

# Compiler

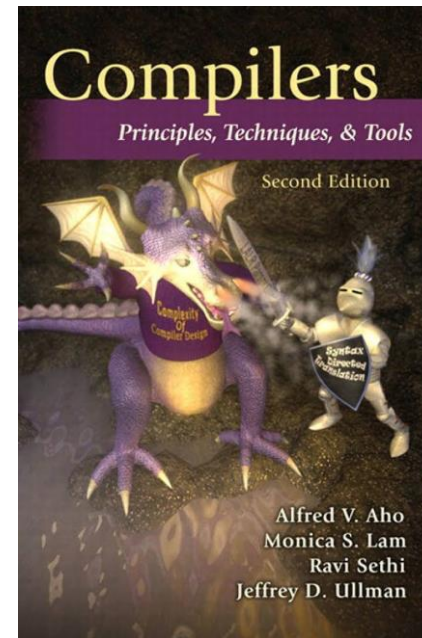
---

Lec 08

# Book

---

Compilers: Principles, Techniques, and Tools is a computer science textbook by Alfred V. Aho, Monica S. Lam, Ravi Sethi, and Jeffrey D. Ullman about compiler construction.



# PowerPoint

<http://www.bu.edu.eg/staff/ahmedaboalatah14-courses/14779>

The screenshot shows a web page for Benha University. The header includes the university logo and name, and a staff profile for Ahmed Hassan Ahmed Abu El Atta with a 'Log out' link. The main content area displays course details for 'Compilers' under the user 'Ass. Lect. Ahmed Hassan Ahmed Abu El Atta'. A sidebar on the left lists various university-related links. A vertical column of social media icons is on the right. The course details are presented in a table-like format with blue headers and white content cells.

**Benha University** Staff Search: **Welcome: Ahmed Hassan Ahmed Abu El Atta (Log out)**

You are in: [Home/Courses/Compilers](#) [Back To Courses](#)

**Ass. Lect. Ahmed Hassan Ahmed Abu El Atta :: Course Details: Compilers** [add course](#) | [edit course](#)

Course name	Compilers
Level	Undergraduate
Last year taught	2018
Course description	Not Uploaded

**Course password**

Course files	<a href="#">add files</a>
Course URLs	<a href="#">add URLs</a>
Course assignments	<a href="#">add assignments</a>
Course Exams & Model Answers	<a href="#">add exams</a>

**Benha University**  
Home  
النسخة العربية  
My C.V.  
About  
Courses  
Publications  
**Inlinks(Competition)**  
Theses  
Reports  
Published books  
Workshops / Conferences  
Supervised PhD  
Supervised MSc  
Supervised Projects  
Education  
Language skills  
Academic Positions  
Administrative Positions

Google  
Benha University  
RG  
in  
f  
Twitter  
g+  
YouTube  
W  
Z  
(edit)

# Syntax Analysis

---

PART V

# LR(0) Item

---

An LR(0) item of G is a **production** of G with the **dot** at some position of the body:

- For  $A \rightarrow XYZ$  we have following items
  - $A \rightarrow \cdot XYZ$
  - $A \rightarrow X \cdot YZ$
  - $A \rightarrow XY \cdot Z$
  - $A \rightarrow XYZ \cdot$
- In a state having  $A \rightarrow \cdot XYZ$  we hope to see a string derivable from XYZ next on the input.
- The production  $A \rightarrow \epsilon$  generates only one item,  $A \rightarrow \cdot$ .

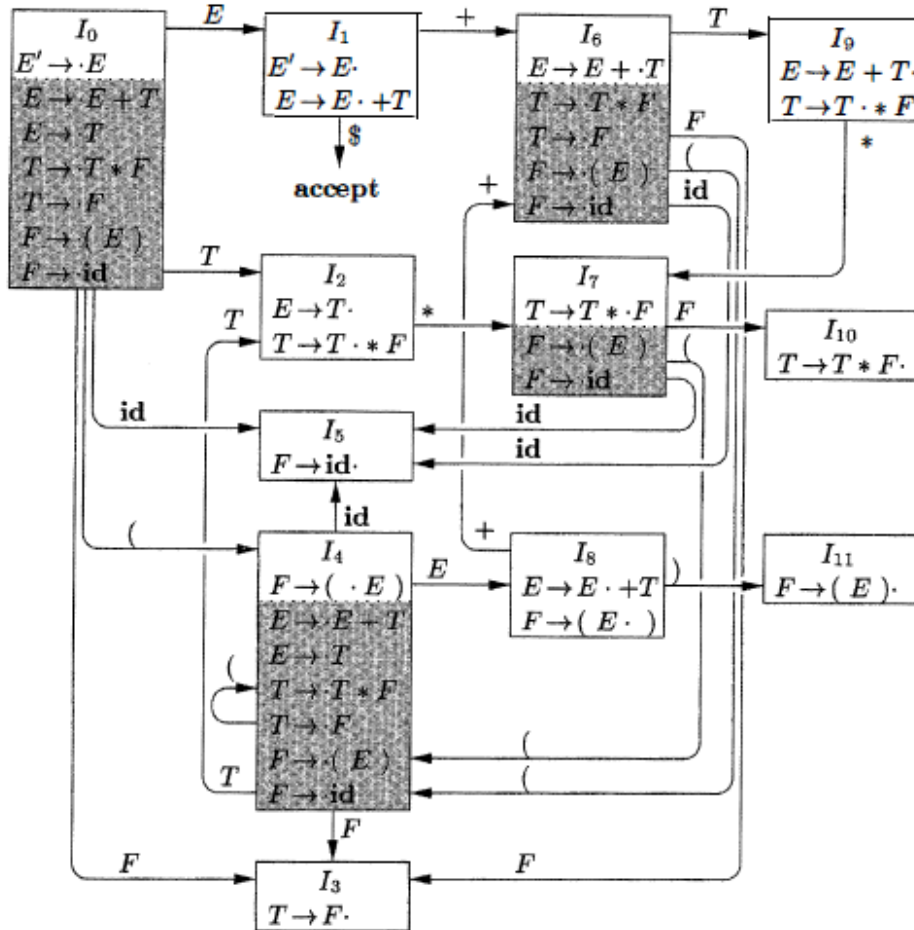
# Closure of Item Sets

---

If  $I$  is a set of items for a grammar  $G$ , then  $\text{Closure}(I)$  is the set of items constructed from  $I$  by the two rules:

1. Initially, add every item in  $I$  to  $\text{CLOSURE}(I)$ .
2. If  $A \rightarrow \alpha \cdot B \beta$  is in  $\text{CLOSURE}(I)$  and  $B \rightarrow \gamma$  is a production, then add the item  $B \rightarrow \cdot \gamma$  to  $\text{CLOSURE}(I)$ , if it is not already there. Apply this rule until no more new items can be added to  $\text{CLOSURE}(I)$ .

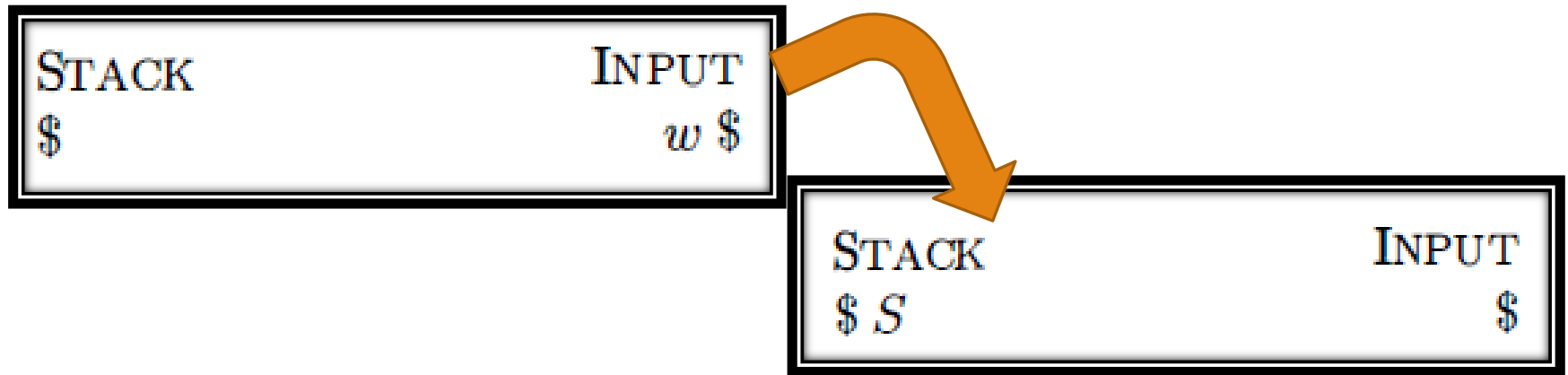
# LR(0) Automaton



$E \rightarrow E + T \mid T$   
 $T \rightarrow T * F \mid F$   
 $F \rightarrow ( E ) \mid id$

---

$S' \rightarrow S.$



To construct the canonical LR(0) collection for a grammar, we define an **augmented grammar**



# Augmented Grammar

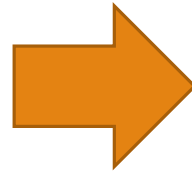
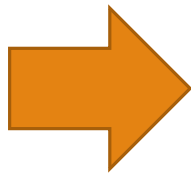
---

If  $G$  is a grammar with start symbol  $S$ , then  $G'$ , the augmented grammar for  $G$ , is  $G$  with a new start symbol  $S'$  and production  $S' \rightarrow S$

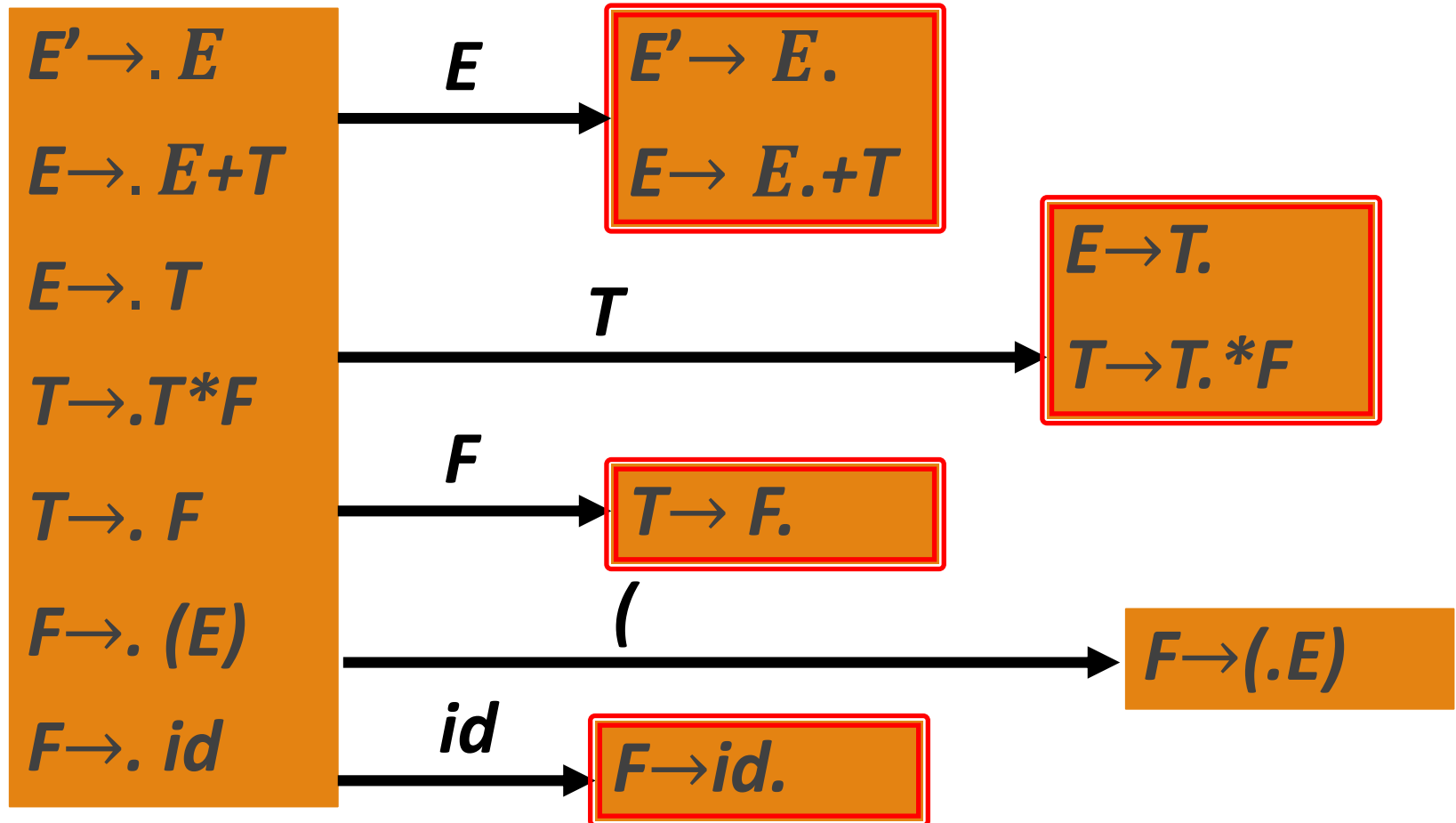
The purpose of this new starting production is to indicate to the parser when it should stop parsing and announce acceptance of the input.

That is, acceptance occurs when and only when the parser is about to reduce by  $S' \rightarrow S$ .

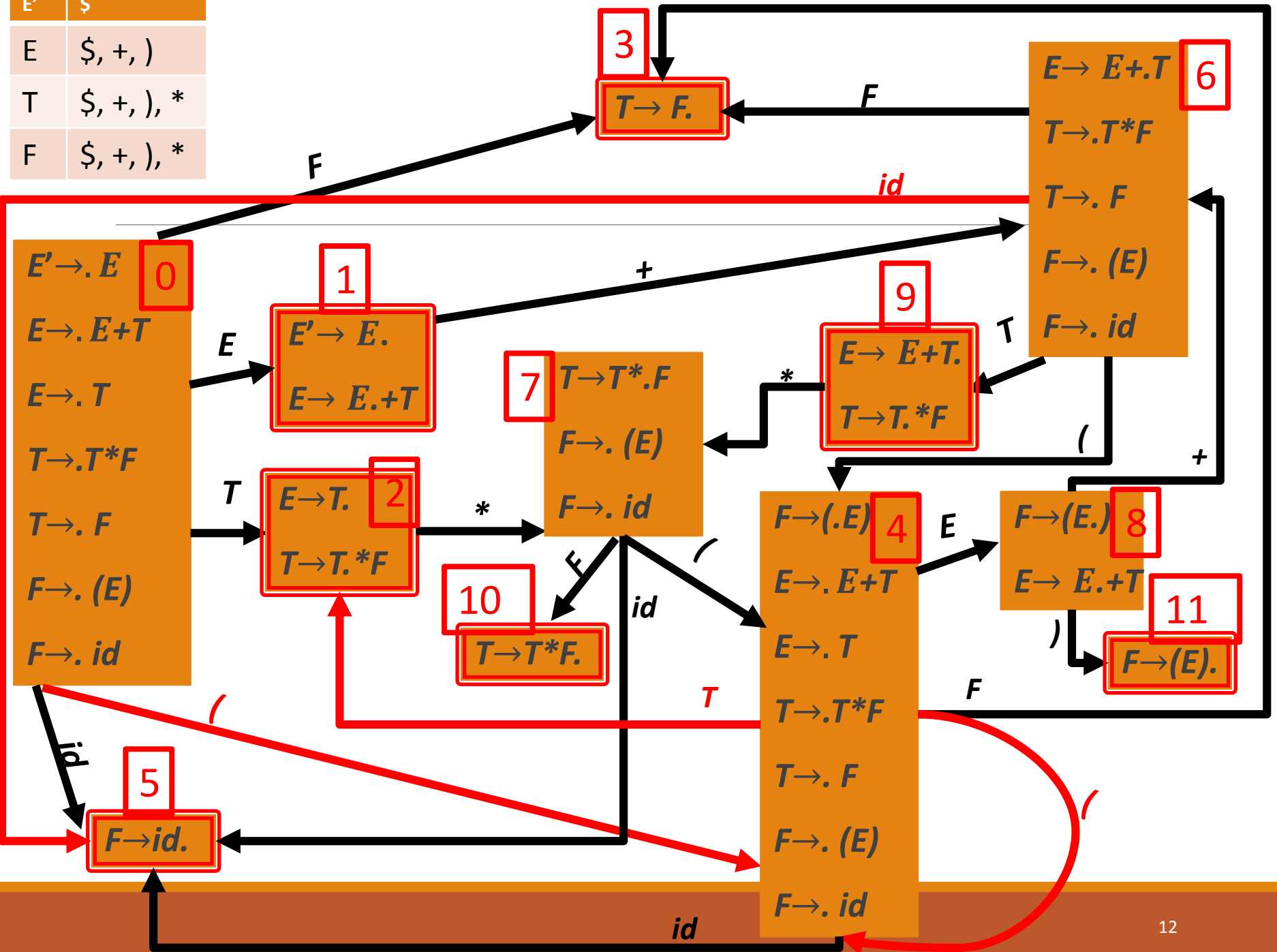
# Example LR(1)

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

$$\begin{aligned} E' &\rightarrow E \\ E &\rightarrow E + T \mid T \\ T &\rightarrow T * F \mid F \\ F &\rightarrow ( E ) \mid \text{id} \end{aligned}$$
$$E' \rightarrow \cdot E$$

$$\begin{aligned} E' &\rightarrow \cdot E \\ E &\rightarrow \cdot E + T \\ E &\rightarrow \cdot T \\ T &\rightarrow \cdot T * F \\ T &\rightarrow \cdot F \\ F &\rightarrow \cdot ( E ) \\ F &\rightarrow \cdot \text{id} \end{aligned}$$

# Example



$E'$	$\$$
E	$\$, +, )$
T	$\$, +, ), *$
F	$\$, +, ), *$



# Example LR(1)

STATE	ACTION					GOTO			
	id	+	*	(	)	\$	<i>E</i>	<i>T</i>	<i>F</i>
0	s5			s4			1	2	3
1		s6				acc			
2		r2	s7		r2	r2			
3		r4	r4		r4	r4			
4	s5			s4			8	2	3
5		r6	r6		r6	r6			
6	s5			s4				9	3
7	s5			s4					10
8		s6			s11				
9		r1	s7		r1	r1			
10		r3	r3		r3	r3			
11		r5	r5		r5	r5			

# Moves of an LR parser on $\text{id} * \text{id} + \text{id}$

	STACK	SYMBOLS	INPUT	ACTION
(1)	0		<b>id * id + id \$</b>	shift
(2)	0 5	<b>id</b>	<b>* id + id \$</b>	reduce by $F \rightarrow \text{id}$
(3)	0 3	$F$	<b>* id + id \$</b>	reduce by $T \rightarrow F$
(4)	0 2	$T$	<b>* id + id \$</b>	shift
(5)	0 2 7	$T *$	<b>id + id \$</b>	shift
(6)	0 2 7 5	$T * \text{id}$	<b>+ id \$</b>	reduce by $F \rightarrow \text{id}$
(7)	0 2 7 10	$T * F$	<b>+ id \$</b>	reduce by $T \rightarrow T * F$
(8)	0 2	$T$	<b>+ id \$</b>	reduce by $E \rightarrow T$
(9)	0 1	$E$	<b>+ id \$</b>	shift
(10)	0 1 6	$E +$	<b>id \$</b>	shift
(11)	0 1 6 5	$E + \text{id}$	<b>\$</b>	reduce by $F \rightarrow \text{id}$
(12)	0 1 6 3	$E + F$	<b>\$</b>	reduce by $T \rightarrow F$
(13)	0 1 6 9	$E + T$	<b>\$</b>	reduce by $E \rightarrow E + T$
(14)	0 1	$E$	<b>\$</b>	accept

