LR(1) Parser

Dinesh Gopalani
dgopalani.cse@mnit.ac.in

LR(1) Parser

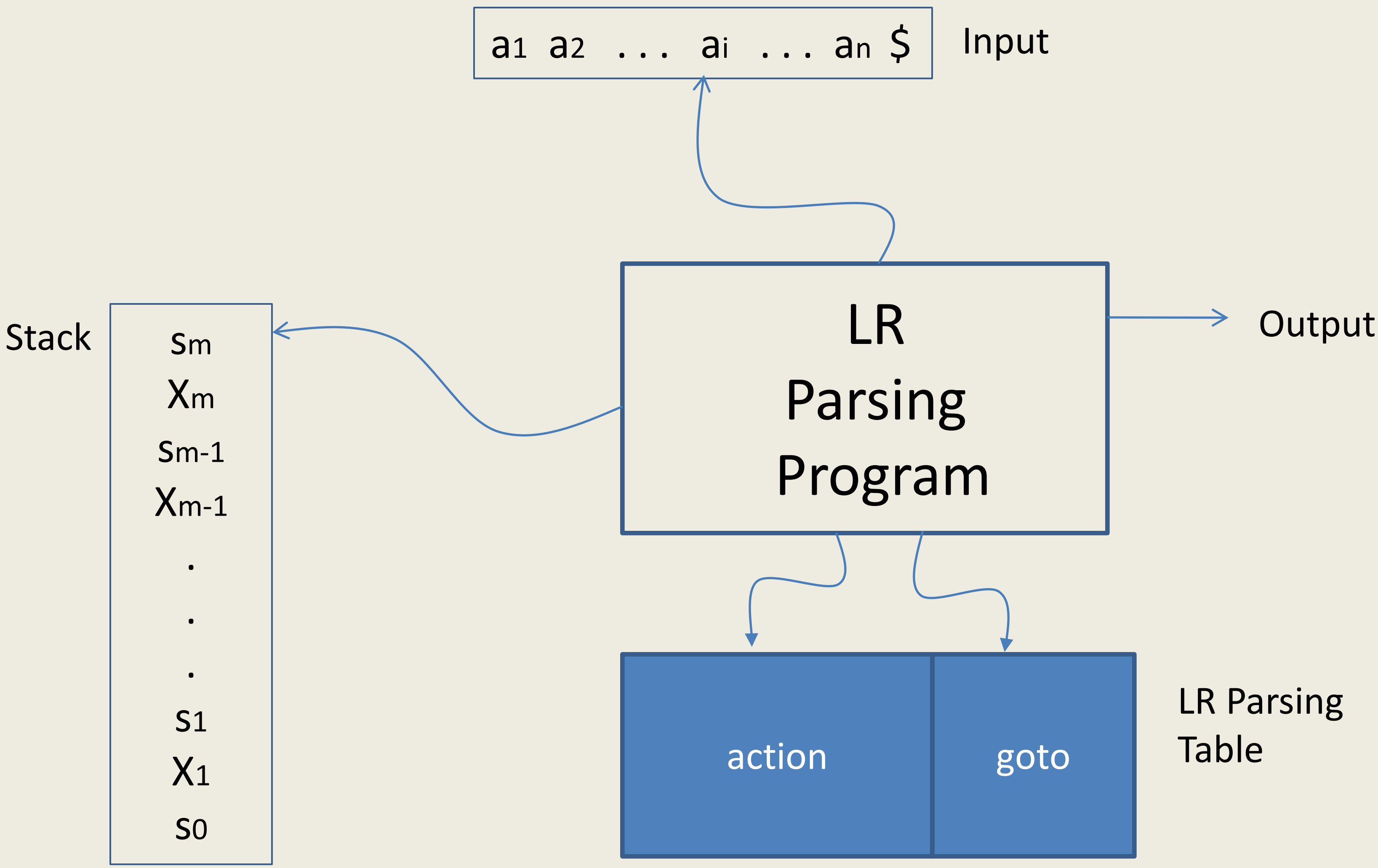
- Based on Bottom-up approach.
- Significance of the name:

  L - Left-to-right scanning of input

  R - Rightmost derivation in Reverse

  (1) - Number of input symbol(s) to make parsing decision
- It can be constructed to recognize virtually all constructs for which CFGs can be written.
- The class of grammar that can be parsed is a proper superset of the class of grammar for other parsers.

LR(1) Parser



a1 a2 . . . ai . . . an $
Input
Stack
sm
Xm
sm-1
Xm-1
.
.
.
s1
X1
s0
LR Parsing Program
Output
action
goto
LR Parsing Table

Configuration of LR(1) Parser

A Configuration of an LR parser is given as:

$$(s_0 \, X_1 \, s_1 \; \ldots \; X_m \, s_m, \;\; a_i \, a_{i+1} \; \ldots \; a_n \, \$ \,)$$

The above configuration represents the right-sentential form:

$$X_1 \;\; X_2 \; \ldots \; X_m \;\; a_i \, a_{i+1} \; \ldots \; a_n$$

Working of LR(1) Parser

The parser determines

s_m, the state currently on TOS

and a_i, the current input symbol

The parser refers action[s_m, a_i]:

1. If action[s_m, a_i] = "shift s_n"

   Shift action is taken and new configuration is

   $(s_0\ X_1\ s_1\ \ldots\ X_m\ s_m\ a_i\ s_n,\ \ a_{i+1}\ \ldots\ a_n\ \$)$

Working of LR(1) Parser

The parser determines

s_m, the state currently on TOS

and a_i, the current input symbol

The parser refers action[s_m, a_i]:

2. If action[s_m, a_i] = “reduce $A \rightarrow \beta$”

   Reduce action is taken and new configuration is

   $(s_0\ X_1\ s_1\ \ldots\ X_{m-r}\ s_{m-r}\ A\ s_p,\ \ a_i\ a_{i+1}\ \ldots\ a_n\ \$\ )$

   where $s_p = goto[s_{m-r}, A]$

   and r is the length of β

Working of LR(1) Parser

The parser determines

s_m, the state currently on TOS

and a_i, the current input symbol

The parser refers action[s_m, a_i]:

3. If action[s_m, a_i] = “accept”

   Parsing is completed successfully.

4. If action[s_m, a_i] = “error”

   The Parser has discovered a error.

Working of LR(1) Parser

Example:

(1) E → E + T

(2) E → T

(3) T → T * F

(4) T → F

(5) F → (E)

(6) F → id

Stack	Input	Action
0	id+id*id$	s5

State	action						goto		
	id	+	*	(	)	$	E	T	F
0	s5			s4			1	2	3
1		s6				A			
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			

Working of LR(1) Parser

Example:

(1) $E \rightarrow E + T$

(2) $E \rightarrow T$

(3) $T \rightarrow T * F$

(4) $T \rightarrow F$

(5) $F \rightarrow (E)$

(6) $F \rightarrow id$

Stack	Input	Action
0	id+id*id$	s5
0id5	+id*id$	r6

State	action						goto		
	id	+	*	(	)	$	E	T	F
0	s5			s4			1	2	3
1		s6				A			
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			

Working of LR(1) Parser

Example:

(1) E → E + T

(2) E → T

(3) T → T * F

(4) T → F

(5) F → (E)

(6) F → id

Stack	Input	Action
0	id+id*id$	s5
0id5	+id*id$	r6
0F3	+id*id$	r4

State	action						goto		
	id	+	*	(	)	$	E	T	F
0	s5			s4			1	2	3
1		s6				A			
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			

Working of LR(1) Parser

Example:

(1) E → E + T

(2) E → T

(3) T → T * F

(4) T → F

(5) F → (E)

(6) F → id

Stack	Input	Action
0	id+id*id$	s5
0id5	+id*id$	r6
0F3	+id*id$	r4
0T2	+id*id$	r2
0E1	+id*id$	s6
0E1+6	id*id$	s5
0E1+6id5	*id$	r6
0E1+6F3	*id$	r4
0E1+6T9	*id$	s7
0E1+6T9*7	id$	s5
0E1+6T9*7id5	$	r6
0E1+6T9*7F10	$	r3
0E1+6T9	$	r1
0E1	$	Accept

State	action						goto		
	id	+	*	(	)	$	E	T	F
0	s5			s4			1	2	3
1		s6				A			
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			

Construction of LR(1) Parsing Table

There are three methods:

1. Simple LR (SLR)
2. Lookahead LR (LALR)
3. Canonical LR (CLR or LR)

Construction of LR(1) Parsing Table

There are three methods:

Power ↓ Complex ↓

1. Simple LR (SLR)
2. Lookahead LR (LALR)
3. Canonical LR (CLR or LR)

Construction of LR(1) Parsing Table

Item (LR(0) Item) :

An item of a grammar G is a production of G with a dot at some position of the right-side of the production.

Construction of LR(1) Parsing Table

Item (LR(0) Item) :

An item of a grammar G is a production of G with a dot at some position of the right-side of the production.

Example:

A production $A \rightarrow XYZ$ yields the following:

$A \rightarrow .XYZ$

$A \rightarrow X.YZ$

$A \rightarrow XY.Z$

$A \rightarrow XYZ.$

Construction of LR(1) Parsing Table

The closure operation:

If I is a set of item for a grammar G, then closure(I) is the set of items constructed from I by the following rules:

1. Initially, every item in I is added to closure(I).
2. If $A \rightarrow \alpha.B\beta$ is in closure(I) and $B \rightarrow \gamma$ is a production then add the item $B \rightarrow .\gamma$ to closure(I) if it is not already there.

   Apply this rule until no more new items can be added to closure(I).

Construction of LR(1) Parsing Table

The closure operation:

Example:

$E \to E + T$

$E \to T$

$T \to T * F$

$T \to F$

$F \to (E)$

$F \to id$

If I = $\{E \to E.+T \, , \, E \to .T \}$

Closure(I) = ?

Construction of LR(1) Parsing Table

The closure operation:

Example:

$E \rightarrow E + T$

$E \rightarrow T$

$T \rightarrow T * F$

$T \rightarrow F$

$F \rightarrow (E)$

$F \rightarrow id$

If I = {$E \rightarrow E.+T$, $E \rightarrow .T$ }

closure(I) = { $E \rightarrow E.+T$,

$E \rightarrow .T$,

}

Construction of LR(1) Parsing Table

The closure operation:

Example:

$E \rightarrow E + T$

$E \rightarrow T$

$T \rightarrow T * F$

$T \rightarrow F$

$F \rightarrow (E)$

$F \rightarrow id$

If I = $\{E \rightarrow E.+T \text{ , } E \rightarrow .T\}$

closure(I) = { $E \rightarrow E.+T$,

$E \rightarrow .T$,

$T \rightarrow .T*F$,

$T \rightarrow .F$,

}

Construction of LR(1) Parsing Table

The closure operation:

Example:

$E \rightarrow E + T$

$E \rightarrow T$

$T \rightarrow T * F$

$T \rightarrow F$

$F \rightarrow (E)$

$F \rightarrow id$

If I = $\{E \rightarrow E.+T , E \rightarrow .T \}$

closure(I) = $\{ E \rightarrow E.+T ,$

$E \rightarrow .T ,$

$T \rightarrow .T*F ,$

$T \rightarrow .F ,$

$F \rightarrow .(E) ,$

$F \rightarrow .id \}$

Construction of LR(1) Parsing Table

The goto operation:

If I is a set of item for a grammar G and X is a grammar symbol, then goto(I, X) is defined to be the closure of all items $A \rightarrow \alpha X.\beta$ such that $A \rightarrow \alpha.X\beta$ is in I.

Construction of LR(1) Parsing Table

The goto operation:

Example:

$E \rightarrow E + T$

$E \rightarrow T$

$T \rightarrow T * F$

$T \rightarrow F$

$F \rightarrow (E)$

$F \rightarrow id$

If I = $\{E \rightarrow E.+T\ ,\ T \rightarrow T*.F\ \}$

goto(I, E) = ?

Construction of LR(1) Parsing Table

The goto operation:

Example:

$E \rightarrow E + T$

$E \rightarrow T$

$T \rightarrow T * F$

$T \rightarrow F$

$F \rightarrow (E)$

$F \rightarrow id$

If I = $\{E \rightarrow E.+T , T \rightarrow T*.F \}$

goto(I, E) = ϕ

Construction of LR(1) Parsing Table

The goto operation:

Example:

$E \rightarrow E + T$

$E \rightarrow T$

$T \rightarrow T * F$

$T \rightarrow F$

$F \rightarrow (E)$

$F \rightarrow id$

If I = $\{E \rightarrow E.+T , T \rightarrow T*.F \}$

goto(I, +) = ?

Construction of LR(1) Parsing Table

The goto operation:

Example:

$E \rightarrow E + T$

$E \rightarrow T$

$T \rightarrow T * F$

$T \rightarrow F$

$F \rightarrow (E)$

$F \rightarrow id$

If I = { $E \rightarrow E.+T$, $T \rightarrow T*.F$ }

goto(I, +) = { $E \rightarrow E+.T$,

}

Construction of LR(1) Parsing Table

The goto operation:

Example:

$E \rightarrow E + T$

$E \rightarrow T$

$T \rightarrow T * F$

$T \rightarrow F$

$F \rightarrow (E)$

$F \rightarrow id$

If I = $\{E \rightarrow E.+T \text{ , } T \rightarrow T*.F \}$

goto(I, +) = $\{ E \rightarrow E+.T \text{ ,}$

$T \rightarrow .T*F \text{ ,}$

$T \rightarrow .F \text{ ,}$

$F \rightarrow .(E) \text{ ,}$

$F \rightarrow .id \}$

Construction of LR(1) Parsing Table

Steps:

1. If input grammar is G with start symbol S, then make augmented grammar G’ with a new start symbol S’ and add a production S’ $\rightarrow$ S .

Construction of LR(1) Parsing Table

Steps:

1. If input grammar is G with start symbol S, then make augmented grammar G' with a new start symbol S' and add a production $S' \rightarrow S$.
2. Construct C = $\{I_0, I_1, \dots, I_n\}$, the collection of sets for items for G'.

Construction of LR(1) Parsing Table

Steps:

1. If input grammar is G with start symbol S, then make augmented grammar G' with a new start symbol S' and add a production $S' \rightarrow S$.
2. Construct C = $\{I_0, I_1, \ldots, I_n\}$, the collection of sets for items for G'.

   C = $\{I_0\}$, where I_0 = closure($\{S' \rightarrow .S\}$)

   repeat

   for each set of items I in C and each grammar symbol X such that goto(I, X) is non-empty and not in C do

   add goto(I, X) to C

   until no more sets of items can be added to C.

Construction of LR(1) Parsing Table

Example:

G : $E \rightarrow E + T$

$E \rightarrow T$

$T \rightarrow T * F$

$T \rightarrow F$

$F \rightarrow (E)$

$F \rightarrow id$

G' : $E' \rightarrow E$

$E \rightarrow E + T$

$E \rightarrow T$

$T \rightarrow T * F$

$T \rightarrow F$

$F \rightarrow (E)$

$F \rightarrow id$

Construction of LR(1) Parsing Table

I_0

$E' \rightarrow .E$

$E \rightarrow .E + T$

$E \rightarrow .T$

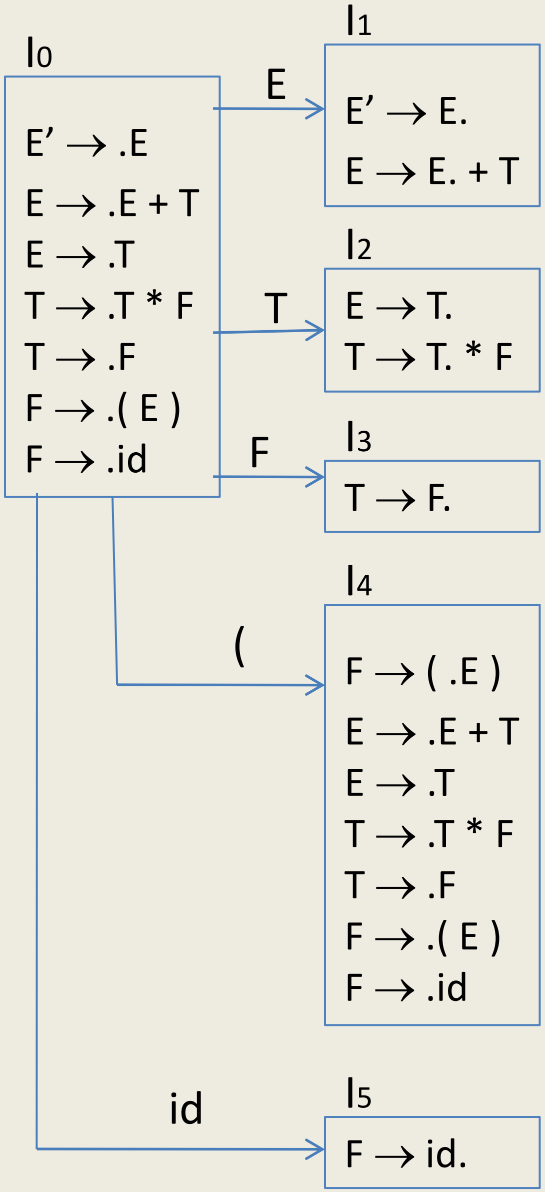
$T \rightarrow .T * F$

$T \rightarrow .F$

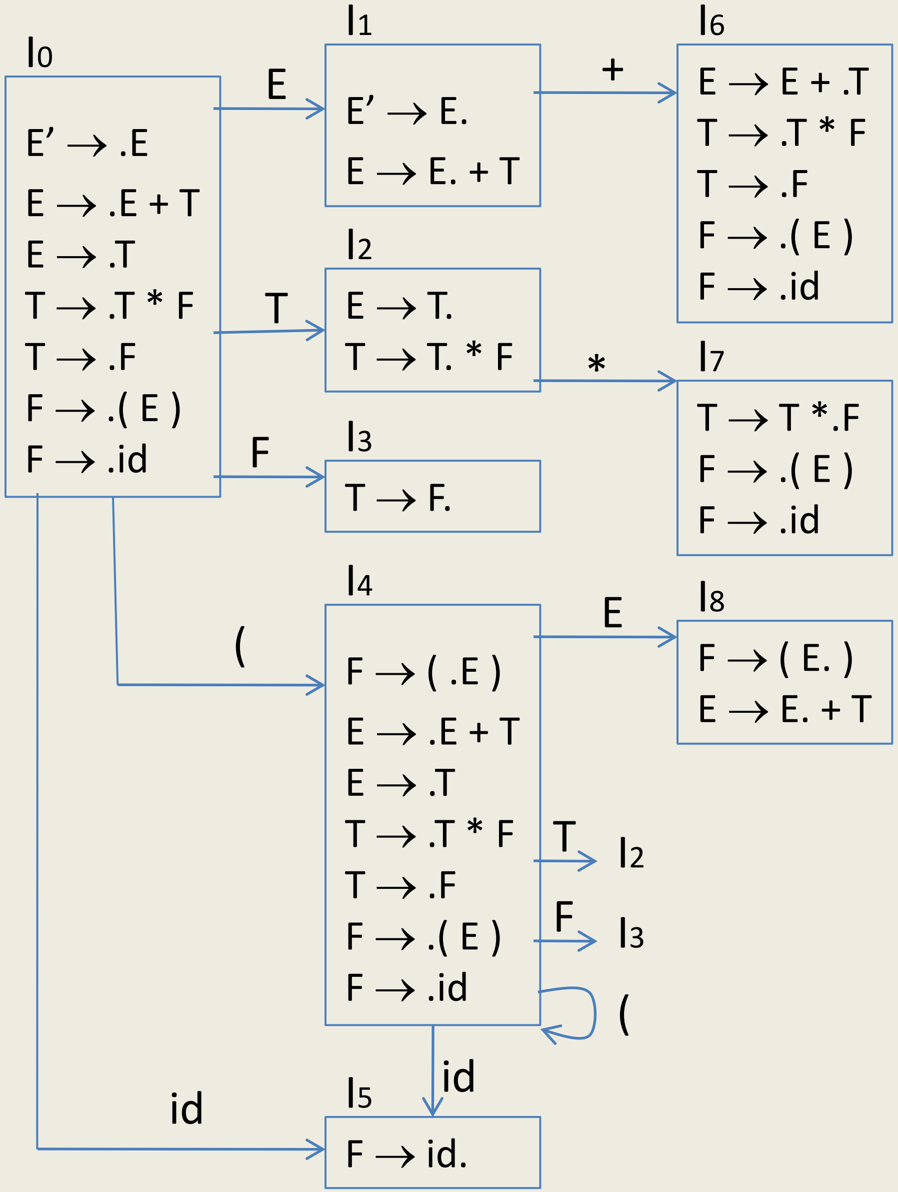
$F \rightarrow .(E)$

$F \rightarrow .id$

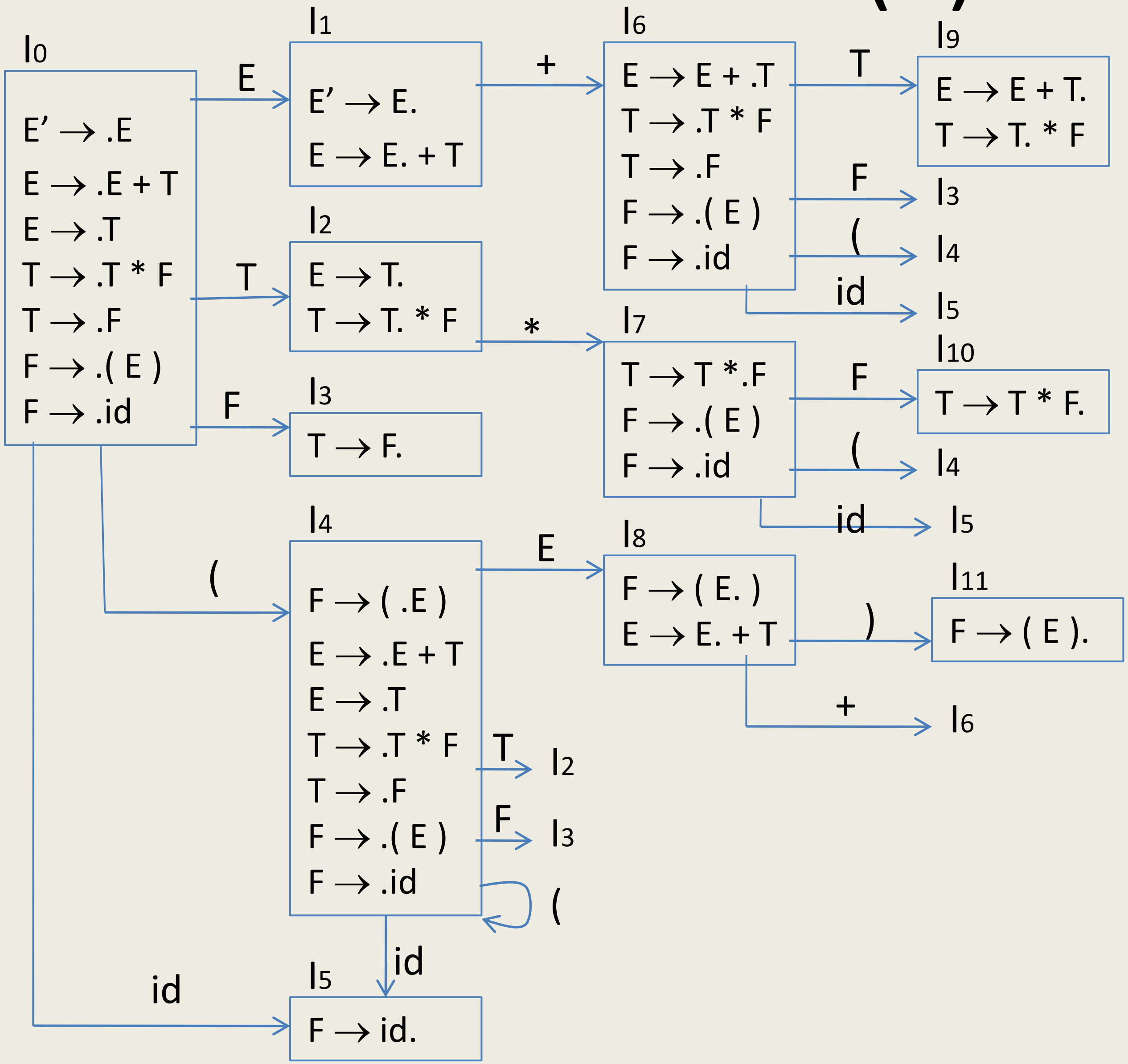
Construction of LR(1) Parsing Table



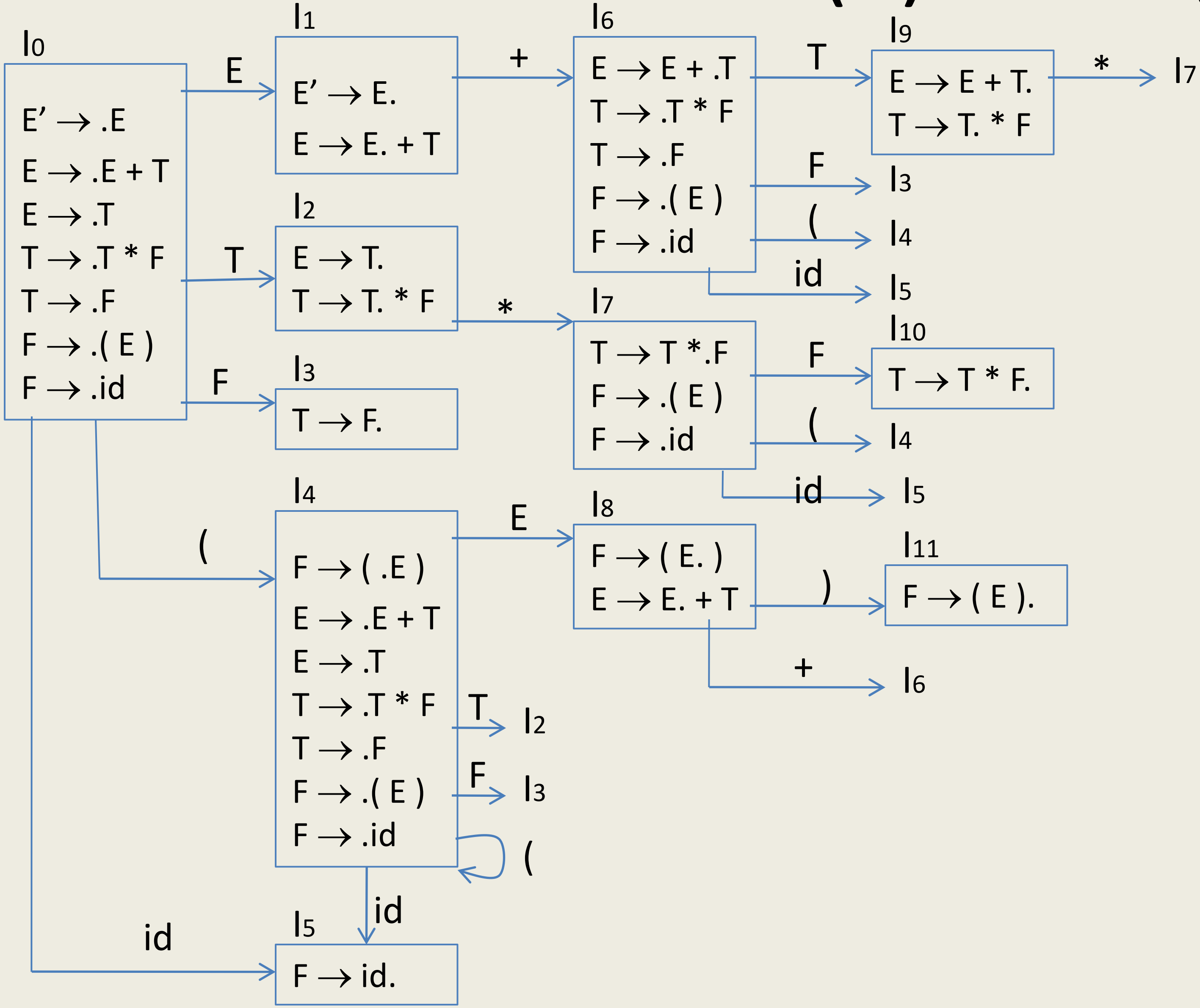
Construction of LR(1) Parsing Table



Construction of LR(1) Parsing Table



Construction of LR(1) Parsing Table



Construction of LR(1) Parsing Table

Steps:

1. If input grammar is G with start symbol S, then make augmented grammar G’ with a new start symbol S’ and add a production S’ $\rightarrow$ S .
2. Construct C = $\{I_0, I_1, \ldots, I_n\}$, the collection of sets for items for G’.

Construction of LR(1) Parsing Table

Steps:

1. If input grammar is G with start symbol S, then make augmented grammar G' with a new start symbol S' and add a production $S' \rightarrow S$.
2. Construct C = {I_0, I_1, … , I_n}, the collection of sets for items for G'.
3. State i is constructed from I_i. The parsing actions for state i are as follows:

a) If $A \rightarrow \alpha.a\beta$ is in I_i and goto(I_i, a) = I_j, then set **action[i, a] = "sj"**.

b) If $A \rightarrow \alpha.$ is in I_i then set **action[i, b] = "rk"**, for all b in Follow(A) and k is the production number of the production $A \rightarrow \alpha$.

c) If $S' \rightarrow S.$ is in I_i then set **action[i, $] = "A".**

d) If goto(I_i, A) = I_j, then **goto[i, A] = "j".**

Construction of LR(1) Parsing Table

Example:

(0) $E' \rightarrow E$

(1) $E \rightarrow E + T$

(2) $E \rightarrow T$

(3) $T \rightarrow T * F$

(4) $T \rightarrow F$

(5) $F \rightarrow (E)$

(6) $F \rightarrow id$

First (E) = { (, id }

First (T) = { (, id }

First (F) = { (, id }

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

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

Follow(F) = { +,), $, * }

Construction of LR(1) Parsing Table

Example:

(0) E' → E

(1) E → E + T

(2) E → T

(3) T → T * F

(4) T → F

(5) F → (E)

(6) F → id

First (E) = { (, id }

First (T) = { (, id }

First (F) = { (, id }

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

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

Follow(F) = { +,), $, * }

State	action						goto		
	id	+	*	(	)	$	E	T	F
0							1	2	3
1									
2									
3									
4							8	2	3
5									
6								9	3
7									10
8									
9									
10									
11									

Construction of LR(1) Parsing Table

Example:

(0) E' → E

(1) E → E + T

(2) E → T

(3) T → T * F

(4) T → F

(5) F → (E)

(6) F → id

First (E) = { (, id }

First (T) = { (, id }

First (F) = { (, id }

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

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

Follow(F) = { +,), $, * }

State	action						goto		
	id	+	*	(	)	$	E	T	F
0	s5			s4			1	2	3
1		s6							
2			s7						
3									
4	s5			s4			8	2	3
5									
6	s5			s4				9	3
7	s5			s4					10
8		s6			s11				
9			s7						
10									
11									

Construction of LR(1) Parsing Table

Example:

(0) E' → E

(1) E → E + T

(2) E → T

(3) T → T * F

(4) T → F

(5) F → (E)

(6) F → id

First (E) = { (, id }

First (T) = { (, id }

First (F) = { (, id }

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

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

Follow(F) = { +,), $, * }

State	action						goto		
	id	+	*	(	)	$	E	T	F
0	s5			s4			1	2	3
1		s6				A			
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			