CS335: Assignment 2

Ayush Tharwani
170201

1 Problem 1

Grammer (G) :
$S \rightarrow (L)|a$
$L \rightarrow L, S|LS|b$
Since this is a left recursive grammar, we will derive a non-left recursive grammar G' from this. We introduce a new non-terminal L' for this.
Grammer (G') :
$S \rightarrow (L)|a$
$L \rightarrow bL'$
$L' \rightarrow, SL'|SL'|\epsilon'$
Next, we find the FIRST and FOLLOW Sets for the non terminals as:

Non-Terminal	FIRST	FOLLOW
S	{'(' , 'a'}	{\$, ',' , '(' , 'a'}
L	{'b'}	{')'}
L'	{',' , '(' , 'a' , 'ϵ'}	{')'}

Using the these Sets, we construct the Predictive Parser Table :

Non-Terminal	'a'	'b'	','	'('	')'	'\$'
S	$S \rightarrow a$		$S \rightarrow (L)$			
L		$L \rightarrow bL'$				
L'	$L' \rightarrow SL'$		$L' \rightarrow SL'$	$L' \rightarrow, SL'$	$L' \rightarrow \epsilon$	

2 Problem 2

Grammer (G) :
$S \rightarrow Lp|qLr|sr|qsp$
$L \rightarrow s$

With new start symbol S', G becomes G':
Grammer (G) :
$S' \rightarrow S$
$S \rightarrow Lp|qLr|sr|qsp$
$L \rightarrow s$

Let's compute the LR(0) Items/States of G' as follows:

$I_0 = Closure([S' \rightarrow .S]) = \{$
(0)S' $\rightarrow .S$,
(1)S $\rightarrow .Lp$,
(2)S $\rightarrow .qLr$,
(3)S $\rightarrow .sr$,
(4)S $\rightarrow .qsp$,
(5)L $\rightarrow .s$

}

$I_1 = Goto(\mathrm{I}_0, S)$ = {
$S' \rightarrow S.$
}

$I_2 = Goto(\mathrm{I}_0, L)$ = {
$S \rightarrow L.p$
}

$I_3 = Goto(\mathrm{I}_0, s)$ = {
$S \rightarrow s.r,$
L $\rightarrow s.$
}

$I_4 = Goto(\mathrm{I}_0, q)$ = {
$S \rightarrow q.Lr,$
S $\rightarrow q.sp,$
L $\rightarrow .s$
}

$I_5 = Goto(\mathrm{I}_2, p)$ = {
$S \rightarrow Lp.$
}

$I_6 = Goto(\mathrm{I}_3, r)$ = {
$S \rightarrow sr.$
}

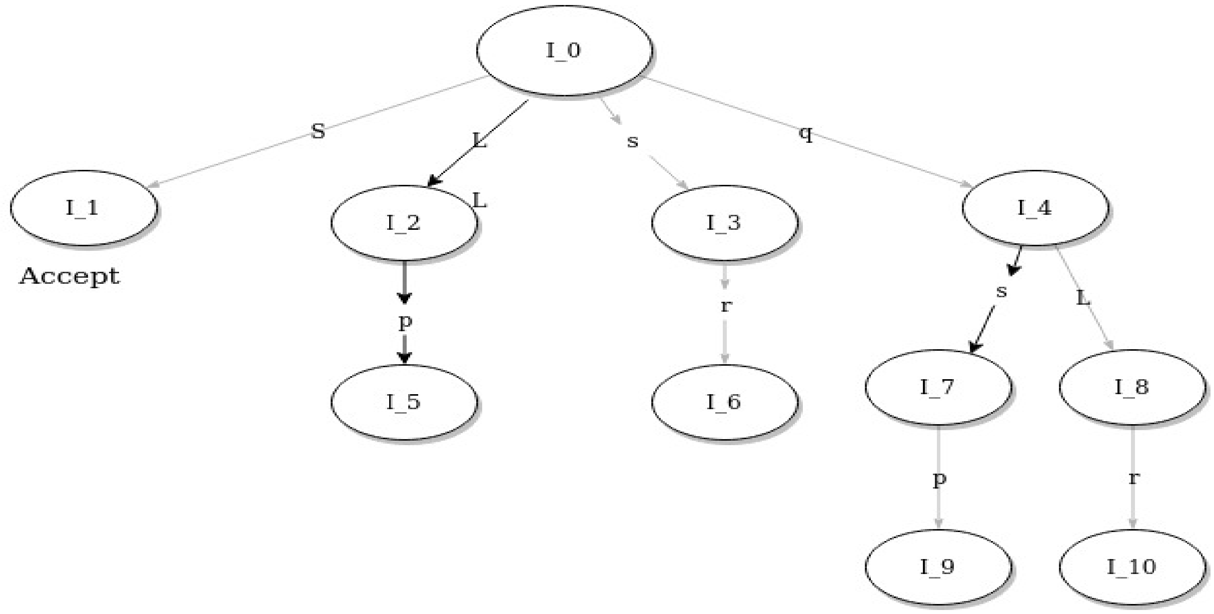
$I_7 = Goto(\mathrm{I}_4, s)$ = {
$S \rightarrow qs.p,$
L $\rightarrow s.$
}

$I_8 = Goto(\mathrm{I}_4, L)$ = {
$S \rightarrow qL.r$
}

$I_9 = Goto(\mathrm{I}_7, p)$ = {
$S \rightarrow qsp.$
}

$I_10 = Goto(\mathrm{I}_8, r)$ = {
$S \rightarrow qLr.$
}

Using the above LR(0) items, we construct the the state diagram as follows:



State	Actions					Goto	
	'p'	'q'	'r'	's'	'$'	'S'	L
0		s4		s3		1	2
1					acc		
2	s5						
3	r5		r5/s6				
4		s7					8
5					r1		
6					r3		
7	r5/s9		r5				
8			s10				
9					r4		
10					r2		

Since, there are shift-reduce conflicts at Action[3,'r'] and Action[7,'p'], the grammar is not SLR(1). Now, we will construct LR(1) items for G' in order to show that it is LALR(1).

$I_0 = Closure([S' \rightarrow .S, \$]) = \{$
  (0)$S' \rightarrow .S, \$$,
  (1) $S \rightarrow .Lp, \$$,
  (2) $S \rightarrow .qLr, \$$,
  (3) $S \rightarrow .sr, \$$,
  (4) $S \rightarrow .qsp, \$$
  (5) $L \rightarrow .s, p$
}

$I_1 = Goto(I_0, S)$ = {
  $S' \rightarrow S., \$$
}

$I_2 = Goto(I_0, L)$ = {
  $S \rightarrow L.p, \$$
}

$I_3 = Goto(I_0, q)$ = {
  $S \rightarrow q.Lr, \$$,
  $S \rightarrow q.sp, \$$,

$L \to .s, r$
}

$I_4 = Goto(\mathrm{I}_0, s)$ = {
$S \to s.r, \$,$
$L \to s., p$
}

$I_5 = Goto(\mathrm{I}_2, p)$ = {
$S \to Lp., \$,$
}

$\mathrm{I}_6 = Goto(\mathrm{I}_3, s)$ = {
$S \to qs.p, \$,$
$L \to s., r,$
}

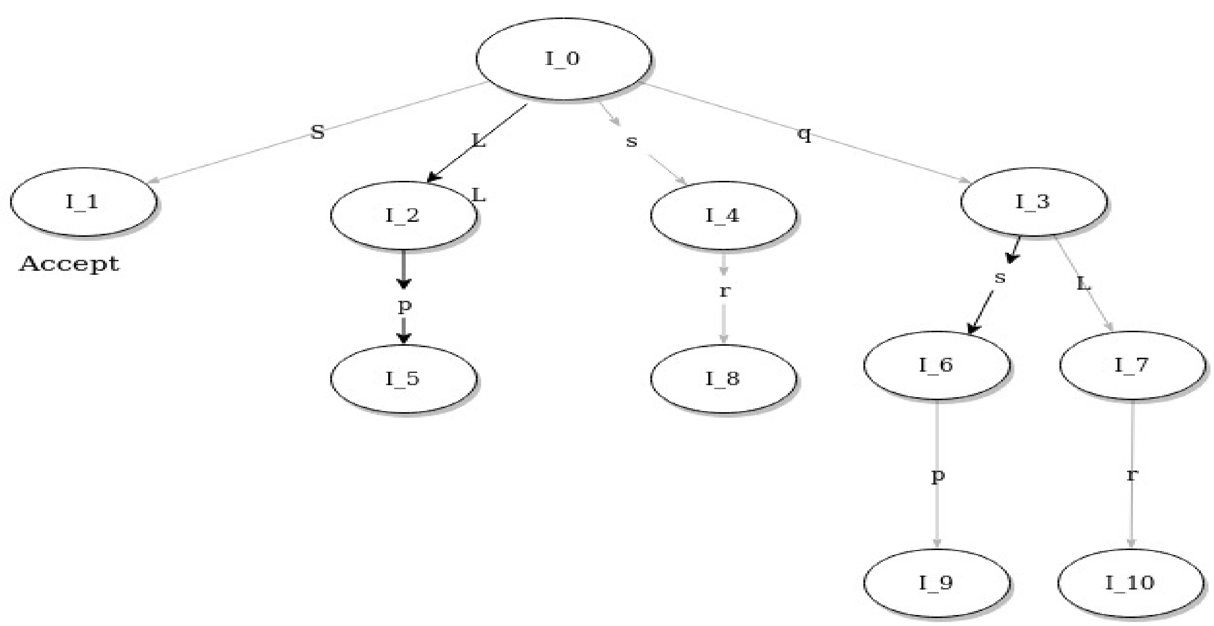
$I_7 = Goto(\mathrm{I}_3, L)$ = {
$S \to qL.r, \$,$
}

$\mathrm{I}_8 = Goto(\mathrm{I}_4, r)$ = {
$S \to sr., \$,$
}

$\mathrm{I}_9 = Goto(\mathrm{I}_6, p)$ = {
$S \to qsp., \$,$
}

$\mathrm{I}_{10} = Goto(\mathrm{I}_7, r)$ = {
$S \to qLr., \$,$
}

Using the above LR(1) items, we construct the the state diagram and LR(1) Table as follows:



State	Actions					Goto	
	'p'	'q'	'r'	's'	'$'	'S'	L
0		s3		s4		1	2
1					acc		
2	s5						
3				s6			6
4	r5		s8				
5					r1		
6	s9		r5				
7			s10				
8					r3		
9					r4		
10					r2		

There are no shift-reduce or reduce-reduce conflicts in the table. Thus, the given grammar G is LALR(1).

3 Problem 3

Given Grammer (G) :
$R \rightarrow R'|'R$
$R \rightarrow RR$
$R \rightarrow R*$
$R \rightarrow (R)$
$R \rightarrow a|b$

We augment grammar G with new start symbol S' and rule $S' \rightarrow R$. Next we compute the LR(0) Items as follows:

$\mathrm{I}_0 = Closure([\text{S'} \rightarrow .R]) = \{$
(0)S' $\rightarrow .R$,
(1) $R \rightarrow .R'|'R$,
(2) $R \rightarrow .RR$,
(3) $R \rightarrow .R*$,
(4) $R \rightarrow .(R)$,
(5) $R \rightarrow .a$
(6) $R \rightarrow .b$
}

$\mathrm{I}_1 = Goto(\mathrm{I}_0, R)$ = {
$S' \rightarrow R.$,
$R \rightarrow R.'|'R$,
$R \rightarrow R.R$,
$R \rightarrow R.*$,
$R \rightarrow .R'|'R$,
$R \rightarrow .RR$,
$R \rightarrow .R*$,
$R \rightarrow .(R)$,
$R \rightarrow .a$
$R \rightarrow .b$
}

$\mathrm{I}_2 = Goto(\mathrm{I}_0,' (')$ = {
$R \rightarrow (.R)$,
$R \rightarrow .R'|'R$,
$R \rightarrow .RR$,

$R \to .R*,$
$R \to .(R),$
$R \to .a$
$R \to .b$
}

$\mathrm{I}_3 = Goto(\mathrm{I}_0,' a')$ = {
$R \to a.$
}

$\mathrm{I}_4 = Goto(\mathrm{I}_0,' b')$ = {
$R \to b.$
}

$\mathrm{I}_5 = Goto(\mathrm{I}_1, R)$ = {
$R \to RR.,$
$R \to R.'|'R,$
$R \to R.R,$
$R \to R.*,$
$R \to .R'|'R,$
$R \to .RR,$
$R \to .R*,$
$R \to .(R),$
$R \to .a$
$R \to .b$
}

$\mathrm{I}_6 = Goto(\mathrm{I}_1,'' |'')$ = {
$R \to R'|'.R,$
$R \to .R'|'R,$
$R \to .RR,$
$R \to .R*,$
$R \to .(R),$
$R \to .a$
$R \to .b$
}

$\mathrm{I}_7 = Goto(\mathrm{I}_1,' *')$ = {
$R \to R * .,$
}

$\mathrm{I}_2 = Goto(\mathrm{I}_1,' (')$
$\mathrm{I}_3 = Goto(\mathrm{I}_1,' a')$
$\mathrm{I}_4 = Goto(\mathrm{I}_1,' b')$

$\mathrm{I}_8 = Goto(\mathrm{I}_2, R)$ = {
$R \to (R.),$
$R \to R.'|'R,$
$R \to R.R,$
$R \to R.*,$
$R \to .R'|'R,$
$R \to .RR,$
$R \to .R*,$
$R \to .(R),$
$R \to .a$
$R \to .b$
}

$\mathrm{I}_2 = Goto(\mathrm{I}_2,' (')$
$\mathrm{I}_3 = Goto(\mathrm{I}_2,' a')$
$\mathrm{I}_4 = Goto(\mathrm{I}_2,' b')$

$\mathrm{I}_5 = Goto(\mathrm{I}_5,' R')$
$\mathrm{I}_6 = Goto(\mathrm{I}_5,'' |'')$
$\mathrm{I}_7 = Goto(\mathrm{I}_5,' *')$
$\mathrm{I}_2 = Goto(\mathrm{I}_5,' (')$
$\mathrm{I}_3 = Goto(\mathrm{I}_5,' a')$
$\mathrm{I}_4 = Goto(\mathrm{I}_5,' b')$

$\mathrm{I}_9 = Goto(\mathrm{I}_6, R)$ = {
$R \to R'|'R.$,
$R \to R.'|'R$,
$R \to R.R$,
$R \to R.*$,
$R \to .R'|'R$,
$R \to .RR$,
$R \to .R*$,
$R \to .(R)$,
$R \to .a$
$R \to .b$
}

$\mathrm{I}_2 = Goto(\mathrm{I}_6,' (')$
$\mathrm{I}_3 = Goto(\mathrm{I}_6,' a')$
$\mathrm{I}_4 = Goto(\mathrm{I}_6,' b')$

$\mathrm{I}_6 = Goto(\mathrm{I}_8, R)$
$\mathrm{I}_5 = Goto(\mathrm{I}_8,'' |'')$
$\mathrm{I}_7 = Goto(\mathrm{I}_8,' *')$
$\mathrm{I}_2 = Goto(\mathrm{I}_8,' (')$
$\mathrm{I}_{10} = Goto(\mathrm{I}_8,')')$ = {
$R \to (R).$,
}

$\mathrm{I}_3 = Goto(\mathrm{I}_8,' a')$
$\mathrm{I}_4 = Goto(\mathrm{I}_8,' b')$
$\mathrm{I}_6 = Goto(\mathrm{I}_9, R)$
$\mathrm{I}_5 = Goto(\mathrm{I}_9,'' |'')$
$\mathrm{I}_7 = Goto(\mathrm{I}_9,' *')$
$\mathrm{I}_2 = Goto(\mathrm{I}_9,' (')$
$\mathrm{I}_3 = Goto(\mathrm{I}_9,' a')$
$\mathrm{I}_4 = Goto(\mathrm{I}_9,' b')$

Note that 'R' is the only Non-Terminal in the grammar G, with FIRST(R) = '(', 'a', 'b' and FOLLOW(R) = "|", '*', ')', '$'. Using this and the above LR(0) items, we construct the the state diagram and SLR Table as follows:

State	Actions							Goto
	”|”	’*’	’(’	’)’	’a’	’b’	’$’	R
0			s2		s3	s4		1
1	s6	s7	s2		s3	s4	acc	5
2			s2		s3	s4		8
3	r5	r5	r5	r5	r5	r5	r5	
4	r6	r6	r6	r6	r6	r6	r6	
5	r2/s6	r2/s7	r2/s2	r2	r2/s3	r2/s4	r2	5
6			s2		s3	s4		9
7	r3	r3	r3	r3	r3	r3	r3	
8	s6	s7	s2	s10	s3	s4		5
9	r1/s6	r1/s7	r1/s2	r1	r1/s3	r1/s4	r1	5
10	r4	r4	r4	r4	r4	r4	r4	

We observe that there are 10 shift-reduce conflicts in above SLR(1) table. Te ambiguity is in Action[5,’|’], Action[5,’|’], Action[5,’*’], Action[5,’|(’], Action[5,’a’], Action[5,’b’], Action[9,’|’], Action[9,’*’], Action[9,’(’], Action[9,’a’] and Action[9,’b’].

To overcome this problem, we introduce precedence and associativity for operators. The precedence is $() > * > concatenate(r2) > |$ and these are left-associative. Based on this, we can construct the new SLR table without any ambiguity :

State	Actions							Goto
	”|”	’*’	’(’	’)’	’a’	’b’	’$’	R
0			s2		s3	s4		1
1	s6	s7	s2		s3	s4	acc	5
2			s2		s3	s4		8
3	r5	r5	r5	r5	r5	r5	r5	
4	r6	r6	r6	r6	r6	r6	r6	
5	r2	s7	r2	r2	r2	r2	r2	5
6			s2		s3	s4		9
7	r3	r3	r3	r3	r3	r3	r3	
8	s6	s7	s2	s10	s3	s4		5
9	r1	s7	s2	r1	s3	s4	r1	5
10	r4	r4	r4	r4	r4	r4	r4	

4 Problem 4

- For this problem, I have used ply. The definition ahead is excerpted from https://www.dabeaz.com/ply/ply.html. *PLY is a pure-Python implementation of the popular compiler construction tools lex and yacc. The main goal of PLY is to stay fairly faithful to the way in which traditional lex/yacc tools work.*
- **INSTALLATION** : **pip3 install ply**.
- Make sure input file and p4.py are in same folder.
- command to run sample.txt :

  ```
  python3 p4.py sample.txt
  ```

  Output will be shown in stdout.
- Code is present in **lex.py**.
- **ERROR HANDLING** if the file contains any errors, the program will terminate with an error symbol.