Info of Exam

- Closed-book examination: 2025-01-08 08:00-09:50
- Q & A (Tencent Meeting): 2024-12-31, 08:30-10:00

- Problem types:
  - Fill-in-the-blank
  - True / False
  - Short questions
  - Long questions

- Grade system:
  - Final exam + Homework + Attendance

Outline

- PPT courseware + Homework

- Notice: specific tools (e.g., Lex or Yacc) or sample languages (e.g., TINY or C-minus), will not be examined.

Outline-1

- The phases of a compiler
- Write regular expressions
  - Pay attention to basic and extended operations
- Convert the regular expression into an NFA, and then convert the NFA into a DFA
  - Subset construction method
- Minimize DFA

Outline-2

- Derivation and parse tree
- Write a Context-Free Grammar
- Chomsky hierarchy
- Construct a LL(1) parsing table
  - Table construction: $M[N, T]$
  - $e \in$ First(α): fill A$\rightarrow\alpha$ in $M[A, e]$
  - $e \in$ Follow(A): fill A$\rightarrow\varepsilon$ in $M[A, e]$
- Parse an input string corresponding to the LL(1) parsing table

Outline-3

- Construct a LR(0)/SLR(1)/LR(1)/LALR(1) parsing table
  - Table construction: state (list) + input (subtable) + goto (subtable)
  - SLR(1):
    - Follow(A): reduce with A→α
  - LR(1)
    - $b \in \text{First}(\gamma a)$: ε-transition from item $[A \rightarrow \alpha \cdot B\gamma, a]$ to item $[B \rightarrow \cdot\beta, b]$
- A grammar is LL or LR grammer?
  - No shift-reduce or reduce-reduce conflicts
- Parse an input string corresponding to the LR parsing table

Exercises

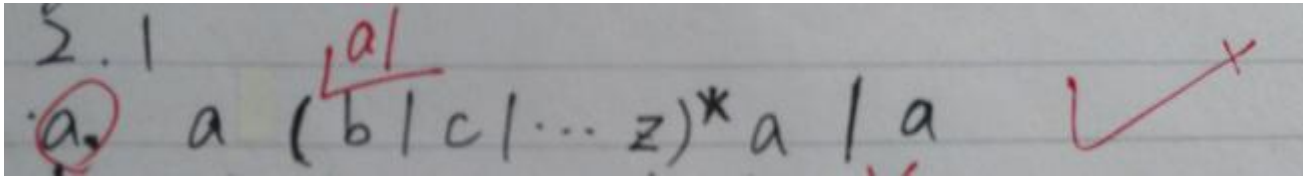
2.1

A. All strings of lowercase letters that begin and end in *a*.

Positive examples (x^+): *a, aa, aba, abaabca, …*

Negative examples (x^-): ε, *ab, ba, abb,…*

Perfect answer: *a*[*a*-*z*]***a* | *a*



2.1

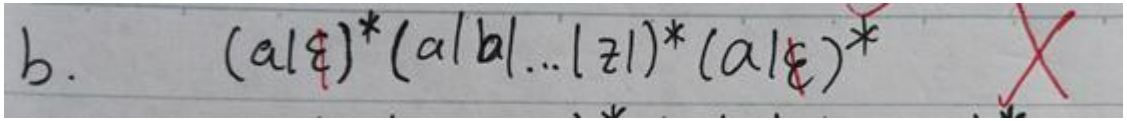
B. All strings of lowercase letters that either begin or end in *a* (or both).

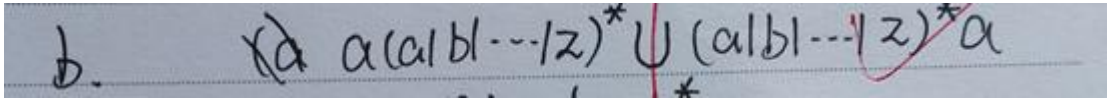
x^+: *a, aa, ab, ba, aba, aabcaca,…*

x^-: ε, *b, bab,…*

Perfect answer: $a[a\text{-}z]^* \mid [a\text{-}z]^*a$

Wrong answer:





2.1

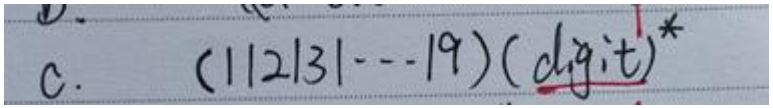
C. All strings of digits that contain no leading zeros.

x^+: *0, 1, 2, 12, 123…*

x^-: *01, 002, 023…*

Perfect answer: 0|[1-9][0-9]*

Wrong answer:



2.1

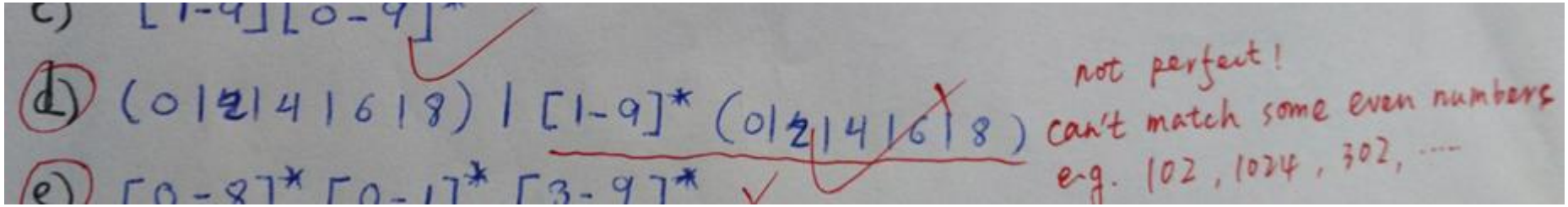
D. All strings of digits that represent even numbers.

x^+: *2, 4, 12, 112…*

x^-: *1, 3, 13…*

Perfect answer: [0|2|4|6|8]|[1-9][0-9]* [0|2|4|6|8]

Wrong answer:



2.1

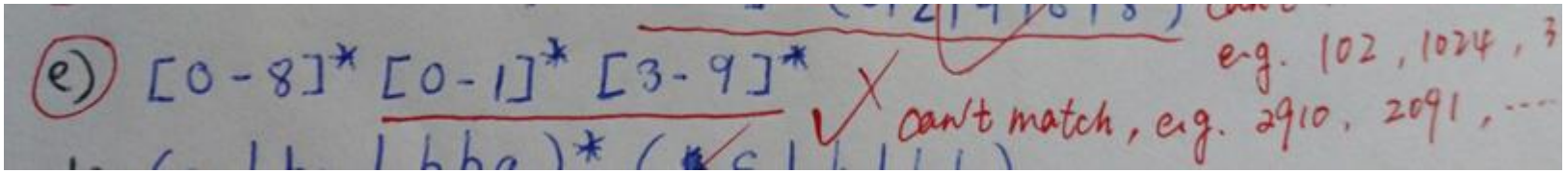
E. All strings of digits such that all the 2's occur before all the 9's.

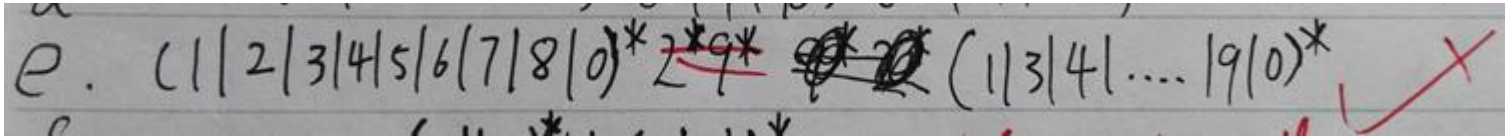
x^+: *2, 9, 29, 229, 12526996…*

x^-: *92, 992, 1913222…*

Perfect answer: $[0\text{-}8]^*[0|1|3|4|5|6|7|8|9]^*$

Wrong answer:





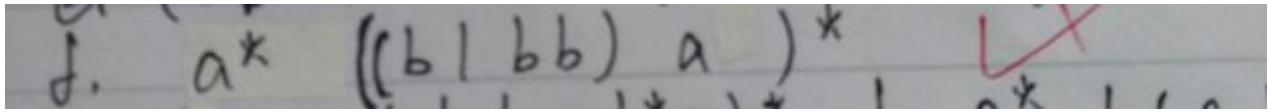
2.1

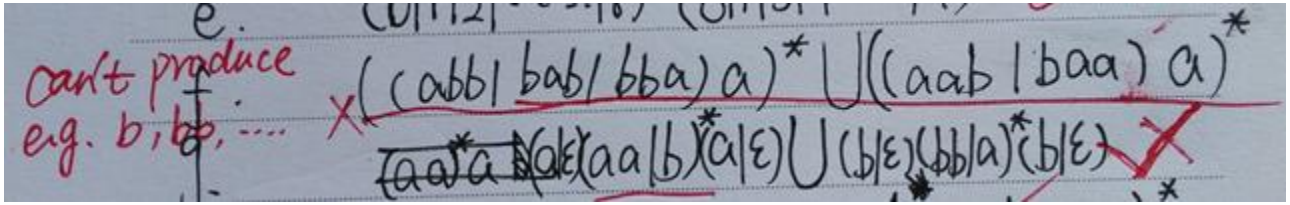
F. All strings of *a*'s and *b*'s that contain no three consecutive *b*'s.

x^+: *ε, b, bb, abb, bba, abbaaaba, bab, bbabb…*

x^-: *bbb, bbbb, abbb, ababaabbba…*

Perfect answer: (ε|*b*|*bb*)(*a*|*ab*|*abb*)* or (*a*|*ba*|*bba*)*(ε|*b*|*bb*)





2.1

G. All strings of *a*'s and *b*'s that contain an odd number of *a*'s or an odd number of *b*'s (or both).

x^+: *a, b, aaa, bbb, ab, aaabb, bababaaba…*

x^-: ε, *aa, bb, abab, aaaabb…*

Perfect answer: $b^*a(b^*ab^*a)^*b^* | a^*b(a^*ba^*b)^*a^*$

Wrong answer:

2.1

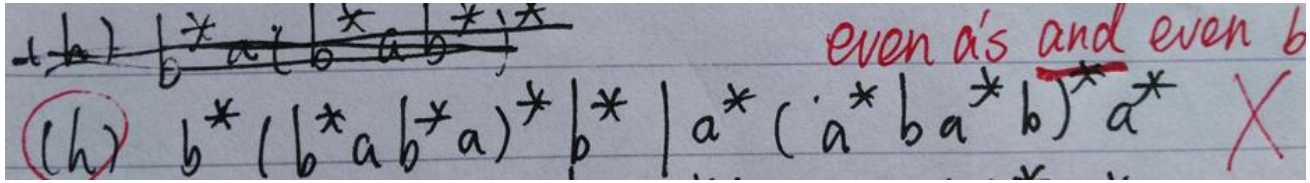
H. All strings of *a*'s and *b*'s that contain an even number of *a*'s and an even number of *b*'s

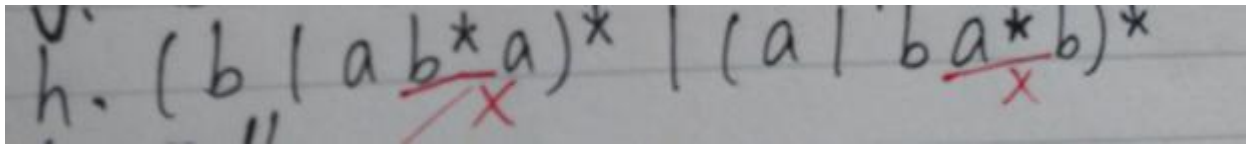
x^+: *ε, aa, bb, abab, aaaabb, bababaabaa…*

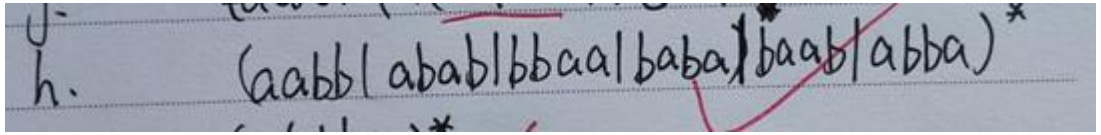
x^-: *a, b, aaa, bbb, ab, aaabb, bababaa…*

Perfect answer: $((aa|bb)|(ab|ba)(aa|bb)^*(ab|ba))^*$

Wrong answer:







2.1

- I. All strings of *a*'s and *b*'s that contain exactly as many *a*'s as *b*'s.

x^+: ε, *abab, bababaabab…*

x^-: *a, b, aaa, bbb, aaaabb…*

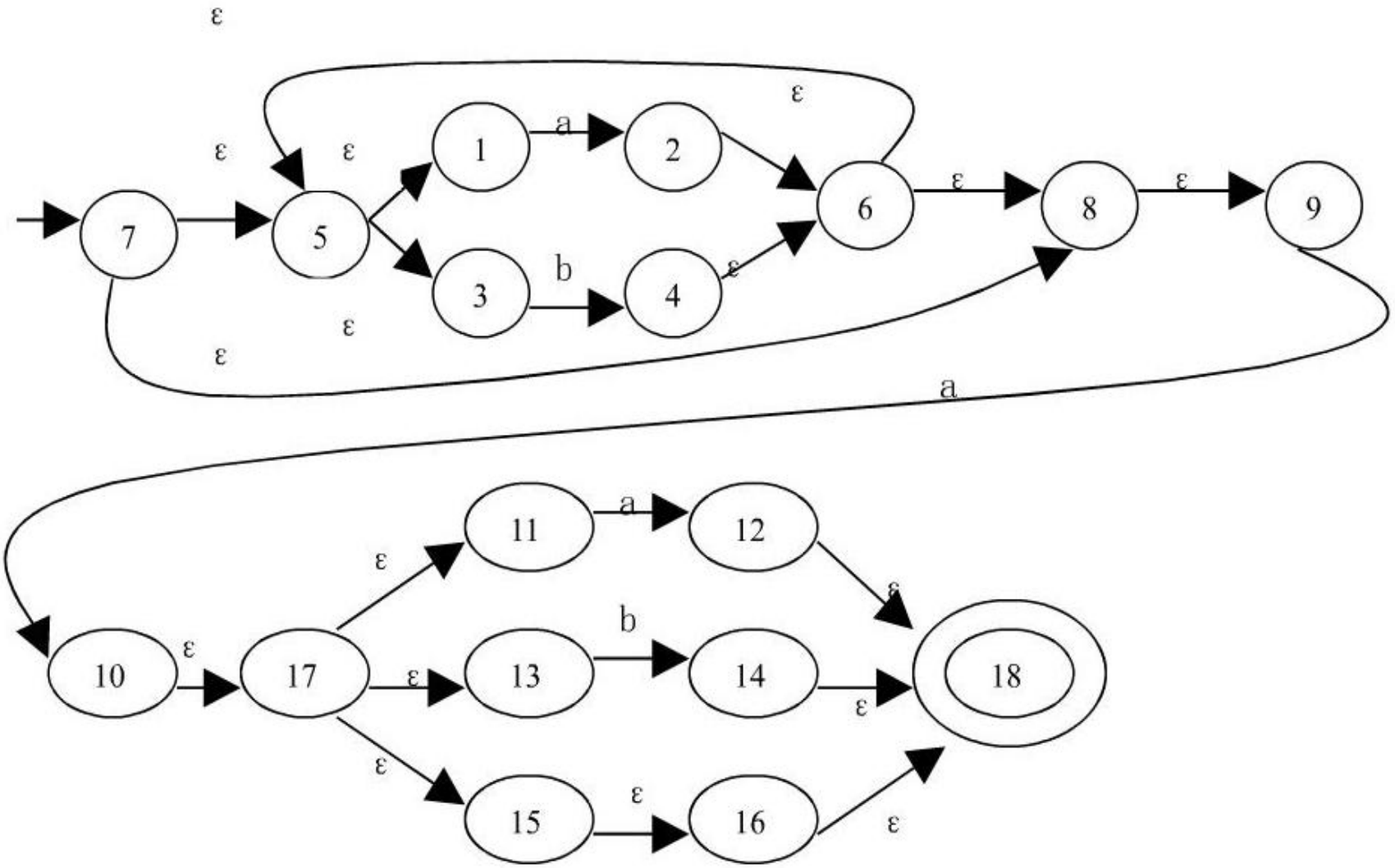
None. Because a regular expression cannot count.

2.12 a. Use Thompson's construction to convert the regular expression (a|b)*a(a|b|ε) into an NFA.

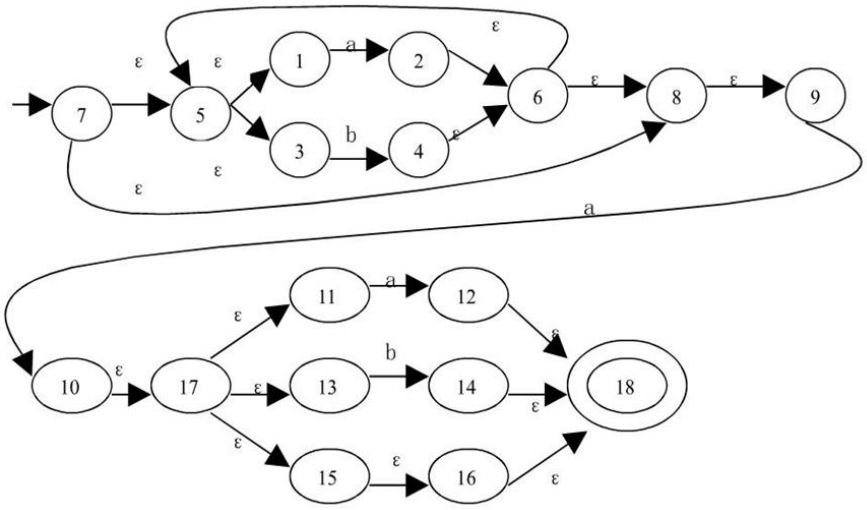
b. Convert the NFA of part (a) into a DFA using the subset construction.

[Solution]

a. An NFA of the regular expression (a|b)*a(a|b|ε)



b. The subsets constructed as follows:



{ 7 } = { 7, 5,1,3,8,9}

{ 7 }$_a$ = {2, 10}

{ 7 }$_b$ = {4}

{2,10} = {2,6,5,1,3,8,9, 10,17,11,13,15,16,18}

{2,10}$_a$ = {2, 10, 12}

{2,10}$_b$ = {4, 14}

{2,10,12} = {2,6,5,1,3,8,9,12,18,10,17,11,13,15,16}

{2,10,12}$_a$ = {2,10,12}

{2,10,12}$_b$ = {4, 14}

{4,14} = {4,6,5,1,3,8,9,14,18}

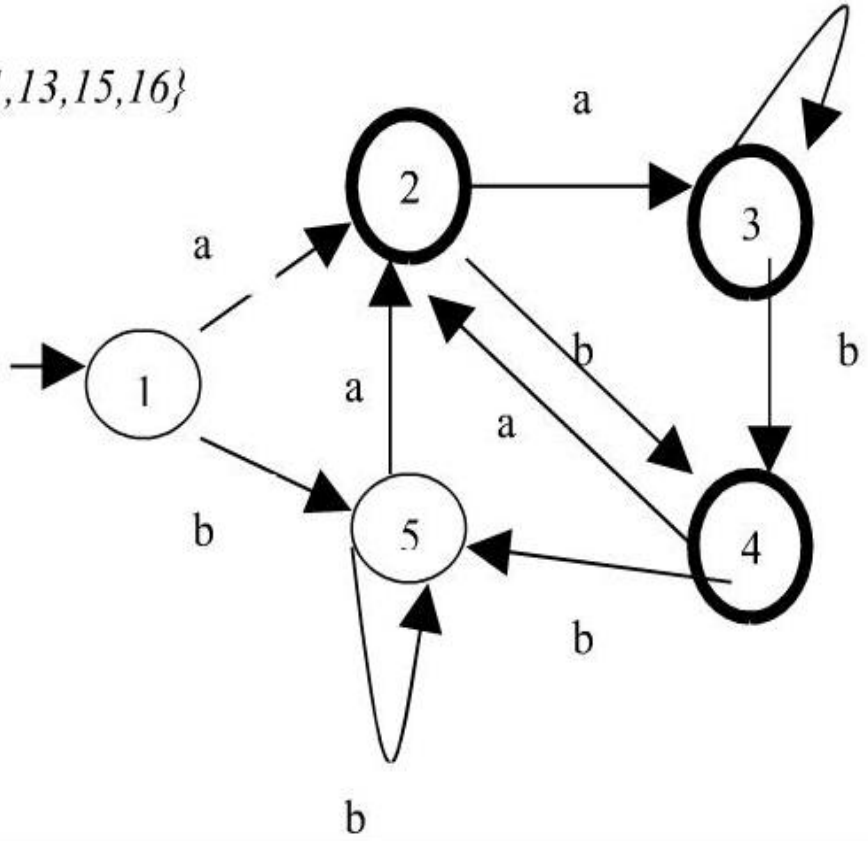
{4,14}$_a$ = {2,10}

{4,14}$_b$ = {4}

{4} = {4,6,5,1,3,8,9}

{4}$_a$ = {2,10}

{4}$_b$ = {4}



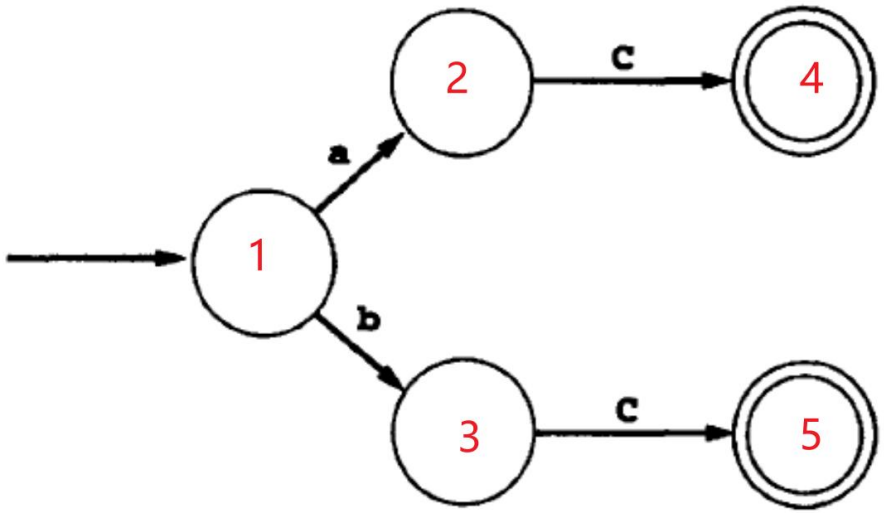
(b) From (a), we can get that:

S		S_a	S_b
7	7, 5, 1, 3, 8, 9	2, 10	4
2, 10	2, 6, 5, 1, 3, 8, 9, 10, 17, 11, 13, 15, 16, 18	2, 10, 12	4, 14
2, 10, 12	2, 6, 5, 1, 3, 8, 9, 10, 17, 11, 13, 15, 16, 18, 12	2, 10, 12	4, 14
4, 14	4, 6, 5, 1, 3, 8, 9, 14, 18	2, 10	4
4	4, 6, 5, 1, 3, 8, 9	2, 10	4

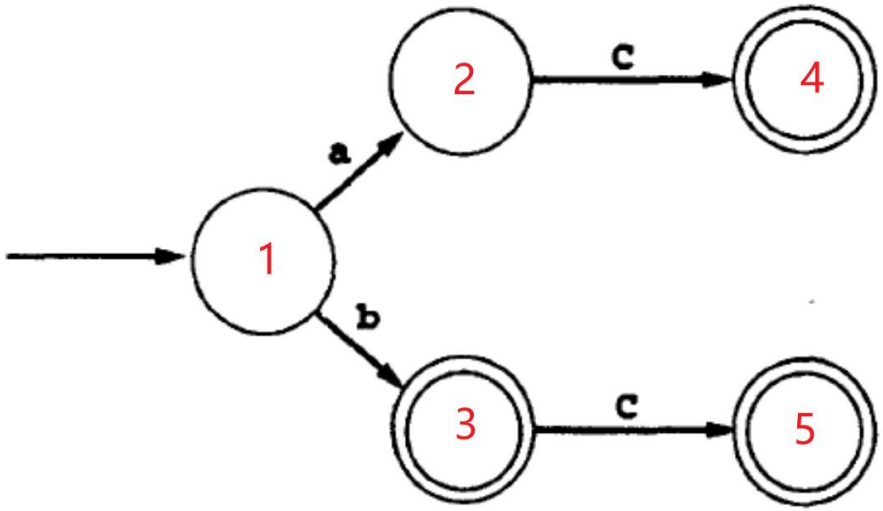
Therefore, the graph is:

2.16 Apply the state minimization algorithm of Section 2.4.4 to the following DFAs:

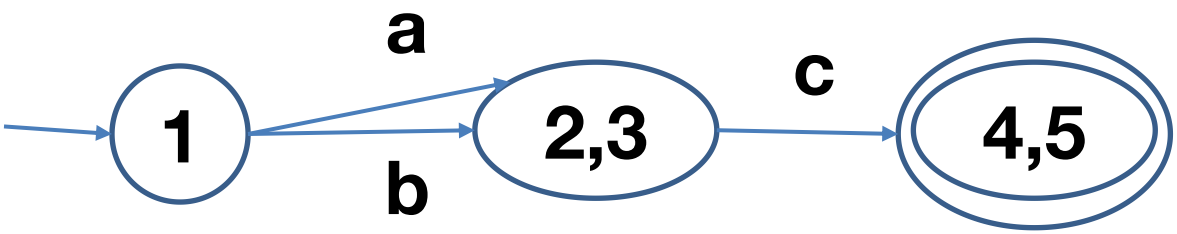
a.

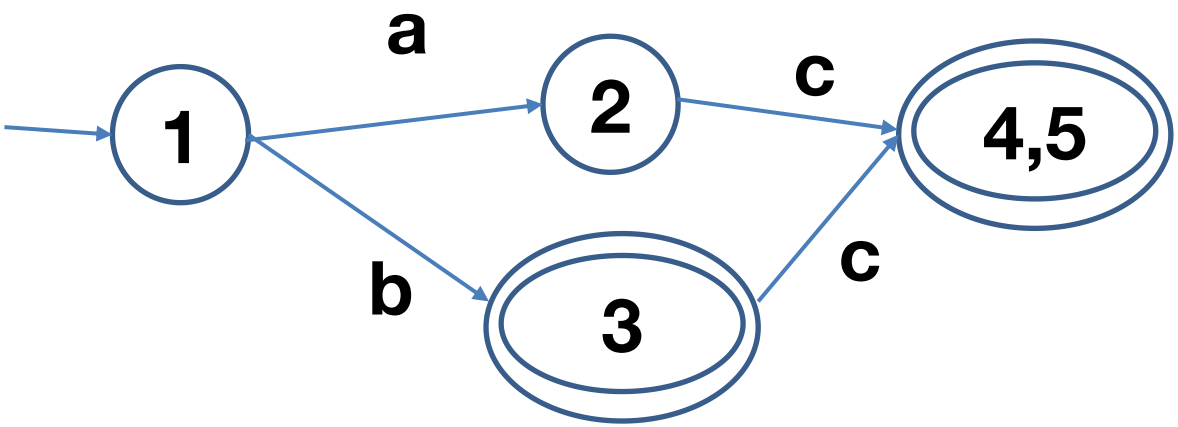


b.



2.16





3.20

3.20 a. Write a regular expression that generates the same language as the following grammar:

$$A \rightarrow aA \mid B \mid \varepsilon$$
$$B \rightarrow bB \mid A$$

b. Write a grammar that generates the same language as the following regular expression:

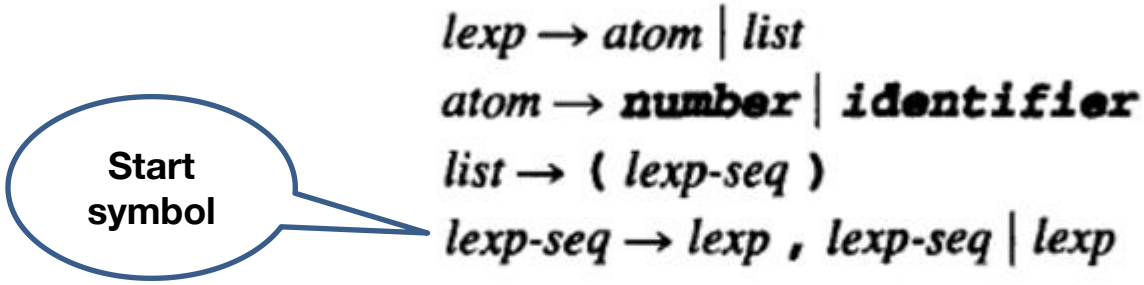
(a|c|ba|bc)*(b|ε)

1. $(a|b)^*$

2. $A \rightarrow Bb|B$
   $B \rightarrow aB|cB|baB|bcB|\varepsilon$

Or $A \rightarrow aA|cA|baA|bcA|B$
   $B \rightarrow b|\varepsilon$

4.9 Consider the following grammar (similar, but not identical to the grammar of Exercise 4.8):



a. Left factor this grammar.

b. Construct First and Follow sets for the nonterminals of the resulting grammar.

c. Show that the resulting grammar is LL(1).

d. Construct the LL(1) parsing table for the resulting grammar.

4.9

1.

lexp → atom|list

atom → number|identifier

list → (lexp-seq)

lexp-seq → lexp lexp-seq'

lexp-seq' → , lexp-seq|ε

$→Follow(lexp-seq)
Follow(lexp)→follow(atom) or follow(list)
)→follow(lexp-seq)
First(lexp-seq') except for ε
→follow(lexp)
Follow(lexp-seq)→follow(lexp)
Follow(lexp-seq)=Follow(lexp-seq')

Nonterminal	First set	Follow set
lexp	number, identifier, (	$,), ,
atom	number, identifier	$,), ,
list	(	$,), ,
lexp-seq	number, identifier, (	$,)
lexp-seq'	, ε	$,)

Nonterminal	First set	Follow set
lexp	number, identifier, (	$,), ,
atom	number, identifier	$,), ,
list	(	$,), ,
lexp-seq	number, identifier, (	$,)
lexp-seq’	, ε	$,)

M[N,T]	number	identifier	(	)	,	$
lexp	lexp → atom	lexp → atom	lexp → list			
atom	atom → number	atom → identifier				
list			list → (lexp-seq)			
lexp-seq	lexp-seq → lexp lexp-seq’	lexp-seq → lexp lexp-seq’	lexp-seq → lexp lexp-seq’			
lexp-seq’				lexp-seq’ → ε	lexp-seq’ → , lexp-seq	lexp-seq’ → ε

5.1 Consider the following grammar:

$$E \rightarrow (\ L\ ) \mid \mathtt{a}$$
$$L \rightarrow L\ ,\ E \mid E$$

a. Construct the DFA of LR(0) items for this grammar.

b. Construct the SLR(1) parsing table.

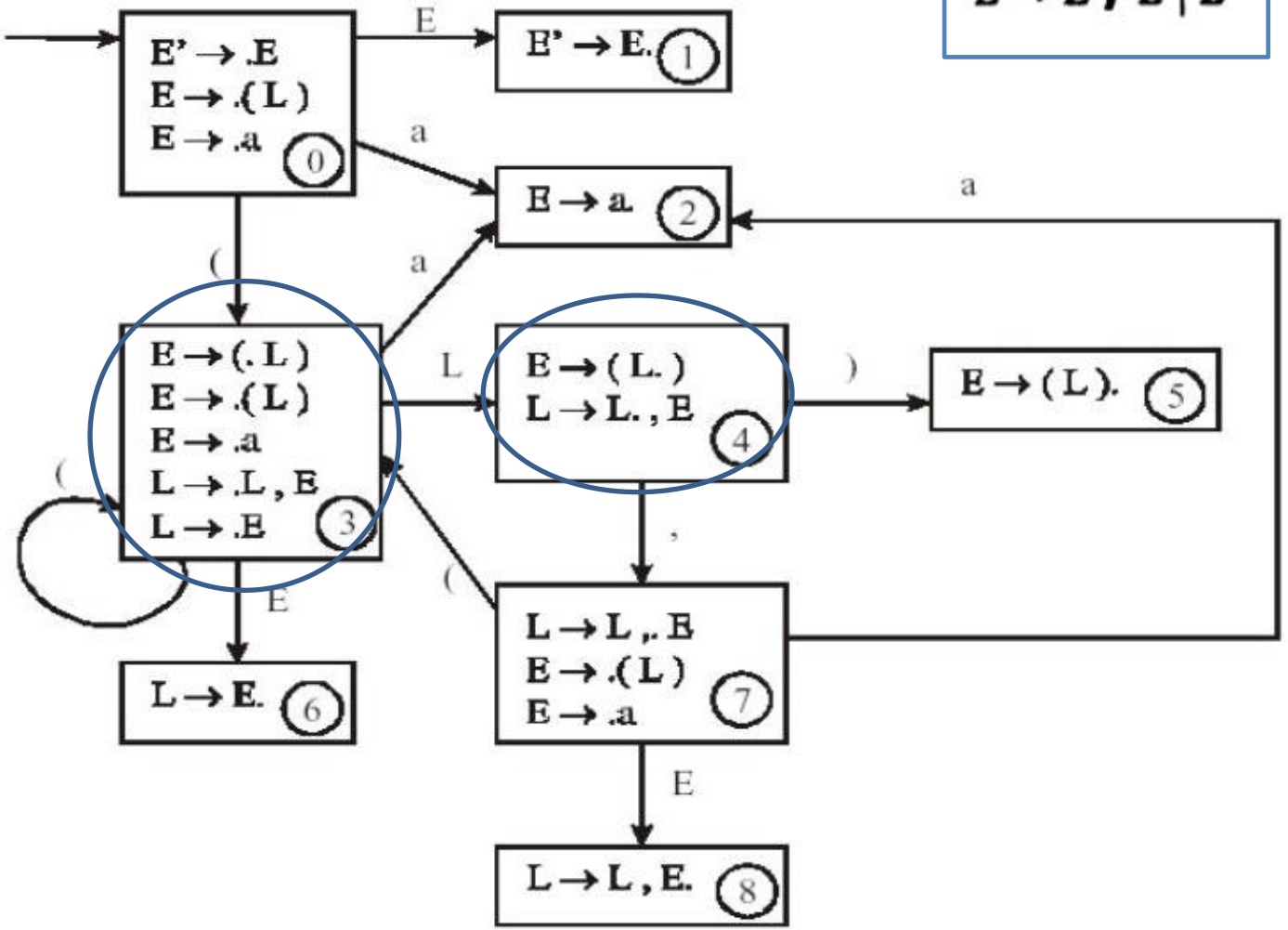
c. Show the parsing stack and the actions of an SLR(1) parser for the input string `((a),a,(a,a))`.

d. Is this grammar an LR(0) grammar? If not, describe the LR(0) conflict. If so, construct the LR(0) parsing table, and describe how a parse might differ from an SLR(1) parse.

5.1

a. DFA of LR(0) items

$E \rightarrow (L) \mid a$
$L \rightarrow L, E \mid E$



5.1

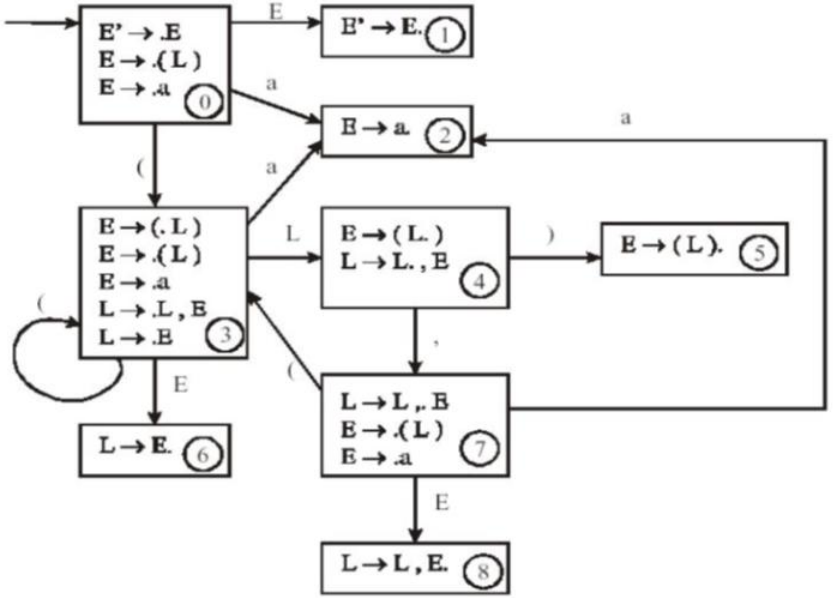
b. SLR(1) parsing table

Follow(E')={$}

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

Follow(L)={), ,}

$$E \rightarrow (\ L\ ) \mid a$$
$$L \rightarrow L\ ,\ E \mid E$$



State	Input					Goto	
	(	)	a	,	$	E	L
0	s3		s2			1	
1					accept		
2		r(E → a)		r(E → a)	r(E → a)		
3	s3		s2			6	4
4		s5		s7			
5		r(E→(L))		r(E→(L))	r(E→(L))		
6		r(L→E)		r(L→E)			
7	s3		s2			8	
8		r(L→L,E)		r(L→L,E)			

c. parse of ((a),a,(a,a))

	Parsing stack	Input	Action
1	$0	((a),a,(a,a))$	s3
2	$0(3	(a),a,(a,a))$	s3
3	$0(3(3	a),a,(a,a))$	s2
4	$0(3(3a2	),a,(a,a))$	r(E → a)
5	$0(3(3E6	),a,(a,a))$	r(L → E)
6	$0(3(3L4	),a,(a,a))$	s5
7	$0(3(3L4)5	,a,(a,a))$	r(E → (L))
8	$0(3E6	,a,(a,a))$	r(L → E)
9	$0(3L4	,a,(a,a))$	s7
10	$0(3L4,7	a,(a,a))$	s2
11	$0(3L4,7a2	,(a,a))$	r(E → a)
12	$0(3L4,7E8	,(a,a))$	r(L → L,E)
13	$0(3L4	,(a,a))$	s7
14	$0(3L4,7	(a,a))$	s3
15	$0(3L4,7(3	a,a))$	s2
16	$0(3L4,7(3a2	,a))$	r(E → a)
17	$0(3L4,7(3E6	,a))$	r(L → E)
18	$0(3L4,7(3L4	,a))$	s7
19	$0(3L4,7(3L4,7	a))$	s2
20	$0(3L4,7(3L4,7a2	))$	r(E → a)
21	$0(3L4,7(3L4,7E8	))$	r(L → L,E)
22	$0(33L4,7(3L4	))$	s5
23	$0(33L4,7(3L4)5	)$	r(E → (L))
24	$0(33L4,7E8	)$	r(L → L,E)
25	$0(3L4	)$	s5
26	$0(3L4)5	$	r(E → (L))
27	$0E1	$	accept

5.1

d. Is LR(0) grammar?

Yes. No conflicts.

State	Action	Rule	Input				Goto	
			(	a	)	,	E	L
0	shift		3	2			1	
1	reduce	E' → E						
2	reduce	E → a						
3	shift		3	2			6	4
4	shift				5	7		
5	reduce	E → (L)						
6	reduce	L → E						
7	shift		3	2			8	
8	reduce	L → L,E						

5.2 Consider the grammar of the previous exercise.

a. Construct the DFA of LR(1) items for this grammar.

b. Construct the general LR(1) parsing table.

c. Construct the DFA of LALR(1) items for this grammar.

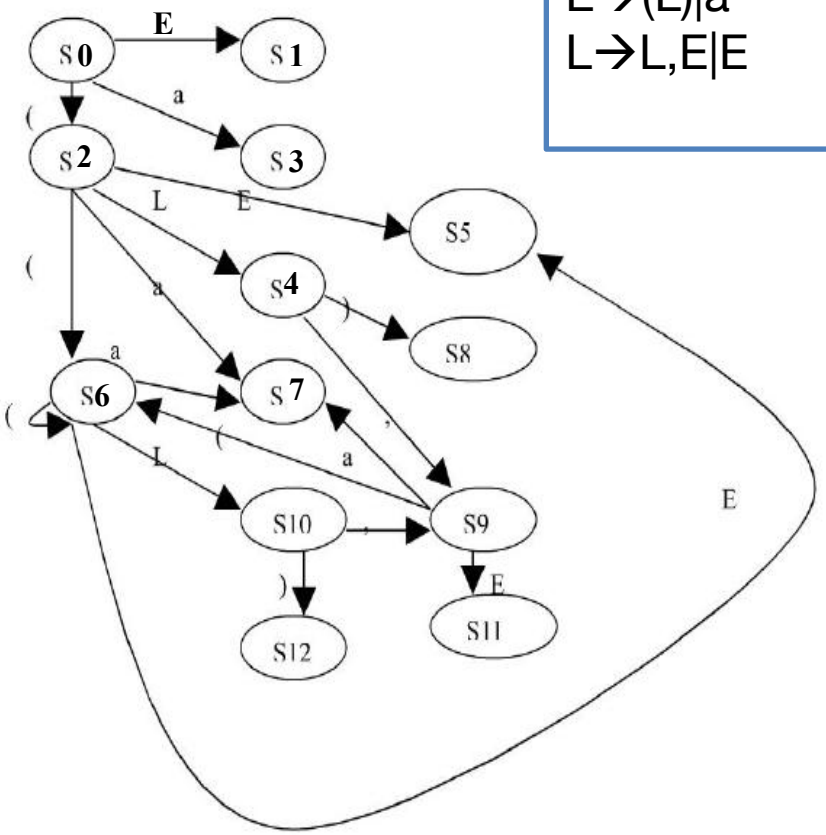
d. Construct the LALR(1) parsing table.

e. Describe any differences that might occur between the actions of a general LR(1) parser and an LALR(1) parser.

5.2

a. DFA of LR(1) items

E'→E
E→(L)|a
L→L,E|E



State 0: [E'→.E,$]
[E→. (L),$]
[E→. a,$]

State 1: [E'→E.,$]

State 2: [E→(.L),$]
[L→. L,E ,)]
[L→. E ,)]
[L→. L,E , ,]
[L→. E , ,]
[E→. (L),)]
[E→. a,)]
[E→. (L), ,]
[E→. a, ,]

State 3: [E→a.,$]

State 4: [E→(L.),$]
[L→L.,E ,)]
[L→L.,E , ,]

State 5: [L→E. ,)]
[L→E. , ,]

State 6: [E→(.L),)]
[E→(.L), ,]
[L→. L,E ,)]
[L→. E ,)]
[L→. L,E , ,]
[L→. E , ,]
[E→. (L),)]
[E→. a,)]
[E→. (L), ,]
[E→. a, ,]

State 7: [E→a.,)]
[E→a., ,]

State 8: [E→(L).,$]

State 9: [L→L,.E ,)]
[E→. (L),)]
[E→. a,)]
[L→L,.E , ,]
[E→. (L), ,]
[E→. a, ,]

State 10: [E→(L.),)]
[E→(L.), ,]
[L→L.,E ,)]
[L→L.,E , ,]

State 11: [L→L,E. ,)]
[L→L,E. , ,]

State 12: [E→(L).,)]
[E→(L)., ,]

5.2

b. LR(1) parsing table

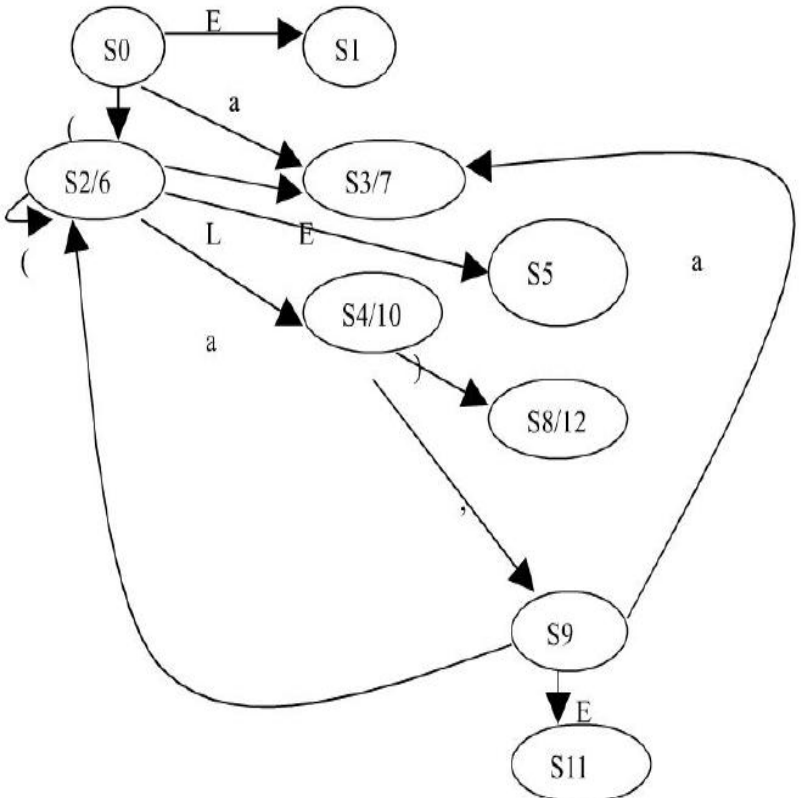
b. r1: E→(L)

r2: E→ a r3: L→L,E r4: L→E

State	Input					Goto	
	(	a	)	,	$	L	E
0	S2	S3					1
1					Accept		
2	S6	S7				4	5
3					r2		
4			S8	S9			
5			r4	r4			
6	S6	S7				10	5
7			r2	r2			
8					r1		
9	S6	S7					11
10			S12	S9			
11			r3	r3			
12			r1	r1			

5.2

c. DFA of LALR(1) items



State 0: [E'→.E,$]
[E→. (L),$]
[E→. a,$]

State 1: [E'→E.,$]

State 2/6: [E→(.L),$/)/,]
[L→. L,E ,)]
[L→. E ,)]
[L→. L,E , ,]
[L→. E , ,]
[E→. (L),)]
[E→. a,)]
[E→. (L), ,]
[E→. a, ,]

State 3/7: [E→a.,$/)/,]

State 4/10: [E→(L.),$/)/,]
[L→L.,E ,)]
[L→L.,E , ,]

State 5: [L→E. ,)]
[L→E. , ,]

State 8/12:[E→(L).,$/)/,]

State 9: [L→L,.E ,)]
[E→. (L),)]
[E→. a,)]
[L→L,.E , ,]
[E→. (L), ,]
[E→. a, ,]

State 11: [L→L,E. ,)]
[L→L,E. , ,]

5.2

d. LALR(1) parsing table

d. r1: E→(L) r2: E→ a r3: L→L,E r4: L→E

State	Input					Goto	
	(	a	)	,	$	L	E
0	S2/6	S3/7					1
1					Accept		
2/6	S2/6	S3/7				4/10	5
3/7			r2	r2	r2		
4/10			S8/12	S9			
5			r4	r4			
8/12			r1	r1	r1		
9	S2/6	S3/7					11
10			S8/12	S9			
11			r3	r3			