CS335A: Assignment 2

Udit Prasad (201055)

Question 1

Solution:
Given rules are:

$$S \to (L)|a \tag{1}$$
$$L \to L, S|LS|b \tag{2}$$

We will first perform Left Factoring over this.
Rule 2 can be left factored after which grammar will be :

$$S \to (L)|a \tag{0}$$
$$L \to LA|b \tag{1}$$
$$A \to, S|S \tag{2}$$

Here we have no indirect Left Recursion but Rule 2 has Direct Left Recursion
To remove that, we will remove that rule and add 2 diff rules like this:

$$L \to bL' \tag{0}$$
$$L' \to AL'|\epsilon \tag{1}$$

Finally, left-factored and left-recursion free grammar will be :

$$S \to (L) \tag{0}$$
$$S \to a \tag{1}$$
$$L \to bL' \tag{2}$$
$$L' \to AL' \tag{3}$$
$$L' \to \epsilon \tag{4}$$
$$A \to, S \tag{5}$$
$$A \to S \tag{6}$$

Now, we will compute the First and Follow sets.

First(S) = {(a}	Follow(S) = {\$ a ( ) , }
First(L) = {b}	Follow(L) = {)}
First(L') = {ϵ , (a}	Follow(L') = {)}
First(A) = {, (a}	Follow(A) = {(, a)}

Finally we will form the Predictive Parsing Table

Non-Terminals	(	a	b	)	,	$
S	$S \to (L)$	$S \to a$				
L			$L \to bL'$			
L'	$L' \to AL'$	$L' \to AL'$		$L' \to \epsilon$	$L' \to AL'$	
A	$A \to S$	$A \to S$			$A \to, S$	

□

Question 2

Solution: Given rules are:

$$S \to Lp|qLr|sr|qsp \tag{0}$$
$$L \to s \tag{1}$$

First we will add a new rule,

$$S' \to S \tag{0}$$
$$S \to Lp \tag{1}$$
$$S \to qLr \tag{2}$$
$$S \to sr \tag{3}$$
$$S \to qsp \tag{4}$$
$$L \to s \tag{5}$$

Now, we will calculate the Follow and First sets of Non-Terminals

First(S) = $\{q\ s\}$ Follow(S) = $\{\$\}$
First(L) = $\{s\}$ Follow(L) = $\{pr\}$

Now we will evaluate the Canonical Collection of Sets of LR(0) Items.

$I_0 = Closure\{\text{S'} \to \cdot S\} = \{$
$S' \to \cdot S,$
$S \to \cdot Lp,$
$S \to \cdot qLr,$
$S \to \cdot sr,$
$S \to \cdot qsp,$
$L \to \cdot s\}$

$I_1 = Goto(I_0, S) = \{$
$\text{S'} \to S\cdot\}$

$I_2 = Goto(I_0, L) = \{$
$\text{S} \to L \cdot p\}$

$I_3 = Goto(I_0, q) = \{$
$\text{S} \to q \cdot Lr,$
$S \to q \cdot sp,$
$L \to \cdot s\}$

$I_4 = Goto(I_0, s) = \{$
$\text{S} \to s \cdot r,$
$L \to s\cdot\}$

$I_5 = Goto(I_2, p) = \{$
$\text{S} \to Lp\cdot\}$

$I_6 = Goto(I_3, L) = \{$
$\text{S} \to qL \cdot r\}$

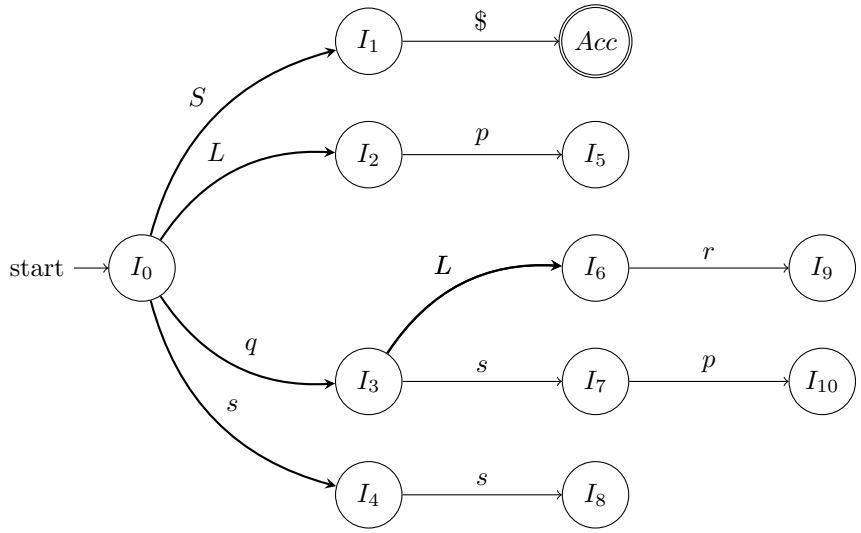
$I_7 = Goto(I_3, s) = \{$
$\text{S} \to qs \cdot p,$
$L \to s\cdot\}$

$I_8 = Goto(I_4, r) = \{$
$\text{S} \to sr\cdot\}$

$I_9 = Goto(I_6, r) = \{$
$\text{S} \to qLr\cdot\}$

$I_{10} = Goto(I_7, p) = \{$
$S \to qsp\cdot\}$

Now, we will draw the Automaton for LR(0) sets



Now we will make the SLR(1) table

	Action					Goto	
State	s	q	r	p	$	S	L
0	s_4	s_3				1	2
1					accepted		
2				s_5			
3	s_7						6
4			s_8, r_5	r_5			
5					r_1		
6			s_9				
7			r_5	s_{10}, r_5			
8					r_3		
9					r_2		
10					r_4		

As we can see we have Shift-Reduce Conflicts in the table, we can infer that the grammar is not SLR(1).
Now, we will show that this grammar is LALR(1).
For, this we first need to construct LR(1) Items

$I_0 = Closure\{[S' \to \cdot S, \$]\} = \{$
$S' \to \cdot S, \$,$
$S \to \cdot Lp, \$,$
$S \to \cdot qLr, \$,$
$S \to \cdot sr, \$,$
$S \to \cdot qsp, \$,$
$L \to \cdot s, p$
$\}$

$I_1 = Goto(I_0, S) = \{$
$S' \to S\cdot, \$$
$\}$

$I_2 = Goto(I_0, L) = \{$
$S \to L \cdot p, \$$
$\}$

$\mathrm{I}_3 = Goto(I_0, q) = \{$
$\mathrm{S} \to q \cdot Lr, \$,$
$S \to q \cdot sp, \$,$
$L \to \cdot s, r$
$\}$

$\mathrm{I}_4 = Goto(I_0, s) = \{$
$\mathrm{S} \to s \cdot r, \$,$
$L \to s\cdot, p$
$\}$

$\mathrm{I}_5 = Goto(I_2, p) = \{$
$\mathrm{S} \to Lp\cdot, \$$
$\}$

$\mathrm{I}_6 = Goto(I_3, L) = \{$
$\mathrm{S} \to qL \cdot r, \$$
$\}$

$\mathrm{I}_7 = Goto(I_3, s) = \{$
$\mathrm{S} \to qs \cdot p, \$,$
$L \to s\cdot, r$
$\}$

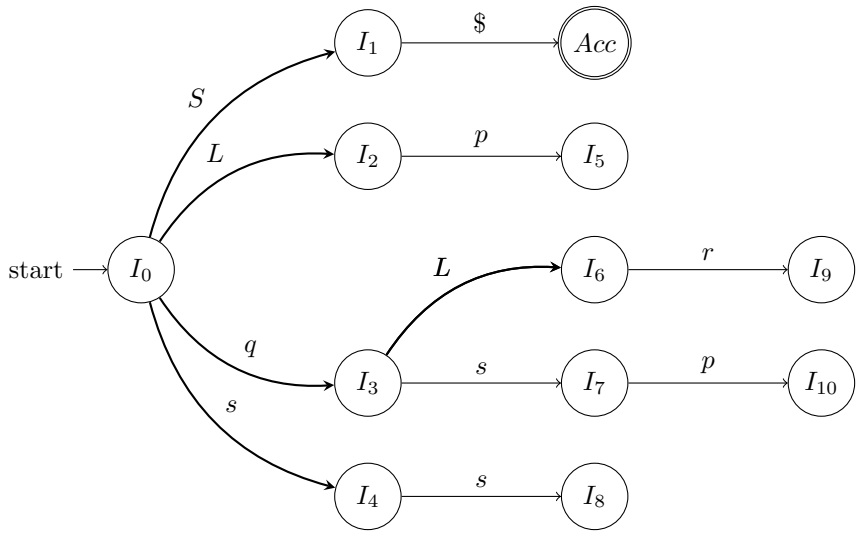
$\mathrm{I}_8 = Goto(I_4, r) = \{$
$\mathrm{S} \to sr\cdot, \$$
$\}$

$\mathrm{I}_9 = Goto(I_6, r) = \{$
$\mathrm{S} \to qLr\cdot, \$$
$\}$

$\mathrm{I}_{10} = Goto(I_7, p) = \{$
$\mathrm{S} \to qsp\cdot, \$$
$\}$

Now, we will look for all the items whose cores are same and merge them into one state, but there are none. So this is our final LALR collection.

Now we will draw the automaton for above LALR collection of items.



Now,we will construct the LALR parsing table

	Action					Goto	
State	s	q	r	p	$	S	L
0	s_4	s_3				1	2
1					accepted		
2				s_5			
3	s_7						6
4			s_8	r_5			
5					r_1		
6			s_9				
7			r_5	s_{10}			
8					r_3		
9					r_2		
10					r_4		

As we can see there is no conflict in LALR table, hence the grammar is LALR(1).

Question 3

Given rules are:

$$R \to R|R \tag{0}$$
$$R \to RR \tag{1}$$
$$R \to R* \tag{2}$$
$$R \to (R) \tag{3}$$
$$R \to a|b \tag{4}$$

We will add the production $R' \to R$ to the given grammar, Now the final grammar looks like:

$$R' \to R \tag{0}$$
$$R \to R|R \tag{1}$$
$$R \to RR \tag{2}$$
$$R \to R* \tag{3}$$
$$R \to (R) \tag{4}$$
$$R \to a \tag{5}$$
$$R \to b \tag{6}$$

Now, we will calculate the First and Follow sets.

First(R) = {(a b} Follow(R) = {| $*$) (a b}

Now, we will evaluate the Canonical Collection of Sets of LR(0) Items

$I_0 = \text{Closure}[R' \to \cdot R] = \{$
$R' \to \cdot R,$
$R \to \cdot R|R,$
$R \to \cdot RR,$
$R \to \cdot R*,$
$R \to \cdot(R),$
$R \to \cdot a,$
$R \to \cdot b$
$\}$

$I_1 = \text{Goto}(I_0, R) = \{$
$R' \to R\cdot,$
$R \to R \cdot |R,$
$R \to R \cdot R,$
$R \to R \cdot *,$
$R \to \cdot R|R,$
$R \to \cdot RR,$
$R \to \cdot R*,$
$R \to \cdot(R),$
$R \to \cdot a,$
$R \to \cdot b$
$\}$

$I_2 = \text{Goto}(I_0, () = \{$
$R \to (\cdot R),$
$R \to \cdot R|R,$
$R \to \cdot RR,$
$R \to \cdot R*,$
$R \to \cdot(R),$
$R \to \cdot a,$
$R \to \cdot b$
$\}$

$I_3 = \text{Goto}(I_0, a) = \{$
$R \to a \cdot$
$\}$

$I_4 = \text{Goto}(I_0, b) = \{$
$R \to b \cdot$
$\}$

$I_5 = \text{Goto}(I_1, |) = \{$
$R \to R| \cdot R,$
$R \to \cdot R|R,$
$R \to \cdot RR,$
$R \to \cdot R*,$
$R \to \cdot(R),$
$R \to \cdot a,$
$R \to \cdot b$
$\}$

$I_6 = \text{Goto}(I_1, R) = \{$
$R \to R \cdot |R,$
$R \to RR\cdot,$
$R \to R \cdot R,$
$R \to R \cdot *,$
$R \to \cdot R|R,$
$R \to \cdot RR,$
$R \to \cdot R*,$
$R \to \cdot(R),$
$R \to \cdot a,$
$R \to \cdot b$
$\}$

$I_7 = \text{Goto}(I_1, *) = \{$
$R \to R * \cdot$
$\}$

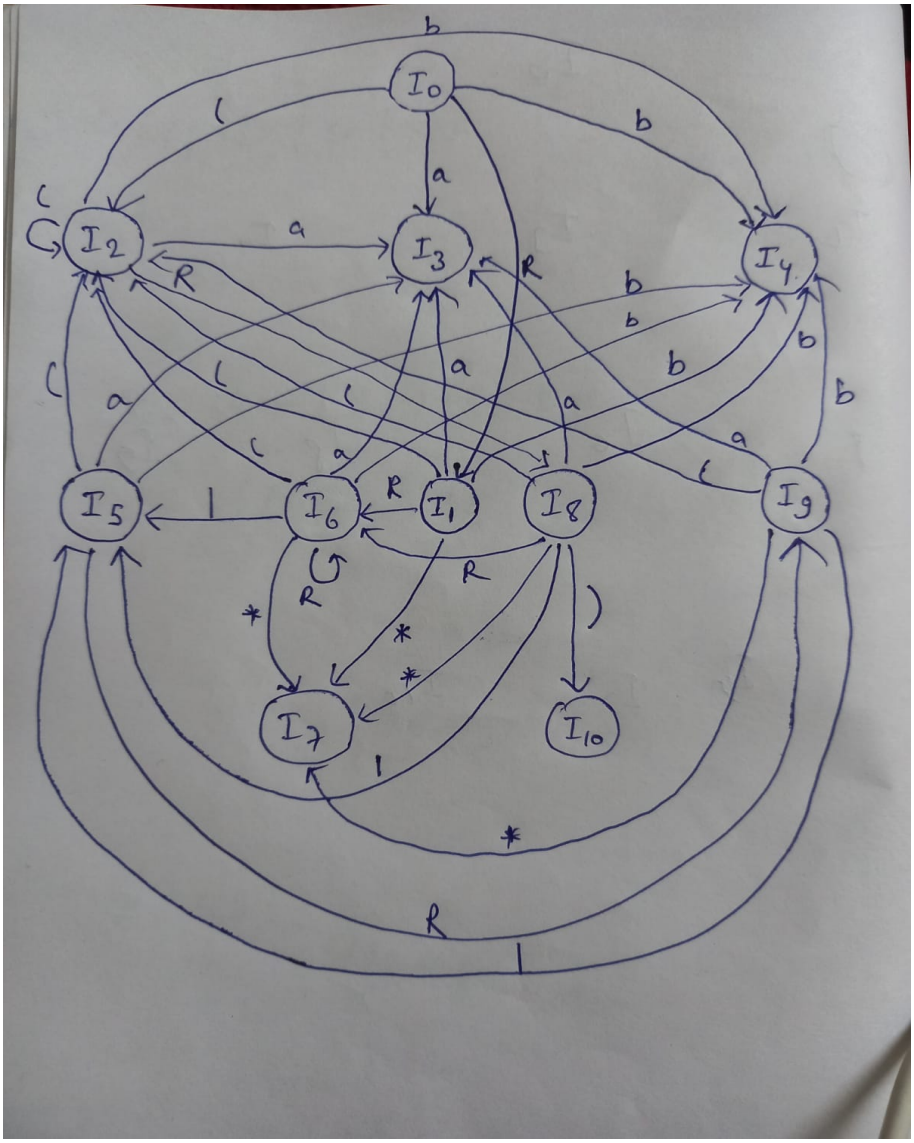
$I_8 = \text{Goto}(I_2, R) = \{$
$R \to (R\cdot),$
$R \to R \cdot |R,$
$R \to R \cdot R,$
$R \to R \cdot *,$
$R \to \cdot R|R,$
$R \to \cdot RR,$
$R \to \cdot R*,$
$R \to \cdot(R),$
$R \to \cdot a,$
$R \to \cdot b$
$\}$

$I_9 = \text{Goto}(I_5, R) = \{$
$R \to R|R\cdot,$
$R \to R \cdot |R,$
$R \to R \cdot R,$
$R \to R \cdot *,$
$R \to \cdot R|R,$
$R \to \cdot RR,$
$R \to \cdot R*,$
$R \to \cdot(R),$
$R \to \cdot a,$
$R \to \cdot b$
$\}$

$I_{10} = \text{Goto}(I_8,)) = \{$
$R \to (R) \cdot$
$\}$

$\text{Goto}(I_1, () = I_2$
$\text{Goto}(I_1, a) = I_3$
$\text{Goto}(I_1, b) = I_4$
$\text{Goto}(I_2, () = I_2$
$\text{Goto}(I_2, a) = I_3$
$\text{Goto}(I_2, b) = I_4$
$\text{Goto}(I_5, () = I_2$
$\text{Goto}(I_5, a) = I_3$
$\text{Goto}(I_5, b) = I_4$
$\text{Goto}(I_6, |) = I_5$
$\text{Goto}(I_6, R) = I_6$
$\text{Goto}(I_6, *) = I_7$
$\text{Goto}(I_6, () = I_2$
$\text{Goto}(I_6, a) = I_3$
$\text{Goto}(I_6, b) = I_4$
$\text{Goto}(I_8, |) = I_5$
$\text{Goto}(I_8, R) = I_6$
$\text{Goto}(I_8, *) = I_7$
$\text{Goto}(I_8, () = I_2$
$\text{Goto}(I_8, a) = I_3$
$\text{Goto}(I_8, b) = I_4$
$\text{Goto}(I_9, |) = I_5$
$\text{Goto}(I_9, R) = I_6$
$\text{Goto}(I_9, *) = I_7$
$\text{Goto}(I_9, () = I_2$
$\text{Goto}(I_9, a) = I_3$
$\text{Goto}(I_9, b) = I_4$

Now we will draw the automaton for above LR(0) collection of items.



Finally, we will construct the LSR Parsing table

	Action							Goto
State	a	b	(	)	*	|	$	R
0	s_3	s_4	s_2					1
1	s_3	s_4	s_2		s_7	s_5	Accept	6
2	s_3	s_4	s_2					8
3	r_5	r_5	r_5	r_5	r_5	r_5	r_5	
4	r_6	r_6	r_6	r_6	r_6	r_6	r_6	
5	s_3	s_4	s_2					9
6	r_2, s_3	r_2, s_4	r_2, s_2	r_2	r_2, s_7	r_2, s_5	r_2	6
7	r_3	r_3	r_3	r_3	r_3	r_3	r_3	
8	s_3	s_4	s_2	s_10	s_7	s_5		6
9	s_3, r_1	s_4, r_1	s_2, r_1	r_1	s_7, r_1	s_5, r_1	r_1	6
10	r_4	r_4	r_4	r_4	r_4	r_4	r_4	

Here, we can clearly there are many shift-reduce conflicts.

We can make use of precedence and associative rules to get rid of these conflicts.

Precedence order of () $>$ $*$ $>$ concate $>$ |

Here concatenate is operator between terminal and non-terminal(Rb) or Non-Terminals (RR) or two terminals (aa).

All of these operators are left associative.

We mainly have conflicts in State 6 and 9.

1. In State 6, on seeing a or b, we reduce following the associativity rule.
2. In State 6, on seeing (, we reduce following the associativity rule.
3. In State 6, on seeing $*$, we shift following the precedence order.
4. In State 6, on seeing |, we reduce following the precedence order.
5. In State 9, on seeing a or b,we shift following the precedence order.
6. In State 9, on seeing (, we shift following the precedence order.
7. In State 9, on seeing $*$, we shift following the precedence order.
8. In State 9, on seeing |, we reduce following the associativity rule

Using these rules, we will now form the reformed SLR Parsing table which will be free of conflicts.

	Action							Goto
State	a	b	(	)	*	|	$	R
0	s_3	s_4	s_2					1
1	s_3	s_4	s_2		s_7	s_5	Accept	6
2	s_3	s_4	s_2					8
3	r_5	r_5	r_5	r_5	r_5	r_5	r_5	
4	r_6	r_6	r_6	r_6	r_6	r_6	r_6	
5	s_3	s_4	s_2					9
6	r_2	r_2	r_2	r_2	s_7	r_2	r_2	6
7	r_3	r_3	r_3	r_3	r_3	r_3	r_3	
8	s_3	s_4	s_2	s_{10}	s_7	s_5		6
9	s_3	s_4	s_2	r_1	s_7	r_1	r_1	6
10	r_4	r_4	r_4	r_4	r_4	r_4	r_4	

□

Question 4

Solution: Tools used :

1. Flex - used to make tokens which will be used by Bison.
2. Bison - It will parse the tokens using grammar written.

Assumptions made :

1. After the last paragraph, there should not be more than 2 EOLs.
2. Chapter will be followed by a number and then by colon.
3. Section will be followed by a float and then by colon.
4. Title, Chapter and Section lines cannot have period ? and !.

Commands for compliation and execution :
Compilation :

```
flex lexer.l
bison -d -t parser.y
flex lexer.l
```

Execution :

```
g++ lex.yy.c parser.tab.c run.cpp -o hello
./hello input.txt
```

□