بسم اللّه الرّحمن الرّحیم

دانشگاه صنعتی اصفهان ـ دانشکدهٔ مهندسی برق و کامپیوتر
(نیم‌سال تحصیلی ۴۰۱۲)

کامپایلر

حسین فلسفین

Example: The canonical parsing table for grammar (4.55)

$$
\begin{aligned}
S' &\rightarrow S \\
S &\rightarrow C\ C \\
C &\rightarrow c\ C \mid d
\end{aligned}
$$

The initial set of items is

$$
\begin{aligned}
I_0: \quad & S \rightarrow \cdot S, \$ \\
& S \rightarrow \cdot CC, \ \$ \\
& C \rightarrow \cdot cC, \ c/d \\
& C \rightarrow \cdot d, \ c/d
\end{aligned}
$$

$\mathrm{GOTO}(I_0, S)$

$$I_1 : S' \to S\bullet, \$$$

$\mathrm{GOTO}(I_0, C)$

$$\begin{aligned} I_2 : \quad & S \to C\cdot C, \ \$ \\ & C \to \cdot cC, \ \$ \\ & C \to \cdot d, \ \$ \end{aligned}$$

$\mathrm{GOTO}(I_0, c)$

$$\begin{aligned} I_3 : \quad & C \to c\cdot C, \ c/d \\ & C \to \cdot cC, \ c/d \\ & C \to \cdot d, \ c/d \end{aligned}$$

$\mathrm{GOTO}(I_0, d)$

$$I_4 : C \to d\bullet, c/d$$

The GOTO *function for* I_2

For GOTO$(I_2,\ C)$ we get

$$I_5: \quad S \to CC\cdot, \$$$

no closure being needed. To compute GOTO$(I_2,\ c)$ we take the closure of $\{[C \to c\cdot C,\ \$]\}$, to obtain

$$\begin{aligned} I_6: \quad & C \to c\cdot C,\ \$ \\ & C \to \cdot cC,\ \$ \\ & C \to \cdot d,\ \$ \end{aligned}$$

Note that I_6 differs from I_3 only in second components. We shall see that it is common for several sets of LR(1) items for a grammar to have the same first components and differ in their second components. When we construct the collection of sets of LR(0) items for the same grammar, each set of LR(0) items will coincide with the set of first components of one or more sets of LR(1) items. We shall have more to say about this phenomenon when we discuss LALR parsing.

Continuing with the GOTO function for I_2, GOTO(I_2, d) is seen to be

$$I_7: \quad C \to d\cdot,\ \$$$

The GOTO *function for* I_3 *and* I_6

Turning now to I_3, the GOTO's of I_3 on c and d are I_3 and I_4, respectively, and $\text{GOTO}(I_3, C)$ is

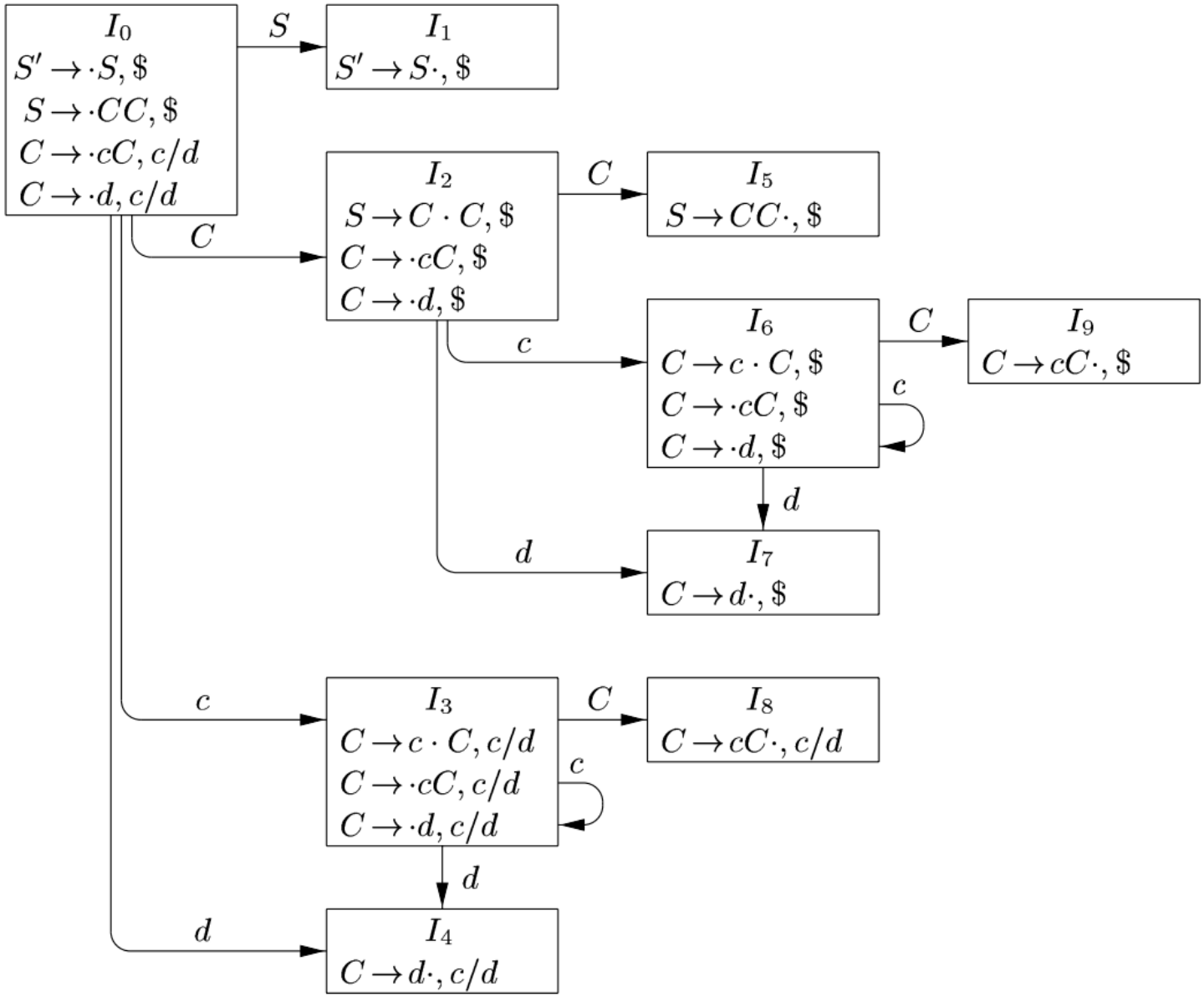
$$I_8: \quad C \to cC\cdot,\ c/d$$

I_4 and I_5 have no GOTO's, since all items have their dots at the right end. The GOTO's of I_6 on c and d are I_6 and I_7, respectively, and $\text{GOTO}(I_6, C)$ is

$$I_9: \quad C \to cC\cdot,\ \$$$

The remaining sets of items yield no GOTO's, so we are done. Figure 4.41 shows the ten sets of items with their goto's. □

The GOTO *graph for grammar (4.55)*



Example:

$$
\begin{aligned}
0.\ & E' ::= E \\
1.\ & E ::= E + T \\
2.\ & E ::= T \\
3.\ & T ::= T * F \\
4.\ & T ::= F \\
5.\ & F ::= (E) \\
6.\ & F ::= \texttt{id}
\end{aligned}
$$

$$
\begin{aligned}
s_0 = \{&[E' ::= \cdot\ E, \#], \\
&[E ::= \cdot\ E + T, +/\#], \\
&[E ::= \cdot\ T, +/\#], \\
&[T ::= \cdot\ T * F, +/*/\#], \\
&[T ::= \cdot\ F, +/*/\#], \\
&[F ::= \cdot (E), +/*/\#], \\
&[F ::= \cdot\ \texttt{id}, +/*/\#]\}
\end{aligned}
\qquad
\begin{aligned}
&\text{goto}(s_0, E) = s_1 \\
&\text{goto}(s_0, T) = s_2 \\
&\text{goto}(s_0, F) = s_3 \\
&\text{goto}(s_0, () = s_4 \\
&\text{goto}(s_0, \texttt{id}) = s_5
\end{aligned}
$$

$$
\begin{aligned}
s_1 = \{&[\text{E}' ::= E\ \cdot, \#], \\
&[E ::= E \cdot + T, +/\#]\}
\end{aligned}
\qquad
\text{goto}(s_1, +) = s_6
$$

$$s_2 = \{[E \ ::= T \cdot, \texttt{+/\#}],$$
$$[T \ ::= T \cdot \texttt{*} \ F, \texttt{+/*/\#}]\}$$

$$\text{goto}(s_2, \texttt{*}) = s_7$$

$$s_3 = \{[T \ ::= F \cdot, \texttt{+/*/\#}]\}$$

$$s_4 = \{[F \ ::= \texttt{(} \cdot E \texttt{)}, \ \texttt{+/*/\#}],$$
$$[E \ ::= \cdot E \texttt{+} T, \texttt{+/)}],$$
$$[E \ ::= \cdot T, \texttt{+/)}],$$
$$[T \ ::= \cdot T \texttt{*} F, \texttt{+/*/)}],$$
$$[T \ ::= \cdot F, \texttt{+/*/)}],$$
$$[F \ ::= \cdot \texttt{(} E \texttt{)}, \texttt{+/*/)}],$$
$$[F \ ::= \cdot \texttt{id}, \texttt{+/*/)}]\}$$

$$\text{goto}(s_4, E) = s_8$$
$$\text{goto}(s_4, T) = s_9$$
$$\text{goto}(s_4, F) = s_{10}$$
$$\text{goto}(s_4, \texttt{(}) = s_{11}$$
$$\text{goto}(s_4, \texttt{id}) = s_{12}$$

$$s_5 = \{[F \ ::= \texttt{id} \cdot, \texttt{+/*/\#}]\}$$

$$
\begin{aligned}
s_6 = \{&[E ::= E + \cdot T, +/\#], \\
&[T ::= \cdot T * F, +/*/\#], \\
&[T ::= \cdot F, +/*/\#], \\
&[F ::= \cdot (E), +/*/\#], \\
&[F ::= \cdot \mathtt{id}, +/*/\#]\}
\end{aligned}
\qquad
\begin{aligned}
&\mathrm{goto}(s_6, T) = s_{13} \\
&\mathrm{goto}(s_6, F) = s_3 \\
&\mathrm{goto}(s_6, () = s_4 \\
&\mathrm{goto}(s_6, \mathtt{id}) = s_5
\end{aligned}
$$

$$
\begin{aligned}
s_7 = \{&[T ::= T * \cdot F, +/*/\#], \\
&[F ::= \cdot (E), +/*/\#], \\
&[F ::= \cdot \mathtt{id}, +/*/\#]\}
\end{aligned}
\qquad
\begin{aligned}
&\mathrm{goto}(s_7, F) = s_{14} \\
&\mathrm{goto}(s_7, () = s_4 \\
&\mathrm{goto}(s_7, \mathtt{id}) = s_5
\end{aligned}
$$

$$
\begin{aligned}
s_8 = \{&[F ::= (E \cdot), +/*/\#], \\
&[E ::= E \cdot + T, +/)]\}
\end{aligned}
\qquad
\begin{aligned}
&\mathrm{goto}(s_8,)) = s_{15} \\
&\mathrm{goto}(s_8, +) = s_{16}
\end{aligned}
$$

$$s_9 = \{[E ::= T \cdot, +/)], \quad [T ::= T \cdot * F, +/*/)]\} \qquad \text{goto}(s_9, *) = s_{17}$$

$$s_{10} = \{[T ::= F \cdot, +/*/)]\}$$

$$s_{13} = \{[E ::= E + T \cdot, +/\#], \quad [T ::= T \cdot * F, +/*/\#]\} \qquad \text{goto}(s_{13}, *) = s_7$$

$$s_{14} = \{[T ::= T * F \cdot, +/*/\#]\}$$

$$s_{15} = \{[F ::= (E) \cdot, +/*/\#]\}$$

$$
\begin{aligned}
s_{16} = \{&[E \ ::= E \texttt{+} \cdot T, \texttt{+/)}], \\
&[T \ ::= \cdot T \texttt{*} F, \texttt{+/*/)}], \\
&[T \ ::= \cdot F, \texttt{+/*/)}], \\
&[F \ ::= \cdot \texttt{(} E \texttt{)}, \texttt{+/*/)}], \\
&[F \ ::= \cdot \texttt{id}, \texttt{+/*/)}]\}
\end{aligned}
\qquad
\begin{aligned}
&\text{goto}(s_{16}, T) = s_{19} \\
&\text{goto}(s_{16}, F) = s_{10} \\
&\text{goto}(s_{16}, \texttt{(}) = s_{11} \\
&\text{goto}(s_{16}, \texttt{id}) = s_{12}
\end{aligned}
$$

$$
\begin{aligned}
s_{17} = \{&[T \ ::= T \texttt{*} \cdot F, \texttt{+/*/)}], \\
&[F \ ::= \cdot \texttt{(} E \texttt{)}, \texttt{+/*/)}], \\
&[F \ ::= \cdot \texttt{id}, \texttt{+/*/)}]\}
\end{aligned}
\qquad
\begin{aligned}
&\text{goto}(s_{17}, F) = s_{20} \\
&\text{goto}(s_{17}, \texttt{(}) = s_{11} \\
&\text{goto}(s_{17}, \texttt{id}) = s_{12}
\end{aligned}
$$

$$
\begin{aligned}
s_{18} = \{&[F \ ::= \texttt{(} E \cdot \texttt{)}, \texttt{+/*/)}], \\
&[E \ ::= E \cdot \texttt{+} T, \texttt{+/)}]\}
\end{aligned}
\qquad
\begin{aligned}
&\text{goto}(s_{18}, \texttt{)}) = s_{21} \\
&\text{goto}(s_{18}, \texttt{+}) = s_{16}
\end{aligned}
$$

$$
\begin{aligned}
s_{19} = \{&[E \ ::= E \texttt{+} T \cdot, \texttt{+/)}], \\
&[T \ ::= T \cdot \texttt{*} F, \texttt{+/*/)}]\}
\end{aligned}
\qquad
\text{goto}(s_{19}, \texttt{*}) = s_{17}
$$

$$
s_{20} = \{[T \ ::= T \texttt{*} F \cdot, \texttt{+/*/)}]\}
$$

$$
s_{21} = \{[F \ ::= \texttt{(} E \texttt{)} \cdot, \texttt{+/*/)}]\}
$$

Canonical LR(1) *Parsing Tables*

Algorithm 4.56: Construction of canonical-LR parsing tables.

INPUT: An augmented grammar G'.

OUTPUT: The canonical-LR parsing table functions ACTION and GOTO for G'.

METHOD:

1. Construct $C' = \{I_0, I_1, \cdots, I_n\}$, the collection of sets of LR(1) items for G'.
2. State i of the parser is constructed from I_i. The parsing action for state i is determined as follows.
   (a) If $[A \to \alpha \cdot a\beta, b]$ is in I_i and $\text{GOTO}(I_i, a) = I_j$, then set ACTION$[i, a]$ to "shift j." Here a must be a terminal.
   (b) If $[A \to \alpha \cdot, a]$ is in I_i, $A \neq S'$, then set ACTION$[i, a]$ to "reduce $A \to \alpha$."
   (c) If $[S' \to S \cdot, \$]$ is in I_i, then set ACTION$[i, \$]$ to "accept."

   If any conflicting actions result from the above rules, we say the grammar is not LR(1). The algorithm fails to produce a parser in this case.
3. The goto transitions for state i are constructed for all nonterminals A using the rule: If $\text{GOTO}(I_i, A) = I_j$, then $\text{GOTO}[i, A] = j$.
4. All entries not defined by rules (2) and (3) are made "error."
5. The initial state of the parser is the one constructed from the set of items containing $[S' \to \cdot S, \$]$.

□

جدول نظیر مثال اول، یعنی گرامر (۴.۵۵) از کتاب آهو

STATE	ACTION			GOTO	
	c	d	\$	S	C
0	s3	s4		1	2
1			acc		
2	s6	s7			5
3	s3	s4			8
4	r3	r3			
5			r1		
6	s6	s7			9
7			r3		
8	r2	r2			
9			r2		

جدول نظیر مثال دوم

	Action						Goto		
	+	*	(	)	id	#	E	T	F
0			s4		s5		1	2	3
1	s6					accept			
2	r2	s7				r2			
3	r4	r4				r4			
4			s11		s12		8	9	10
5	r6	r6				r6			
6			s4		s5			13	3
7			s4		s5				14
8	s16			s15					
9	r2	s17		r2					
10	r4	r4		r4					
11			s11		s12		18	9	10
12	r6	r6		r6					
13	r1	s7				r1			
14	r3	r3				r3			
15	r5	r5				r5			
16			s11		s12			19	10
17			s11		s12				20
18	s16			s21					
19	r1	s17		r1					
20	r3	r3		r3					
21	r5	r5		r5					

The table formed from the parsing ACTION ***and*** GOTO ***functions produced by Algorithm 4.56 is called the canonical*** $\mathrm{LR}(1)$ ***parsing table. An LR parser using this table is called a canonical-***$\mathrm{LR}(1)$ ***parser. If the parsing action function has no multiply defined entries, then the given grammar is called an*** $\mathrm{LR}(1)$ ***grammar.***

Every $\mathrm{SLR}(1)$ ***grammar is an*** $\mathrm{LR}(1)$ ***grammar, but for an*** $\mathrm{SLR}(1)$ ***grammar the canonical LR parser may have more states than the SLR parser for the same grammar.***