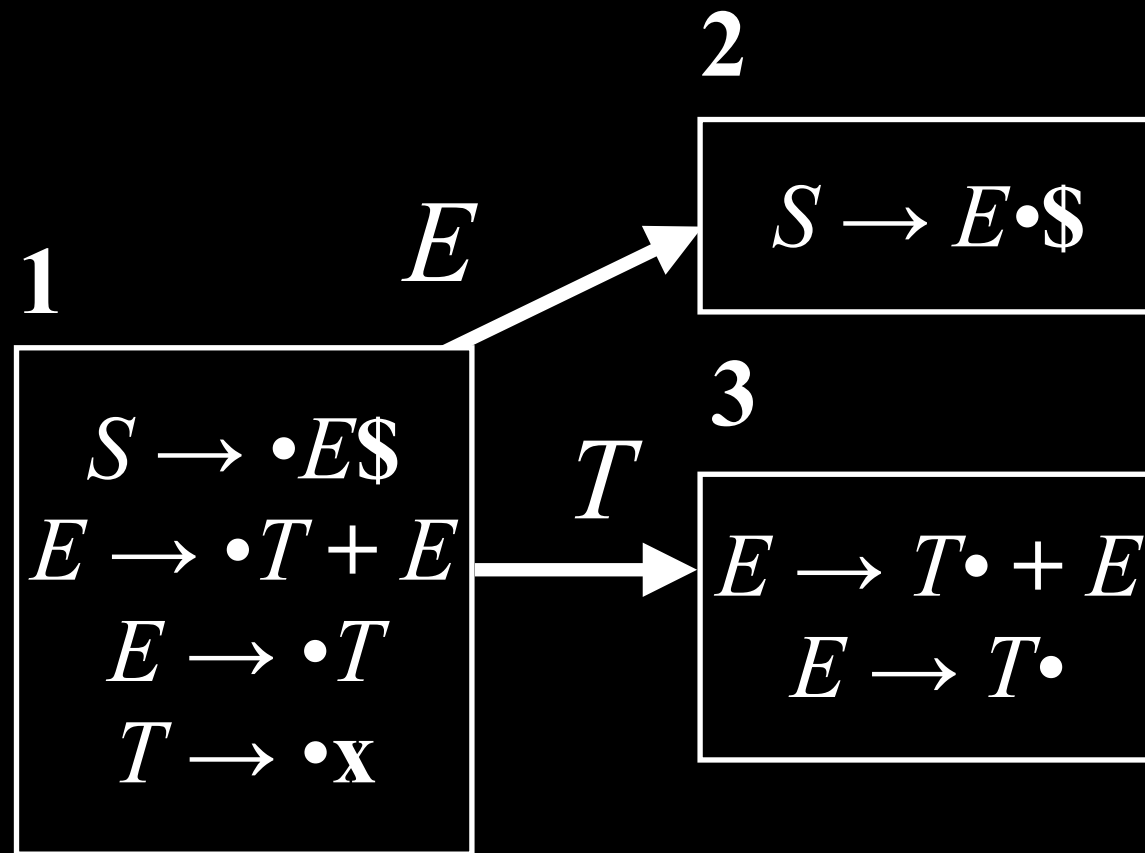


$$_0 S' \rightarrow ES$$

$$_1 E \rightarrow T + E$$

$$_2 E \rightarrow T$$

$$_3 T \rightarrow \mathbf{x}$$

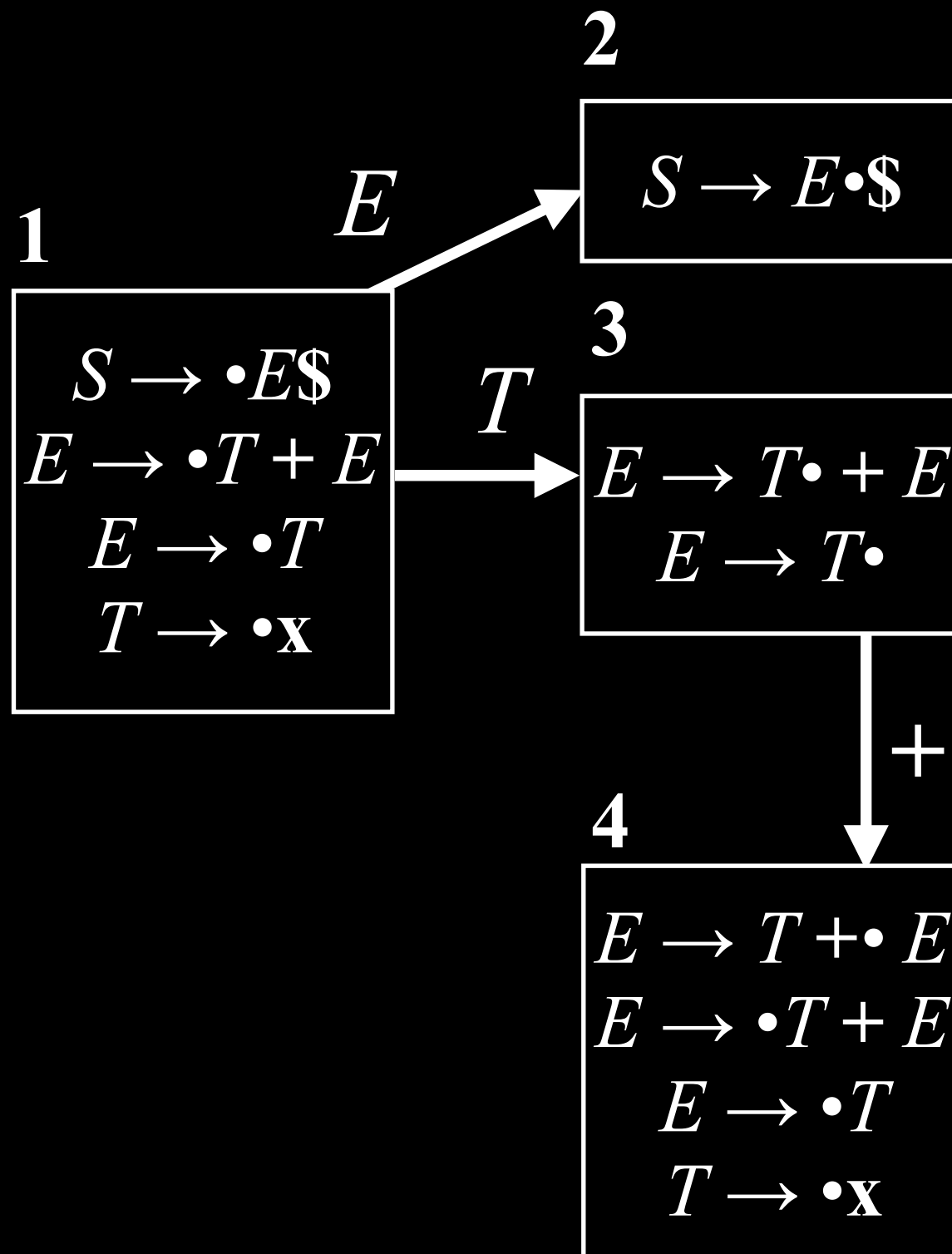


$$_0 S' \rightarrow ES$$

$$_1 E \rightarrow T + E$$

$$_2 E \rightarrow T$$

$$_3 T \rightarrow \mathbf{x}$$

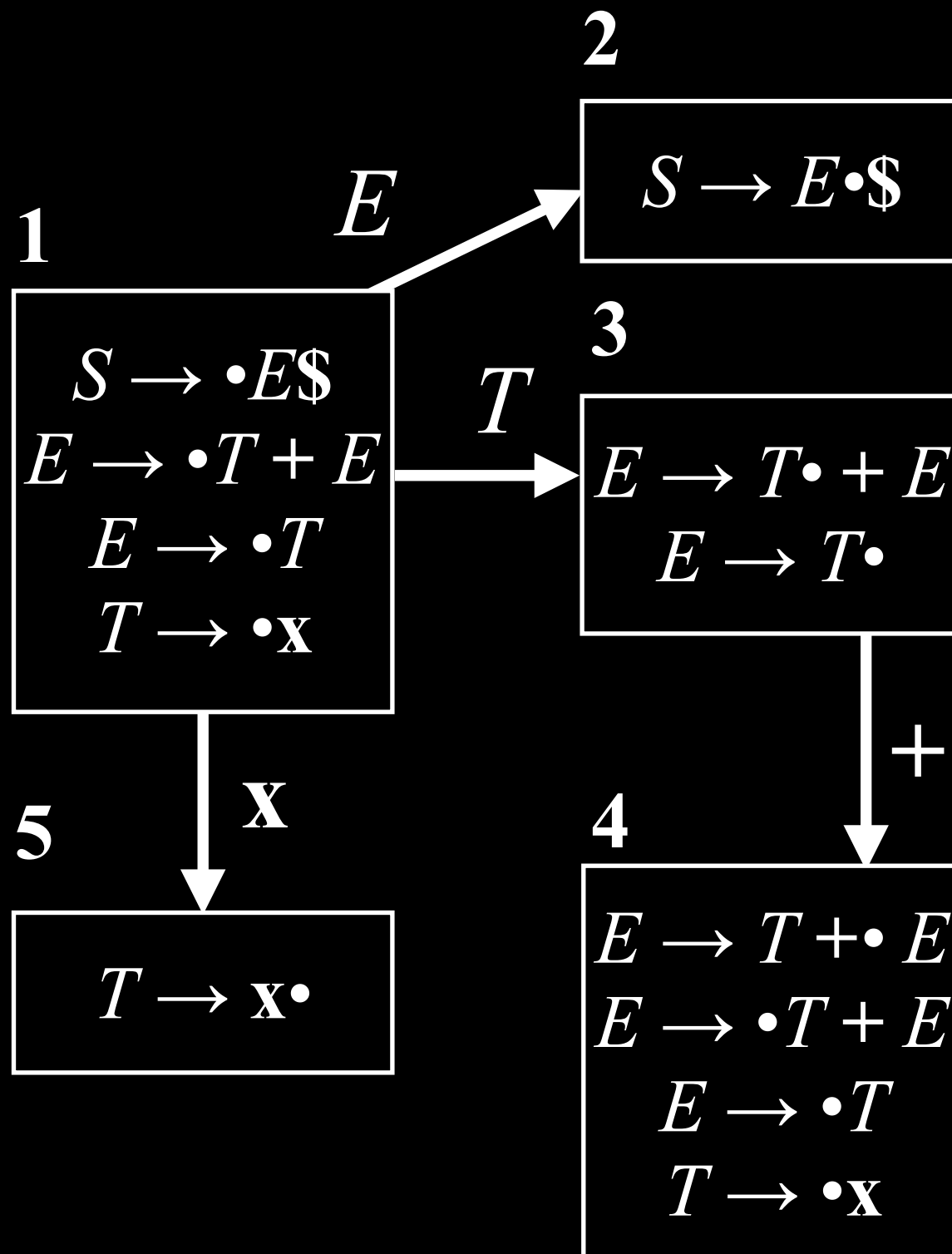


$$_0 S' \rightarrow E\$$$

$$_1 E \rightarrow T + E$$

$$_2 E \rightarrow T$$

$$_3 T \rightarrow \mathbf{x}$$

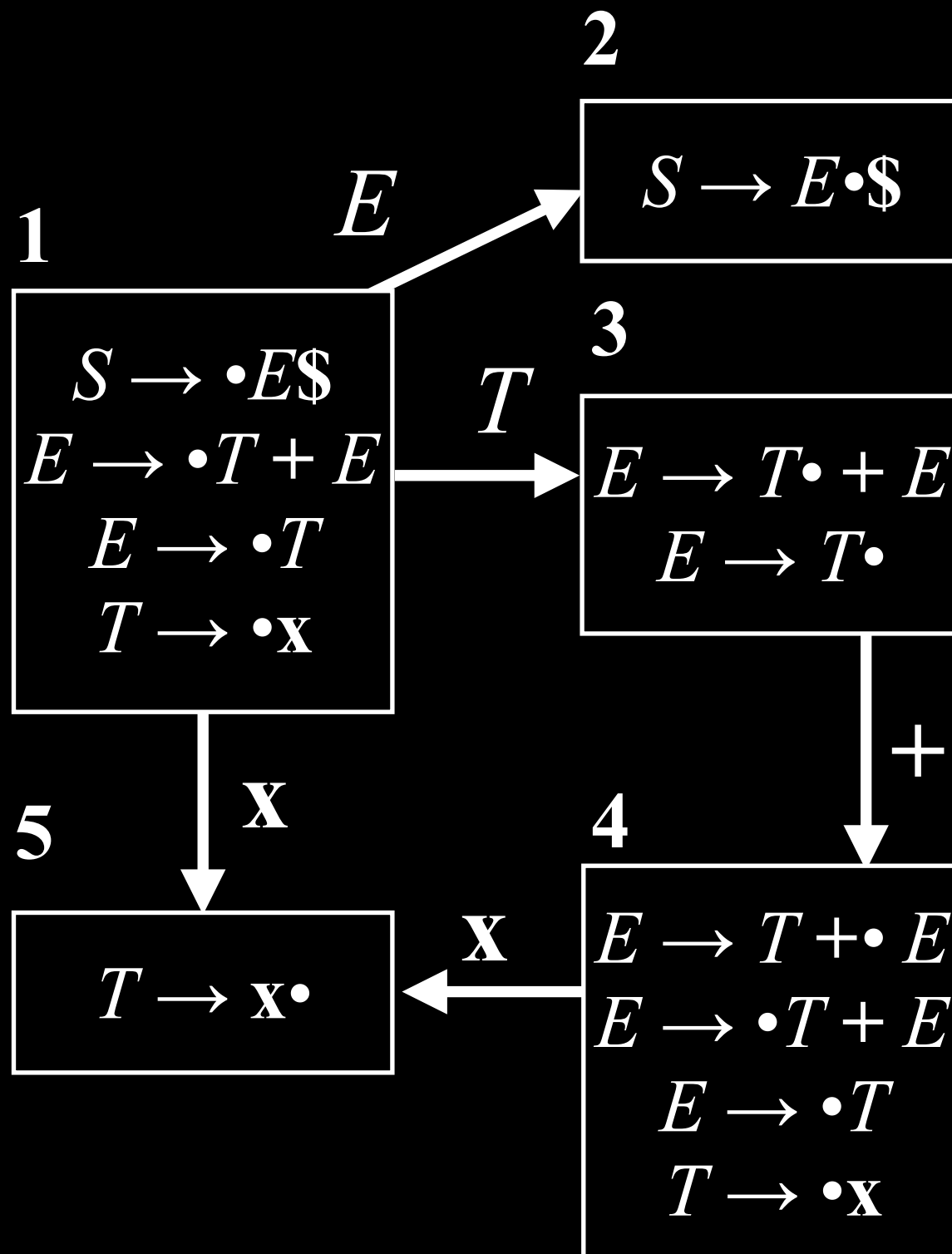


$$_0 S' \rightarrow ES$$

$$_1 E \rightarrow T + E$$

$$_2 E \rightarrow T$$

$$_3 T \rightarrow \mathbf{x}$$

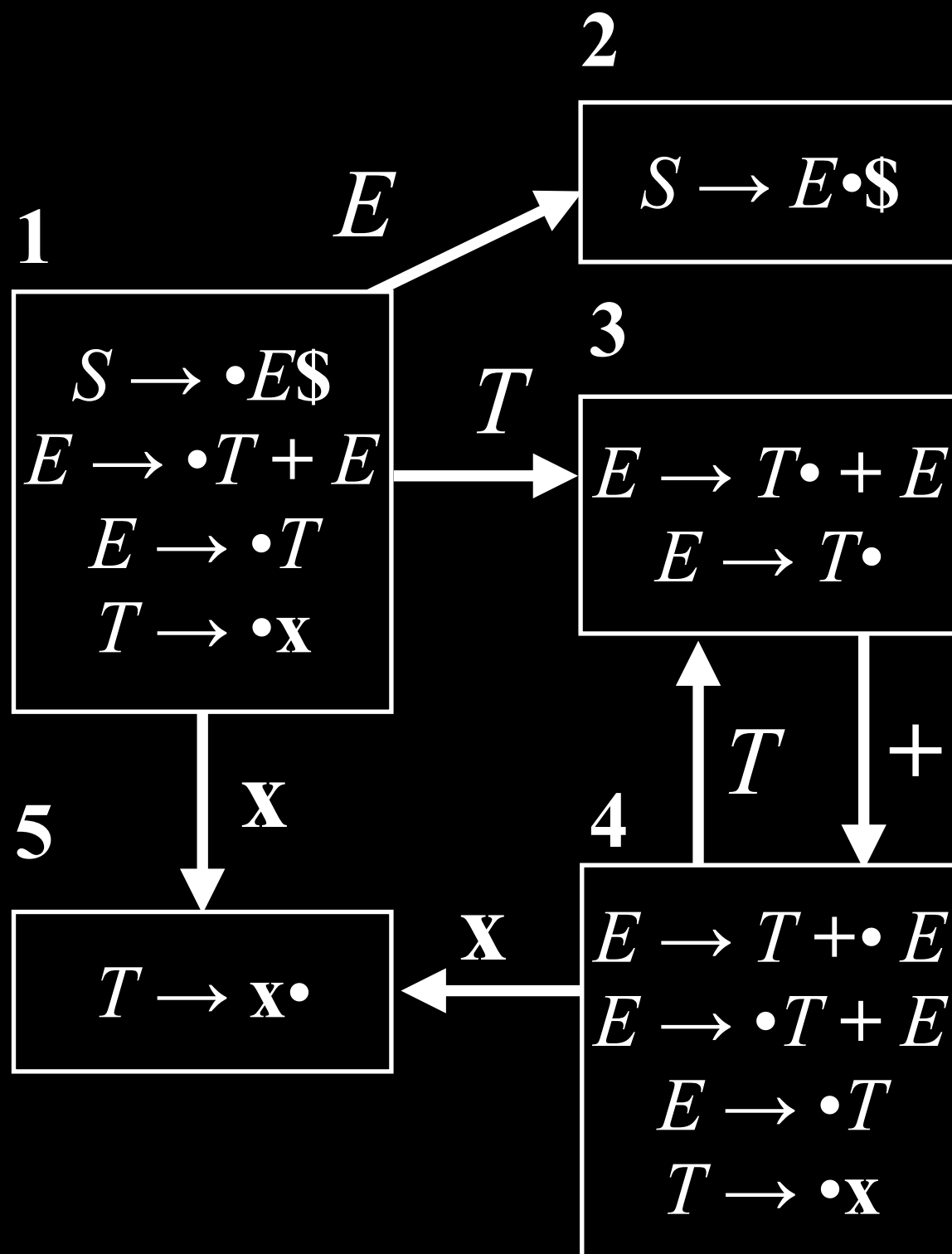


$$_0 S' \rightarrow ES$$

$$_1 E \rightarrow T + E$$

$$_2 E \rightarrow T$$

$$_3 T \rightarrow \mathbf{x}$$

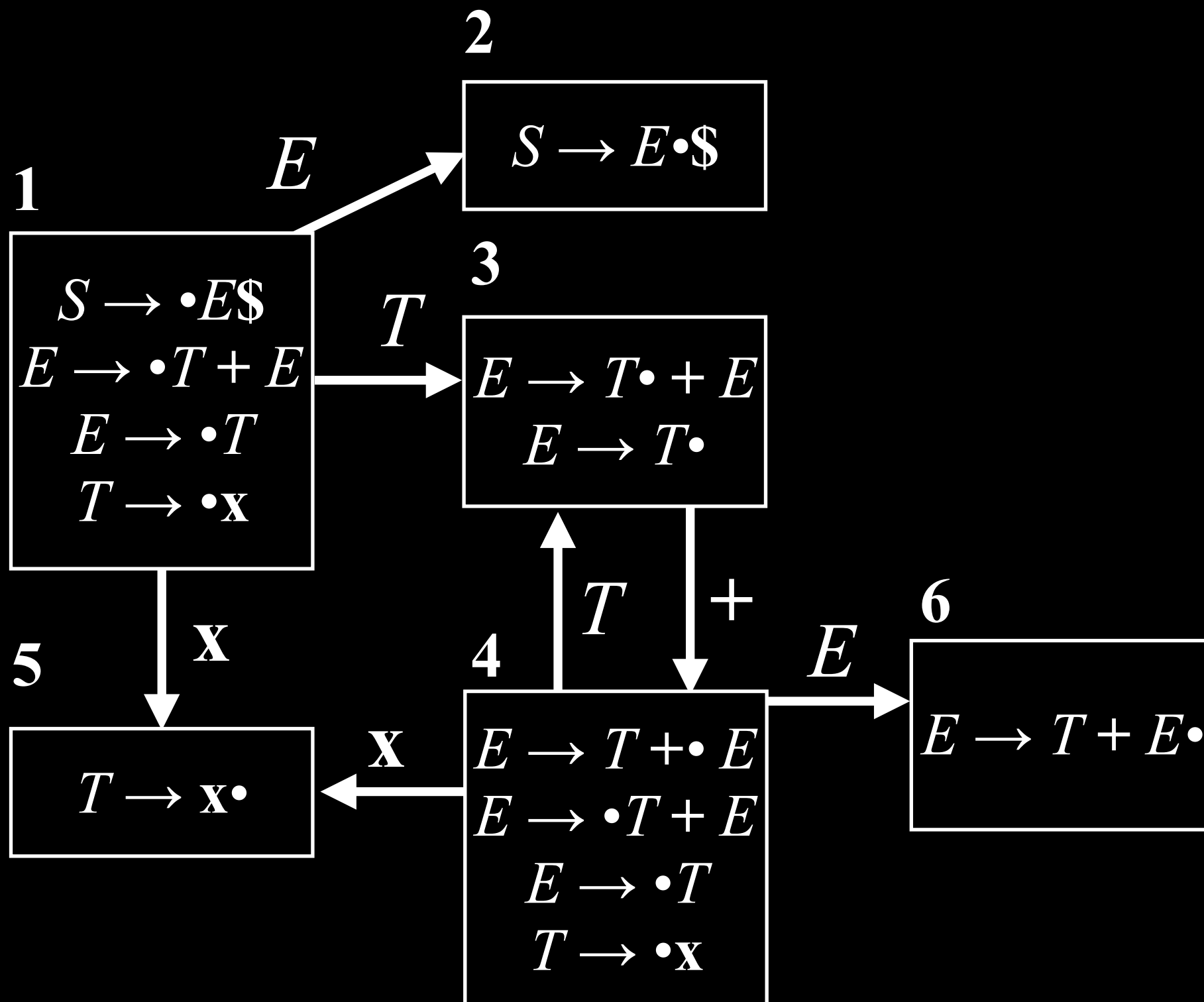


$$_0 S' \rightarrow ES$$

$$_1 E \rightarrow T + E$$

$$_2 E \rightarrow T$$

$$_3 T \rightarrow \mathbf{x}$$



$$_0 S' \rightarrow ES$$

$$_1 E \rightarrow T + E$$

$$_2 E \rightarrow T$$

$$_3 T \rightarrow \mathbf{x}$$

	\mathbf{x}	$+$	$\$$	E	T
1					
2					
3					
4					
5					
6					

$_0 S' \rightarrow ES$

$_1 E \rightarrow T + E$

$_2 E \rightarrow T$

$_3 T \rightarrow \mathbf{x}$

	\mathbf{x}	$+$	$\$$	E	T
1	s5			g2	g3
2			a		
3	r2	s4, r2	r2		
4	s5			g6	g3
5	r3	r3	r3		
6	r1	r1	r1		

Problemas

- Problemas na gramática ou limitações da técnica escolhida podem levar a conflitos
- shift-reduce: não consegue decidir entre uma ação de shift (ou mais) ou reduce
- reduce-reduce: não tem como decidir entre duas ou mais ações de reduce, geralmente por ambiguidade ou algum bug na gramática

Análise SLR

- Utiliza o conjunto FOLLOW para resolver conflitos
- Intuição: só reduz se o próximo token (lookahead) estiver no conjunto FOLLOW do não-terminal associado
- Na tabela de parsing, só inclui ação de reduce, caso o terminal esteja no conjunto FOLLOW

Autômatos SLR

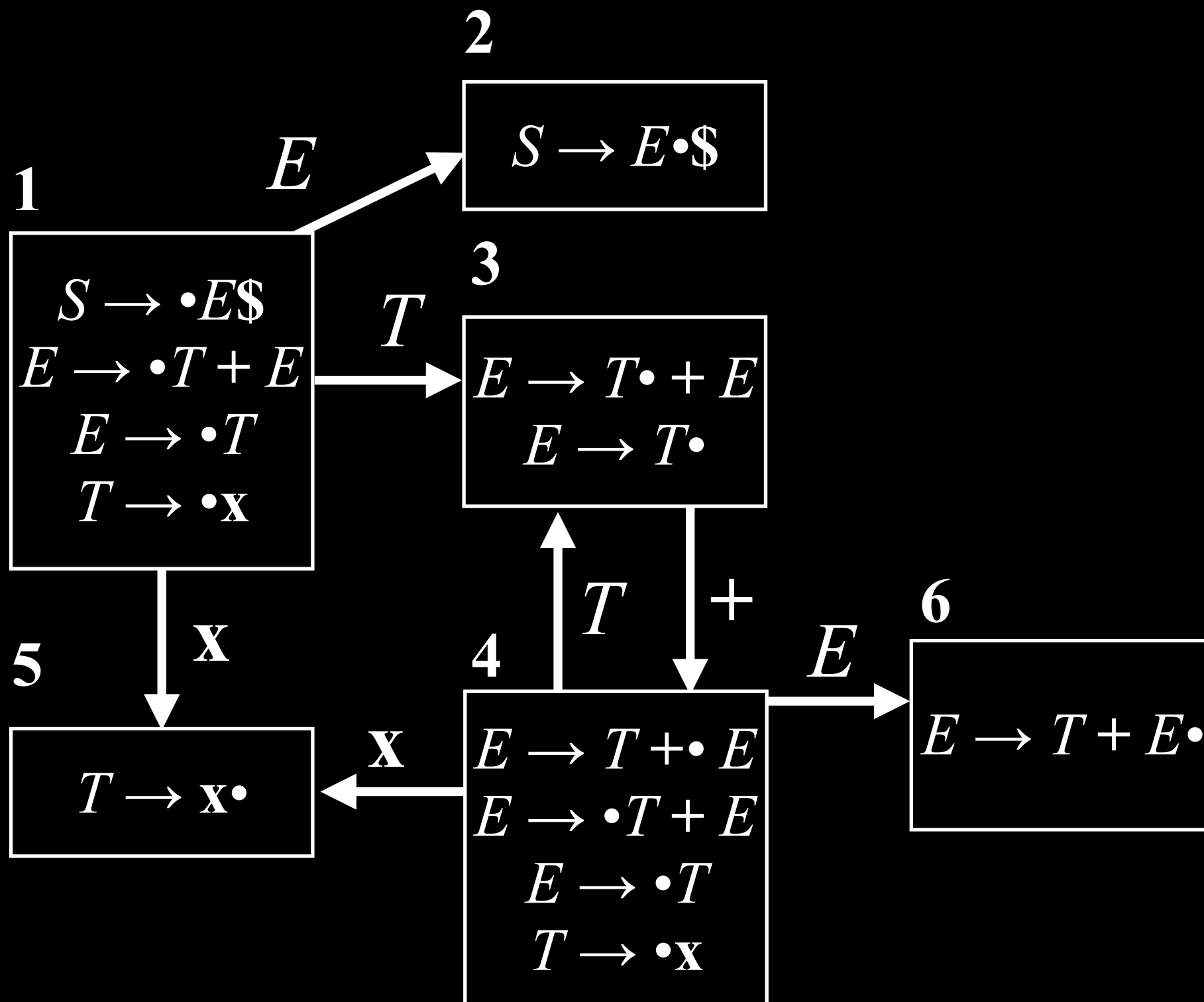
- Estados podem ter mais de um item de redução, caso conjuntos FOLLOW sejam distintos
- Estados podem misturar itens de shift com itens de redução, caso os terminais associados ao shift não estejam no FOLLOW dos itens de redução
- Isso não elimina todos os tipos de conflitos

$$_0 S' \rightarrow ES$$

$$_1 E \rightarrow T + E$$

$$_2 E \rightarrow T$$

$$_3 T \rightarrow \mathbf{x}$$



$$_0 S' \rightarrow ES$$

$$_1 E \rightarrow T + E$$

$$_2 E \rightarrow T$$

$$_3 T \rightarrow \mathbf{x}$$

	\mathbf{x}	$+$	$\$$	E	T
1	s5			g2	g3
2			a		
3		s4	r2		
4	s5			g6	g3
5		r3	r3		
6			r1		