CS335 Assignment2

Dhruv Garg

February 14, 2024

Problem 1

(i) Explain why the given grammar is not LL(1).
The given grammar is not LL(1) because the given context-free grammar is left recursive and when the grammar is also not left factored.

Type $\rightarrow$ Type $*$ is the rule that makes the grammar left recursive hence not LL(1).

ArgList $\rightarrow$ Type id, ArgList and ArgList $\rightarrow$ Type id are the two rules which make it, not LL(1) because looking the lookahead symbol does not tell with guarantee which rule to select.

(ii) Perform required transformations on the grammar to make it LL(1). You may introduce new nonterminals. Show your transformed grammar.
The two main transformations are removing left recursion and performing left factoring. After applying the above transformations we get the grammar defined below:

Function	$\rightarrow$ Type id (Arguments)
Type	$\rightarrow$ id Type'
Type'	$\rightarrow$ * Type'
Type'	$\rightarrow \epsilon$
Arguments	$\rightarrow$ ArgList
Arguments	$\rightarrow \epsilon$
ArgList	$\rightarrow$ Type id Arghelp
Arghelp	$\rightarrow$, ArgList
Arghelp	$\rightarrow \epsilon$

(iii) Show FIRST and FOLLOW set for the nonterminals in your transformed grammar.

NT/SETS	FIRST	FOLLOW
Function	id	$
Type	id	id
Type'	* ϵ	id
Arguements	id ϵ	)
ArgList	id	)
Arghelp	, ϵ	)

(iv) Show the predictive LL(1) parsing table for your transformed grammar. You do not need to show the working of the parser with an example input string.

State	id	(	)	*	,	$
Function	Function $\rightarrow$ Type id (Arguments)					
Type	Type $\rightarrow$ id Type'					
Type'	Type' $\rightarrow \epsilon$			Type' $\rightarrow *$ Type'		
Arguments	Arguments $\rightarrow$ ArgList		Arguments $\rightarrow \epsilon$			
ArgList	ArgList $\rightarrow$ Type id Arghelp					
Arghelp			Arghelp $\rightarrow \epsilon$		Arghelp $\rightarrow$, ArgList	

Problem 2

(i) FIRST and FOLLOW sets of grammar

NT/SETS	FIRST	FOLLOW
S'	q,a,s,t	$
S	q,a,s,t	$
L	a,s,t	p,r,t
M	t	$,b

(ii) LR(0) Canonical Collection

State	Closure/GOTO	Productions
I_0	Closure($S' \to \cdot S$)	{ S' $\to \cdot S, S \to \cdot LM, S \to \cdot Lp, S \to \cdot qLr, S \to \cdot sr, S \to \cdot qsp, L \to \cdot aMb, L \to \cdot s, L \to \cdot t$}
I_1	GOTO(I_0, S)	S' $\to S\cdot$
I_2	GOTO(I_0, L)	S $\to L \cdot M, S \to L \cdot p, M \to \cdot t$
I_3	GOTO(I_0, q)	S $\to q \cdot Lr, S \to q \cdot sp, L \to \cdot aMb, L \to \cdot s, L \to \cdot t$
I_4	GOTO(I_0, s)	S $\to s \cdot r, L \to s\cdot$
I_5	GOTO(I_0, a), GOTO(I_3, a)	L $\to a \cdot Mb, M \to \cdot t$
I_6	GOTO(I_0, t), GOTO(I_3, t)	L $\to t\cdot$
I_7	GOTO(I_2, M)	S $\to LM\cdot$
I_8	GOTO(I_2, p)	S $\to Lp\cdot$
I_9	GOTO(I_2, t), GOTO(I_5, t)	M $\to t\cdot$
I_{10}	GOTO(I_3, L)	S $\to qL \cdot r$
I_{11}	GOTO(I_3, s)	S $\to qs \cdot p, L \to s\cdot$
I_{12}	GOTO(I_4, r)	S $\to sr\cdot$
I_{13}	GOTO(I_5, M)	L $\to aM \cdot b$
I_{14}	GOTO(I_{10}, r)	S $\to qLr\cdot$
I_{15}	GOTO(I_{11}, p)	S $\to qsp\cdot$
I_{16}	GOTO(I_{13}, b)	L $\to aMb\cdot$

(iii) SLR Parsing table

State	a	b	p	q	r	s	t	S	L	M	$
0	s5			s3		s4	s6	1	2		
1											Accept
2			s9				s8			7	
3	s5					s11	s6		10		
4			r7		s12/r7		r7				
5							s9			13	
6			r8		r8		r8				
7											r1
8											r2
9		r9									r9
10					s14						
11			s15/r7		r7		r7				
12											r4
13		s16									
14											r3
15											r5
16			r6		r6		r6				

Now we can conclude that the grammar is not SLR(1) because there are entries in the table where there are shift-reduce conflicts. For example, when we are $I_1$1, there is a possibility of either shifting the next letter or reducing using rule 7 (L $\to$ s). Similarly, when we are I_4, there are 2 possibilities either shift to state 4 or reduce using rule 7. So grammar is not SLR(1).

(iv) LR(0) Automaton

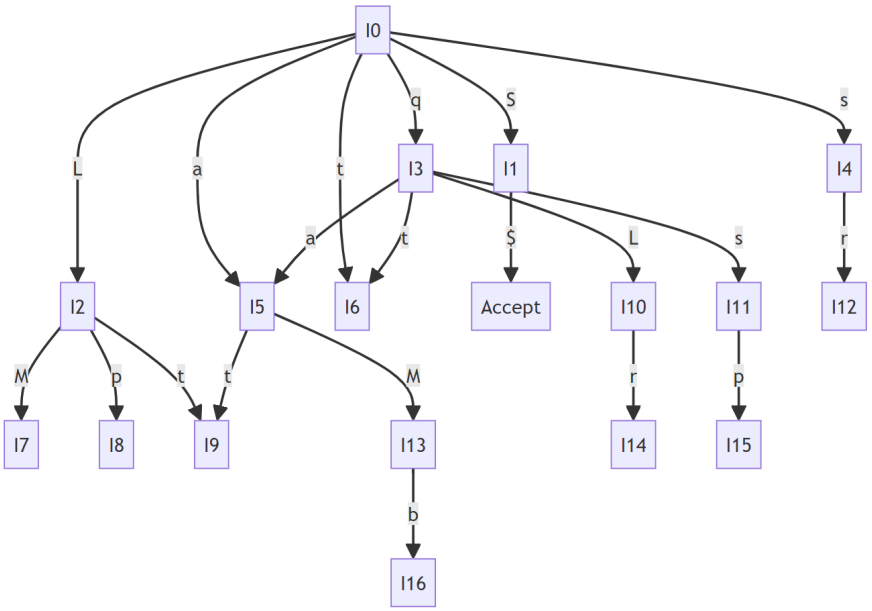

Figure 1: LR(0) Automaton

(v) LALR Collection :

First let us see LR(1) collection:

State	Closure/GOTO	Productions
I_0	Closure(S' $\rightarrow \bullet S, \$$)	{ [S' $\rightarrow \bullet S, \$$], [$S \rightarrow \bullet LM, \$$], [$S \rightarrow \bullet Lp, \$$], [$S \rightarrow \bullet qLr, \$$], [$S \rightarrow \bullet sr, \$$], [$S \rightarrow \bullet qsp, \$$], [$L \rightarrow \bullet aMb, t/p$], [$L \rightarrow \bullet s, t/p$], [$L \rightarrow \bullet t, t/p$]}
I_1	GOTO(I_0, S)	{ S' $\rightarrow S\bullet, \$$}
I_2	GOTO(I_0, L)	{ [S $\rightarrow L \bullet M, \$$], [$S \rightarrow L \bullet p, \$$], [$M \rightarrow \bullet t, \$$]}
I_3	GOTO(I_0, q)	{ [S $\rightarrow q \bullet Lr, \$$], [$S \rightarrow q \bullet sp, \$$], [$L \rightarrow \bullet aMb, r$], [$L \rightarrow \bullet s, r$], [$L \rightarrow \bullet t, r$]}
I_4	GOTO(I_0, s)	{ [S $\rightarrow s \bullet r, \$$], [$L \rightarrow s\bullet, t/p$]}
I_5	GOTO(I_0, a)	{ [L $\rightarrow a \bullet Mb, t/p$], [$M \rightarrow \bullet t, b$]}
I_6	GOTO(I_0, t)	{ [L $\rightarrow t\bullet, t/p$]}
I_7	GOTO(I_2, M)	{ [S $\rightarrow LM\bullet, \$$]}
I_8	GOTO(I_2, p)	{ [S $\rightarrow Lp\bullet, \$$]}
I_9	GOTO(I_2, t)	{ [M $\rightarrow t\bullet, \$$]}
I_{10}	GOTO(I_3, L)	{ [S $\rightarrow qL \bullet r, \$$]}
I_{11}	GOTO(I_3, s)	{ [S $\rightarrow qs \bullet p, \$$], [$L \rightarrow s\bullet, r$]}
I_{12}	GOTO(I_3, a)	{ [L $\rightarrow a \bullet Mb, r$], [$M \rightarrow \bullet t, b$]}
I_{13}	GOTO(I_3, t)	{ [L $\rightarrow t\bullet, r$]}
I_{14}	GOTO(I_4, r)	{ [S $\rightarrow sr\bullet, \$$]}
I_{15}	GOTO(I_5, M)	{ [L $\rightarrow aM \bullet b, t/p$]}
I_{16}	GOTO(I_5, t), $GOTO(I_{12}, t)$	{ [M $\rightarrow t\bullet, b$]}
I_{17}	GOTO(I_{10}, r)	{ [S $\rightarrow qLr\bullet, \$$]}
I_{18}	GOTO(I_{11}, p)	{ [S $\rightarrow qsp\bullet, \$$]}
I_{19}	GOTO(I_{12}, M)	{ [L $\rightarrow aM \bullet b, r$]}
I_{20}	GOTO(I_{15}, b)	{ [L $\rightarrow aMb\bullet, t/p$]}
I_{21}	GOTO(I_{19}, b)	{ [L $\rightarrow aMb\bullet, r$]}

Items having the same core are : $\{I_5, I_{12}\}, \{I_6, I_{13}\}, \{I_9, I_{16}\}, \{I_{15}, I_{19}\}, \{I_{20}, I_{21}\}$

New combined states are : $\{I_{5/12}\}, \{I_{6/13}\}, \{I_{9/16}\}, \{I_{15/19}\}, \{I_{20/21}\}$

Now as we have combined states, we can get LALR collection from this. Here is the LALR(1) collection:

State	Closure/GOTO	Productions
I_0	Closure(S' $\rightarrow \bullet S, \$$)	{ $[S' \rightarrow \bullet S, \$], [S \rightarrow \bullet LM, \$], [S \rightarrow \bullet Lp, \$], [S \rightarrow \bullet qLr, \$], [S \rightarrow \bullet sr, \$], [S \rightarrow \bullet qsp, \$], [L \rightarrow \bullet aMb, t/p], [L \rightarrow \bullet s, t/p], [L \rightarrow \bullet t, t/p]$ }
I_1	GOTO(I_0, S)	{ $S' \rightarrow S\bullet, \$$}
I_2	GOTO(I_0, L)	{ $[S \rightarrow L \bullet M, \$], [S \rightarrow L \bullet p, \$], [M \rightarrow \bullet t, \$]$}
I_3	GOTO(I_0, q)	{ $[S \rightarrow q \bullet Lr, \$], [S \rightarrow q \bullet sp, \$], [L \rightarrow \bullet aMb, r], [L \rightarrow \bullet s, r], [L \rightarrow \bullet t, r]$}
I_4	GOTO(I_0, s)	{ $[S \rightarrow s \bullet r, \$], [L \rightarrow s\bullet, t/p]$}
$I_{5/12}$	GOTO(I_0, a), $GOTO(I_3, a)$	{ $[L \rightarrow a \bullet Mb, t/p/r], [M \rightarrow \bullet t, b]$}
$I_{6/13}$	GOTO(I_0, t), $GOTO(I_3, t)$	{ $[L \rightarrow t\bullet, t/p/r]$}
I_7	GOTO(I_2, M)	{ $[S \rightarrow LM\bullet, \$]$}
I_8	GOTO(I_2, p)	{ $[S \rightarrow Lp\bullet, \$]$}
$I_{9/16}$	GOTO(I_2, t), $GOTO(I_{5/12}, t)$	{ $[M \rightarrow t\bullet, b/\$]$}
I_{10}	GOTO(I_3, L)	{ $[S \rightarrow qL \bullet r, \$]$}
I_{11}	GOTO(I_3, s)	{ $[S \rightarrow qs \bullet p, \$], [L \rightarrow s\bullet, r]$}
I_{14}	GOTO(I_4, r)	{ $[S \rightarrow sr\bullet, \$]$}
$I_{15/19}$	GOTO($I_{5/12}, M$)	{ $[L \rightarrow aM \bullet b, t/p/r]$}
I_{17}	GOTO(I_{10}, r)	{ $[S \rightarrow qLr\bullet, \$]$}
I_{18}	GOTO(I_{11}, p)	{ $[S \rightarrow qsp\bullet, \$]$}
$I_{20/21}$	GOTO($I_{15/19}, b$)	{ $[L \rightarrow aMb\bullet, t/p/r]$}

(v) LALR(1) Parsing table :

State	p	q	r	s	t	a	b	$	S	L	M
I_0		S3		S4	S6/13	S5/12			1	2	
I_1								Accept			
I_2	S8				S9/16						7
I_3				S11	S6/13	S5/12				10	
I_4	R7		S14		R7						
$I_{5/12}$					S9/16						15/19
$I_{6/13}$	R8		R8		R8						
I_7								R1			
I_8								R2			
$I_{9/16}$							R9	R9			
I_{10}			S17								
I_{11}	S18		R7								
I_{14}								R4			
$I_{15/19}$							S20/21				
I_{17}								R3			
I_{18}								R5			
$I_{20/21}$	R6		R6		R6						

Now since we can see from the parsing table that there are no shift-reduce or reduce-reduce conflicts, we can say that our grammar is LALR(1) grammar.

(vi) LR(1) Automaton :

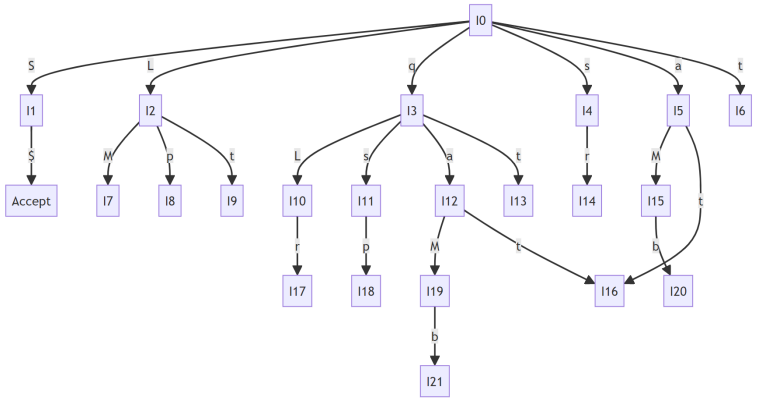

Figure 2: LR(1) Automaton

Some additional notes for problem 2:

1. The parsing table has 4 types of entries:
   (a) **Empty:** Error or the transition can not take place
   (b) **Accept:** The grammar accepts it.
   (c) **S#:** There is a shift from the input string over the stack and the state is changed to State a.
   (d) **R#:** The stack is reduced using Rule #.
   The rules are as follows:

R1: $S \rightarrow LM$
R2: $S \rightarrow Lp$
R3: $S \rightarrow qLr$
R4: $S \rightarrow sr$
R5: $S \rightarrow qsp$
R6: $L \rightarrow aMb$
R7: $L \rightarrow s$
R8: $L \rightarrow t$
R9: $M \rightarrow t$

Problem 3

I have named my files "flex.l" and "parse.y" for lexer and parser respectively (without double quotes). Let's say the test file name is "test.txt" then you need to run the following commands.

```
flex flex.l
bison -d -t parse.y
g++ parse.tab.c lex.yy.c
./a.out test.txt
```

Some important points to be noted:

1. Correct/choice tags are ignored if they are outside questions.
2. I have reported the line of the opening tag if that tag has no closing tag before an error is received.
3. If there is a stray closing tag then I have reported the line number of the same.
4. I have added some explanations in the error statements to make clear what the error is exactly about.