LALR(1) Parsers

In LALR parsing we generate LR(1) item sets and combine like cores. Remember that LR(1) items have the form

$$[\, A \to \alpha \bullet \beta,\ \mu \,]$$

where the first component is a marked production, $A \to \alpha \bullet \beta$, called the ***core*** of the item and μ is a ***lookahead*** character.

LALR Parsing Functions

1. Obtain the LR(1) item sets, $\{ C_0, C_1, \ldots C_m \}$

2. Merge all sets that have the same core yielding, $\{ M_0, M_1, \ldots, M_n \}$, $n \le m$

3. Next State Transition Table
   Let M_p be the new merged item set, $M_p = C_{j1} \cup C_{j2} \cup \ldots \cup C_{jk}$. Each item set C_{jc} has the same core. The next state transition entry from M_p to M_q on X can be obtained from the next state transition entries of the LR(1) machine.

4. Feed the new item sets and next state transitions into the LR(1) Constructor.

Let's look at the LR(1) item sets for the previous grammar. From these item we obtain the following LALR(1) item sets by combining like cores from the original LR(1) item sets.

Remember the following grammar:

0.	$G \to S$
1.	$S \to E = E$
2.	$\quad\mid f$
3.	$E \to T$
4.	$\quad\mid E + T$
5.	$T \to f$
6.	$\quad\mid T * f$

and the LR(1) item sets for that grammar

State	**Basis Set**	**Closure Set**	**Next State or Reduce**
0	$[\, G \to \bullet S, \lambda \,]$		17
		$[\, S \to \bullet E = E, \lambda \,]$	1
		$[\, S \to \bullet f, \lambda \,]$	3
		$[\, E \to \bullet T, =:+ \,]$	2
		$[\, T \to \bullet T * f, =:+:* \,]$	2
		$[\, T \to \bullet f, =:+:* \,]$	3
		$[\, E \to \bullet E + T, =:+ \,]$	1

1	[S → E • = E, λ]		4
	[E → E • + T, =:+]		5
2	[E → T •, =:+]		R3
	[T → T • * f, =:+:*]		11
3	[S → f •, λ]		R2
	[T → f •, =:+:*]		R5
4	[S → E = • E, λ]		6
		[E → • T, λ:+]	7
		[T → • f, λ:+:*]	8
		[T → • T * f, λ:+:*]	7
		[E → • E + T, λ:+]	6
5	[E → E + • T, =:+]		10
		[T → • f, =:+:*]	9
		[T → • T * F, =:+:*]	10
6	[S → E = E •, λ]		R1
	[E → E • + T, λ:+]		13
7	[E → T •, λ:+]		R3
	[T → T • * f, λ:+:*]		15
8	[T → f •, =:+:*]		R5
9	[T → f •, =:+:*]		R5
10	[E → E + T •, =:+]		R4
	[T → T • * f, =:+:*]		11
11	[T → T * • f, =:+:*]		12
12	[T → T * f •, =:+:*]		R6
13	[E → E + • T, λ:+]		14
		[T → • f, λ:+:*]	8
		[T → • T * f, λ:+:*]	14
14	[E → E + T •, λ:+]		R4
	[T → T • * f, λ:+:*]		15
15	[T → T * • f, λ:+:*]		16
16	[T → T * f •, λ:+:*]		R6
17	[G → S •, λ]		A

Now we will look for common core elements to combine item set or states in the LR(1) machine to obtain the item sets or states in an LALR(1) machine.

States with a common core are

LALR(1) States	LR(1) States
[0]	
[1]	
[2]	
[3]	
[4]	
[5]	
[6]	
[7]	
[8]	
[9]	
[10]	
[11]	

Click here for all of the LALR(1) combined states.

Combining the LR(1) states yields the following LALR(1) item sets or states

State	Basic Set	Closure Set	Next State or Reduce
[0]			
[1]			
[2]			
[3]			
[4]			
[5]			
[6]			
[7]			

Click here for the complete LALR(1) item sets.

State	Basic Set	Closure Set	Next State or Reduce
[0]	[G → • S, λ]		[11]
		[S → • E = E, λ]	[1]
		[S → • f, λ]	[3]
		[E → • T, =:+]	[2]
		[T → • T * f, =:+:*]	[2]
		[T → • f, =:+:*]	[3]
		[E → • E + T, =:+]	[1]
[1]	[S → E • = E, λ]		[4]
	[E → E • + T, =:+]		[5]
[2]	[E → T •, λ:=:+]		R3
	[T → T • * f, λ:=:+:*]		[9]
[3]	[S → f •, λ]		R2
	[T → f •, =:+:*]		R5
[4]	[S → E = • E, λ]		[6]
		[E → • T, λ:+]	[2]
		[T → • f, λ:+:*]	[7]
		[T → • T * f, λ:+:*]	[2]
		[E → • E + T, λ:+]	[6]

Applying the LR(1) Constructor to the above partial item sets we have

Action Function F

State	f	=	+	*	λ
[0]					
[1]					
[2]					
[3]					
[4]					

Click here for the complete Action Table.

State	Basic Set	Closure Set	Next State or Reduce
[0]	[G → • S, λ]		[11]
		[S → • E = E, λ]	[1]
		[S → • f, λ]	[3]
		[E → • T, =:+]	[2]
		[T → • T * f, =:+:*]	[2]
		[T → • f, =:+:*]	[3]
		[E → • E + T, =:+]	[1]
[1]	[S → E • = E, λ]		[4]
	[E → E • + T, =:+]		[5]
[2]	[E → T •, λ:=:+]		R3
	[T → T • * f, λ:=:+:*]		[9]
[3]	[S → f •, λ]		R2
	[T → f •, =:+:*]		R5
[4]	[S → E = • E, λ]		[6]
		[E → • T, λ:+]	[2]
		[T → • f, λ:+:*]	[7]
		[T → • T * f, λ:+:*]	[2]
		[E → • E + T, λ:+]	[6]

The next state or goto function is

Next State Function G

State	S	E	T	f	=	+	*
[0]							
[1]							
[2]							
[3]							
[4]							

Click here for the complete Next State Table.

Repeating the LALR(1) parse tables we have

	Action Function F					**Next State Function G**						
	f	=	+	*	λ	S	E	T	f	=	+	*
[0]	S	.	.	.	.	[11]	[1]	[2]	[3]	.	.	.
[1]	.	S	S	.	.	.	.	.	.	[4]	[5]	.
[2]	.	R3	R3	S	R3	.	.	.	.	.	.	[9]
[3]	.	R5	R5	R5	R2	.	.	.	.	.	.	.
[4]	S	.	.	.	.	.	[6]	[2]	[7]	.	.	.
[5]	S	.	.	.	.	.	.	[8]	[7]	.	.	.
[6]	.	.	S	.	R1	.	.	.	.	.	[5]	.
[7]	.	R5	R5	R5	R5	.	.	.	.	.	.	.
[8]	.	R4	R4	S	R4	.	.	.	.	.	.	[9]
[9]	S	.	.	.	.	.	.	.	[10]	.	.	.
[10]	.	R6	R6	R6	R6	.	.	.	.	.	.	.
[11]	.	.	.	.	A	.	.	.	.	.	.	.

Using the above LALR(1) parse tables, the LALR parse configuration for f * f = f + f yields

N	**S**	**α**	**A/π**
-	0	f * f = f + f	-

Grammar

0. $G \rightarrow S$
1. $S \rightarrow E = E$
2. $\quad\mid f$
3. $E \rightarrow T$
4. $\quad\mid E + T$
5. $T \rightarrow f$
6. $\quad\mid T * f$

Click here for the complete parse sequence.

ANSWERS

The combined states are

LALR(1) States	LR(1) States
[0]	0
[1]	1
[2]	2, 7
[3]	3
[4]	4
[5]	5, 13
[6]	6
[7]	8, 9
[8]	10, 14
[9]	11, 15
[10]	12, 16
[11]	17

Return

The complete LALR(1) item sets are

State	Basic Set	Closure Set	Next State or Reduce
[0]	[G → • S, λ]		[11]
		[S → • E = E, λ]	[1]
		[S → • f, λ]	[3]
		[E → • T, =:+]	[2]
		[T → • T * f, =:+:*]	[2]
		[T → • f, =:+:*]	[3]
		[E → • E + T, =:+]	[1]
[1]	[S → E • = E, λ]		[4]
	[E → E • + T, =:+]		[5]
[2]	[E → T •, λ:=:+]		R3
	[T → T • * f, λ:=:+:*]		[9]
[3]	[S → f •, λ]		R2
	[T → f •, =:+:*]		R5
[4]	[S → E = • E, λ]		[6]
		[E → • T, λ:+]	[2]
		[T → • f, λ:+:*]	[7]
		[T → • T * f, λ:+:*]	[2]
		[E → • E + T, λ:+]	[6]
[5]	[E → E + • T, λ:=:+]		[8]
		[T → • f, λ:=:+:*]	[7]
		[T → • T * F, λ:=:+:*]	[8]
[6]	[S → E = E •, λ]		R1
	[E → E • + T, λ:+]		[5]

[7]	[T → f •, λ:=:+:*]	R5
[8]	[E → E + T •, λ:=:+] [T → T • * f, λ:=:+:*]	R4 [9]
[9]	[T → T * • f, λ:=:+:*]	[10]
[10]	[T → T * f •, λ:=:+:*]	R6
[11]	[G → S •, λ]	A

Return

Action Function F

State	f	=	+	*	λ
[0]	S	.	.	.	.
[1]	.	S	S	.	.
[2]	.	R3	R3	S	R3
[3]	.	R5	R5	R5	R2
[4]	S	.	.	.	.
[5]	S	.	.	.	.
[6]	.	.	S	.	R1
[7]	.	R5	R5	R5	R5
[8]	.	R4	R4	S	R4
[9]	S	.	.	.	.
[10]	.	R6	R6	R6	R6
[11]	.	.	.	.	A

Return

Next State Function G

State	S	E	T	f	=	+	*
[0]	[11]	[1]	[2]	[3]	.	.	.
[1]	.	.	.	.	[4]	[5]	.
[2]	.	.	.	.	.	.	[9]
[3]	.	.	.	.	.	.	.
[4]	.	[6]	[2]	[7]	.	.	.
[5]	.	.	[8]	[7]	.	.	.
[6]	.	.	.	.	.	[5]	.
[7]	.	.	.	.	.	.	.
[8]	.	.	.	.	.	.	[9]
[9]	.	.	.	[10]	.	.	.
[10]	.	.	.	.	.	.	.
[11]	.	.	.	.	.	.	.

Return

The complete LALR parse sequence.

N	**S**	α	**A**	π
-	0	f * f = f + f	-	-
3	~~3~~ 0	* f = f + f	S	-
2	2 0	* f = f + f	R	5
9	9 2 0	f = f + f	S	-
10	~~10 9 2~~ 0	= f + f	S	-
2	~~2~~ 0	= f + f	R	6
1	1 0	= f + f	R	3
4	4 1 0	f + f	S	-
7	~~7~~ 4 1 0	+ f	S	-
2	~~2~~ 4 1 0	+ f	R	5
6	6 4 1 0	+ f	R	3
5	5 6 4 1 0	f	S	-
7	~~7~~ 5 6 4 1 0	λ	S	-
8	~~8 5 6 4~~ 1 0	λ	R	5
6	~~6 4 1~~ 0	λ	R	4
11	11 0	λ	R	1
			Accept	