Minimization of automata

Distinguishable state

2 states p and q of DFA are distinguishable if and only if

$\delta^*(p,w) \in F$ and $\delta^*(q,w) \notin F$ or, $\delta^*(p,w) \notin F$ and $\delta^*(q,w) \in F$

'w' is a string.

F : final state

δ^* : extended transition function

Non-distinguishable state

2 states p and q in DFA are non-distinguishable iff

$\delta^*(p,w) \in F$ and $\delta^*(q,w) \in F$ or $\delta^*(p,w) \notin F$ and $\delta^*(q,w) \notin F$.

TABLE FILLING ALGORITHM

1) For each pair (p,q) where $p \in Q$ and $q \in Q$, if $p \in F$ and $q \notin F$ or vice versa, then the pair (p,q) is distinguishable and mark the pair (p,q).

2) For each pair (p,q) and for each $a \in \Sigma$ (an alphabet in Σ), find $\delta(p,a) = r$ and $\delta(q,a) = s$. If the pair (r,s) is already marked as distinguishable, then pair (p,q) is also distinguishable and mark it.

Repeat step 2.

Example

	0	1
→ A	B	A
B	G	C
(C)	A	C
D	C	G
E	H	F
F	C	G
G	G	E
H	G	C

	A	B	C	D	E	F	G
B	X						
C	X	X					
D	X	X	X				
E		X	X	X			
F	X	X	X		X		
G	X	X	X	X	X	X	
H	X		X	X	X	X	X

A ≡ E

A is initial state
C is final state

here w=1 (length=1)

Note

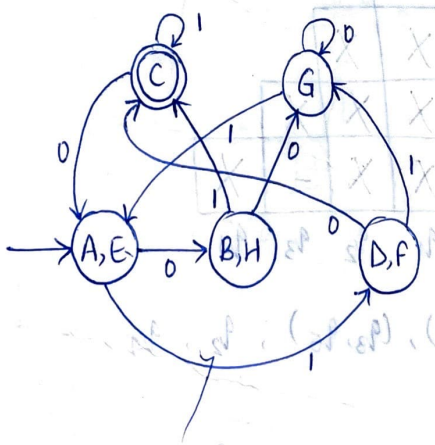
	0	1
A	B	F
G	G	E

→ already marked

∴ mark AG and BG as well.

The unmarked states are {A,E}, {B,H} and {D,F}

The states not mentioned here are C and G ⇒ make them separate states



If the transition from either A or E on the same i/p does not go to the same next state, then the table is wrong (valid for all states)

If all the intersections b/w states are marked, then the no. of states cannot be minimized.

8.

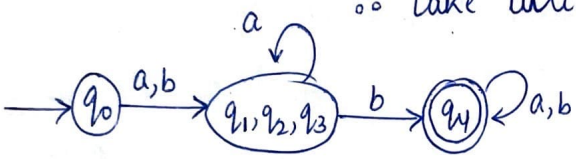
	a	b
→ q_0	q_1	q_3
q_1	q_2	q_4
q_2	q_1	q_4
q_3	q_2	q_4
q_4	q_4	q_4

q_1	X			
q_2	X	-		
q_3	X	-	-	
q_4	X	X	X	X
	q_0	q_1	q_2	q_3

(q_1, q_3), (q_1, q_2), (q_2, q_3)

share q_1 share q_2

∴ take union ⇒ (q_1, q_2, q_3)

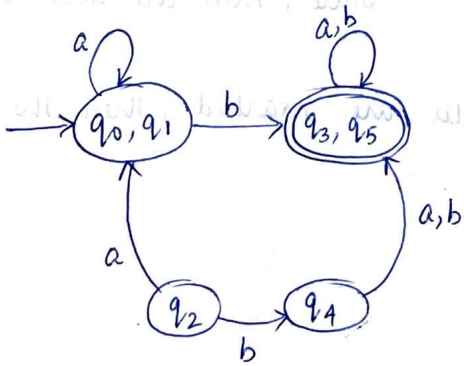


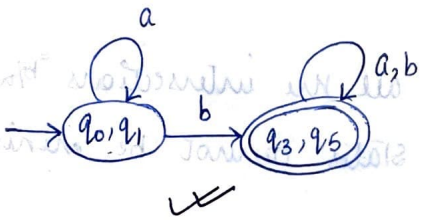
8.

	a	b
ini. → q_0	q_1	q_3
q_1	q_0	q_3
q_2	q_1	q_4
q_3	q_5	q_5
q_4	q_3	q_3
q_5	q_5	q_5

q_1	-				
q_2	X	X			
q_3	X	X	X		
q_4	X	X	X	X	
q_5	X	X	X	-	X
	q_0	q_1	q_2	q_3	q_4

(q_0, q_1), (q_3, q_5), q_2, q_4





is wrong.

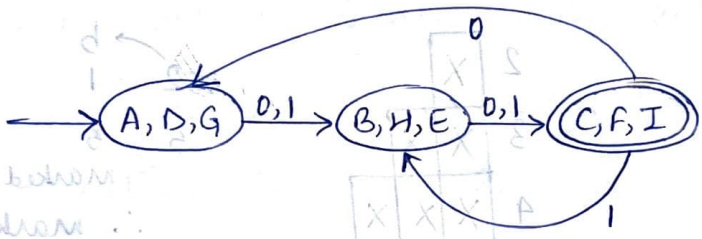
States q_2 and q_4 are unreachable / trap states.

Q.

	0	1
→ A	B	E
B	C	F
C	D	H
D	E	H
E	F	I
F	G	B
G	H	B
H	I	C
I	A	E

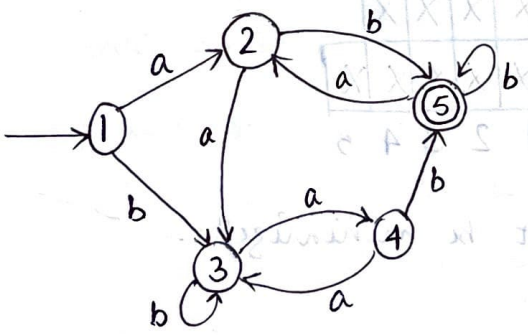
	A	B	C	D	E	F	G	H
B	X							
C	X	X						
D		X	X					
E	X		X	X				
F	X	X		X	X			
G		X	X		X	X		
H	X		X	X		X	X	
I	X	X		X	X		X	X

(A,G) (A,D) (B,H) (B,E) (C,I) (C,F) (D,G) (E,H) (F,I)

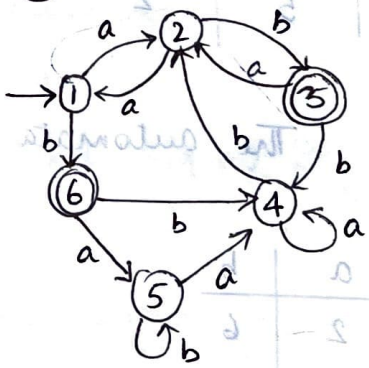


Q.

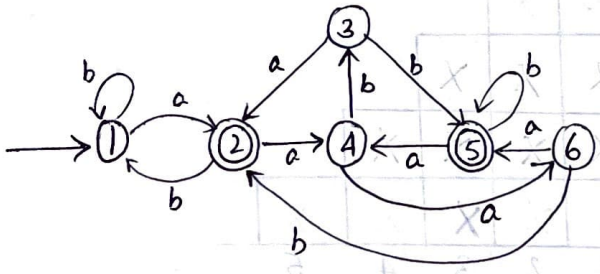
①



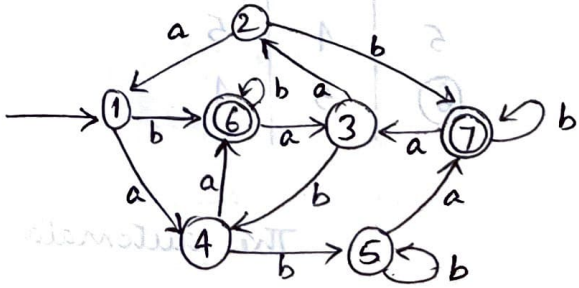
③



②



④

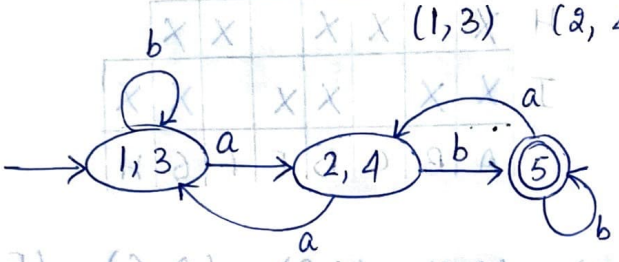


① ~

	a	b
→ 1	2	3
2	3	5
3	4	3
4	3	5
5	2	5

	1	2	3	4
2	X			
3	-	X		
4	X	-	X	
5	X	X	X	X

(1, 3) (2, 4) 5



② ~

	a	b
→ 1	2	1
2	4	1
3	2	5
4	6	3
5	4	5
6	5	2

	1	2	3	4	5
2	X				
3	X	X			
4	X	X	X		
5	X	X	X	X	
6	X	X	X	X	X

b
2 1
5 5
Marked
∴ mark 2 & 5

The automata cannot be minimized.

③ ~

	a	b
→ 1	2	6
2	1	3
3	2	4
4	4	2
5	4	5
6	5	4

	1	2	3	4	5
2	X				
3	X	X			
4	X	X	X		
5	X	X	X	X	
6	X	X	X	X	X

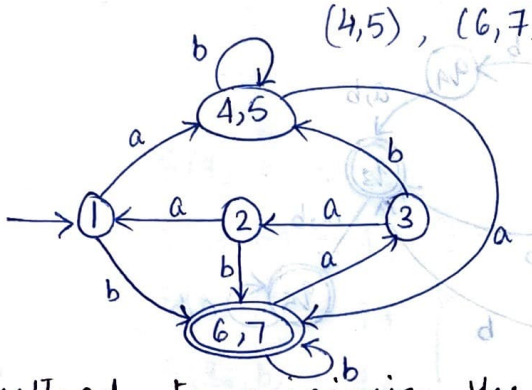
The automata cannot be minimized.

(4)

	a	b
→ 1	4	6
2	1	7
3	2	4
4	6	5
5	7	5
6	3	6
7	3	7

	1	2	3	4	5	6
2	X					
3	X	X				
4	X	X	X			
5	X	X	X	•		
6	X	X	X	X	X	
7	X	X	X	X	X	•

(4,5), (6,7), 1, 2, 3



* Another method to minimize the automata :

Step 1) Remove all unreachable states from the start state

Step 2) Construction of X : Partition the states to 2 groups, one consisting of final states and the other consisting of non-final states.

Step 3) Construct Y from X as shown below :

Copy X to Z for each a in Σ do

For each subgroup G in Z, find $\delta(q,a)$ for each q in G on input symbol a.

Suppose $q_i, q_j, \dots q_n$ are states in subgrp. G and $p_i, p_j, \dots p_n$ are the resulting states on an input symbol a if $p_i, p_j, \dots p_n$ all belong to one specific subgroup in x, then there is no need of partition.

If (p_i, p_l), ~~(p_j, p_k)~~ belongs to one group, (p_j, p_k) belongs to another group and (p_n, p_m) belongs to some another grp., then partition the subgroup G into groups (q_i, q_l), (q_j, q_k) and (q_n, q_m).

END FOR

Let the resulting subgroup be Z.

END FOR

Copy the final Z to Y.

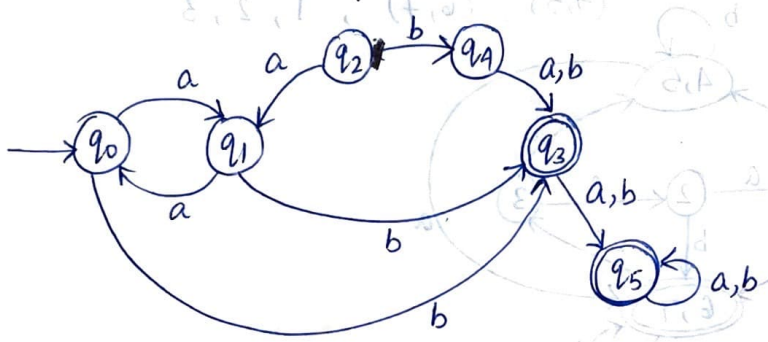
Step 4) If $Y \neq X$, copy Y to X and goto step 3.

Step 5) Pick a representative from each group Z. These representatives will be the states of reduced DFA.

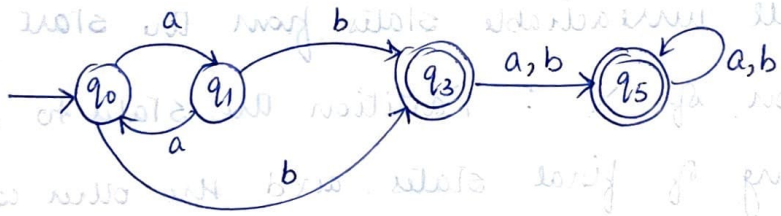
Obtain the transition diag. for reduced DFA.

All states unreachable from start state are unreachable states.

Eg :



Step 1)



Step 2) $G_1 = (q_0, q_1)$ $G_2 = (q_3, q_5)$

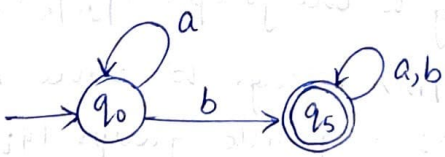
Step 3)

$\delta(q_0, a) = q_1$	$\delta(q_3, a) = q_5$
$\delta(q_1, a) = q_0$	$\delta(q_5, a) = q_5$
$\delta(q_0, b) = q_3$	$\delta(q_3, b) = q_5$
$\delta(q_1, b) = q_3$	$\delta(q_5, b) = q_5$

All these partitions lead to the same group.

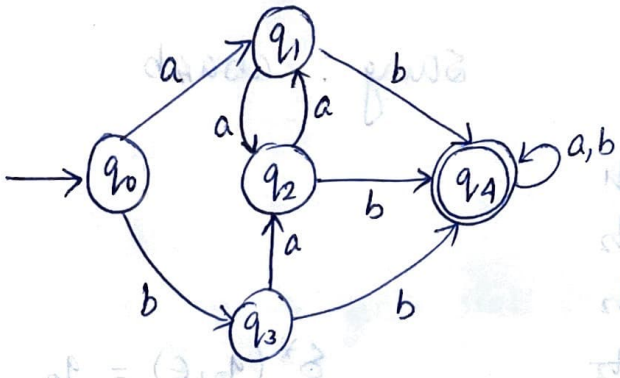
Step 4) –

Step 5)



Q.

	a	b
q_0	q_1	q_3
q_1	q_2	q_4
q_2	q_1	q_4
q_3	q_2	q_4
q_4	q_4	q_4



$G_1 = (q_0, q_1, q_2, q_3)$

$G_2 = (q_4)$

$\delta(q_0, a) = q_1$ $\quad$ $\delta(q_0, b) = q_3$ $\quad$ $\delta(q_4, a) = q_4$

$\delta(q_1, a) = q_2$ $\quad$ $\delta(q_1, b) = q_4$ $\quad$ $\delta(q_4, b) = q_4$

$\delta(q_2, a) = q_1$ $\quad$ $\delta(q_2, b) = q_4$

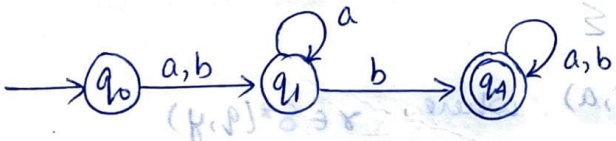
$\delta(q_3, a) = q_2$ $\quad$ $\delta(q_3, b) = q_4$

$\therefore G_{11} = (q_0)$, $G_{12} = (q_1, q_2, q_3)$

$\delta(q_0, a) = q_1$ $\quad$ $\delta(q_1, a) = q_2$ $\quad$ $\delta(q_1, b) = q_4$

$\delta(q_0, b) = q_3$ $\quad$ $\delta(q_2, a) = q_1$ $\quad$ $\delta(q_2, b) = q_4$

$\delta(q_3, a) = q_2$ $\quad$ $\delta(q_3, b) = q_4$



Extended transition function for DFA (δ^*)

(accepts a string)

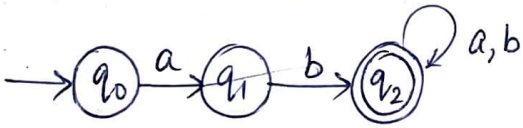
Let $M = (Q, \Sigma, \delta, q_0, F)$ be a finite automata.

we define the func. δ^* as follows, $(\delta^* : Q \times \Sigma^* \to Q)$

1) For any $q \in Q$, $\delta^*(q, \epsilon) = q$

2) For any $q \in Q$, $y \in \Sigma^*$ and $a \in \Sigma$,

$$\delta^*(q, ya) = \delta(\delta^*(q, y), a)$$



String : abaab

$\delta(q_0, a) = q_1$
$\delta(q_1, b) = q_2$
$\delta(q_2, a) = q_2$
~~$\delta(q_2, a) = q_2$~~
$\delta(q_2, b) = q_2$

$\delta^*(q_0, \epsilon) = q_0$

$\delta^*(q_0, a) = \delta(\delta^*(q_0, \epsilon), a) = \delta(q_0, a) = q_1$

$\delta^*(q_0, ab) = \delta(\delta^*(q_0, a), b) = \delta(q_1, b) = q_2$

$\delta^*(q_0, aba) = \delta(\delta^*(q_0, ab), a) = \delta(q_2, a) = q_2$

$\delta^*(q_0, abab) = \delta(\delta^*(q_0, aba), b) = \delta(q_2, b) = \underline{\underline{q_2}}$

final state

As the final state is achieved,
thus the string is accepted by the automata.

Extended transition function for NFA

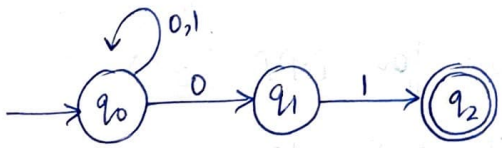
Let $M = (Q, \Sigma, \delta, q_0, F)$, be a NFA, then the function,

$\delta^* : Q \times \Sigma^* \to 2^Q$ is defined as follows,

1) For any $q \in Q$, $\delta^*(q, \epsilon) = \{q\}$
2) For any $q \in Q$, $y \in \Sigma^*$, $a \in \Sigma$,

$$\delta^*(q, ya) = \bigcup \delta(r, a) \quad \text{where, } r \in \delta^*(q, y)$$

Eg :



String : 0001

$\delta^*(q_0, \epsilon) = q_0$

$\delta^*(q_0, 0) = \delta(q_0, 0) = \{q_0, q_1\}$

$\delta^*(q_0, 00) = \delta(q_0, 0) \cup \delta(q_1, 0) = \{q_0, q_1\} \cup \phi = \{q_0, q_1\}$

$\delta^*(q_0, 000) = \delta(q_0, 0) \cup \delta(q_1, 0) = \{q_0, q_1\}$

$\delta^*(q_0, 0001) =$

String : 0101

$\delta^*(q_0, \epsilon) = q_0$

$\delta^*(q_0, 0) = \delta(q_0, 0) = \{q_0, q_1\}$

$\delta^*(q_0, 01) = \delta(q_0, 1) \cup \delta(q_1, 1) = \{q_0, q_2\}$

$\delta^*(q_0, 010) = \delta(q_0, 0) \cup \delta(q_2, 0) = \{q_0, q_1\}$

$\delta^*(q_0, 0101) = \delta(q_0, 1) \cup \delta(q_1, 1) = \{q_0, \underline{q_2}\}$ → final state is reached

Regular languages & expressions

A language is said to be regular if a finite automata or a regular expression can be written for it.

Eg : $L = \{(a+b)^n ab (a+b)^m ; n, m \geq 0\} \Rightarrow (a+b)^* ab (a+b)^*$

↳ Reads substring 'ab'

Regular expression

- ϕ (null)
- ϵ (empty string)
- Any alphabet in Σ
- Any alphabet with operations union (+), concatenation (·) or closure (*)

$R_1^* \rightarrow$ Kleene's closure

Note : (1) $a + \phi = a$ (3) $a \cdot \epsilon = a$

(2) $a \cdot \phi = \phi$ (4) $a + \epsilon = a + \epsilon$

Regular expression to accept,

(1) one occurence of a → a $L = \{a\}$

(2) one or more occurences : $aa^* \simeq a^+$ $L = \{a, aa, aaa \ldots\}$

(3) zero " " " of ab → $(ab)^*$ $L = \{\epsilon, ab, abab, ababab, \ldots\}$

(4) alternate occurences of a and b : $(ab)^*(a+\epsilon) + (ba)^*(b+\epsilon)$

only $\underbrace{(ab)^* + (ba)^*}$ does not read strings like $\underbrace{aba}_{\text{odd length}}$ or bab

↳ valid when i/p size is even

$L = \{\epsilon, ab, ba, aba, bab, \ldots\}$

$L = \{ (a+b)^n \, ab \, (a+b)^m ; n, m \geq 2 \}$

Reg. exp^n : $(a+b)(a+b)^+ \, ab \, (a+b)(a+b)^+$

Q. Find the reg. exp^n(s) for the following :

(1) String with 0s & 1s with ~~g~~even length

(2) String with odd no. of 1s

(3) String of 0s & 1s with length 6 or less

(4) String of 0s & 1s ~~not~~ ending in 1 and not containing consecutive

(5) Atleast 2 a(s)

(6) Exactly 2 a(s)

(7) Atleast one a and atleast one b

(8) Strings of 0s & 1s with odd length

(9) " " " " " atleast 3 consecutive 0s

(10) " " " " " no 2 consecutive 0s

(11) " " as and bs with more than 2 letters but beginning & ending with same letter.

(12) Strings ending with 11 .

(13) Strings with next to last symbol 0 . (second last symbol is 0)

① ~ $((0+1)(0+1))^* = (00+01+10+11)^*$

② ~ $1(1 \; 1)^*$ → always gives odd no. of 1s

ans : $0^* 1 0^* (0^* 1 0^* 1 0^*)^* 0^*$ // 0s can come anywhere

③ ~ $(0+1+\epsilon)(0+1+\epsilon)(0+1+\epsilon)(0+1+\epsilon)(0+1+\epsilon)(0+1+\epsilon)$ OR, $(0+1+\epsilon)^6$

④ ~ $(01+1)^+$ $\quad (01+1)^*(1+\epsilon)$ → accepts 'ϵ' also, hence wrong

⑤ ~ $(a+b)^* a (a+b)^* a (a+b)^*$

⑥ ~ $b^* a b^* a b^*$

⑦ ~ $(a+b)^* a (a+b)^* b (a+b)^* + (a+b)^* b (a+b)^* a (a+b)^*$

⑧ ~ $(0+1)((0+1)(0+1))^*$

⑨ ~ $(0+1)^* (000)^+ (0+1)^*$

⑩ ~ $(01+1)^*(0+\epsilon)$ OR $(0+\epsilon)(10+1)^*$

⑪ ~ $a(a+b)^+ a + b(a+b)^+ b$

⑫ ~ $(0+1)^* 11$

⑬ ~ $(0+1)^* 0 (0+1)$

Converting RE to ε-NFA

3 operations are defined on regular expressions.

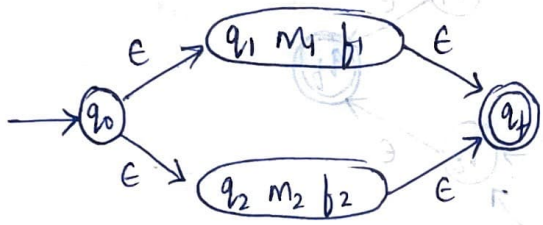
Order of precedence :

$* > \cdot > +$

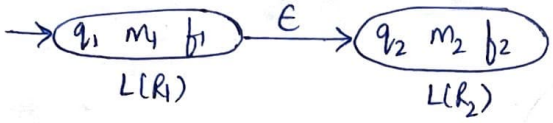
$\cdot$ ↑ concat^n

$+$ ↑ union

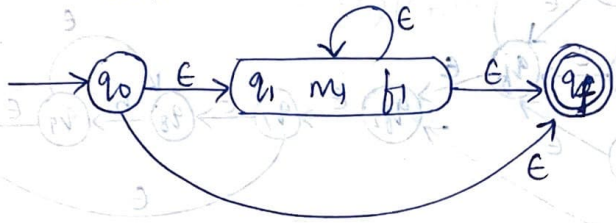
① Union $L(R_1+R_2)$



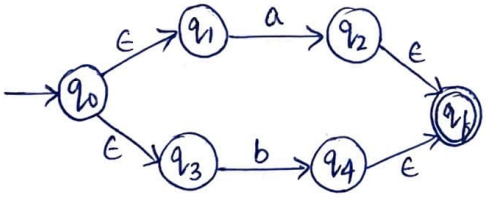
Concatenation $L(R_1, R_2)$



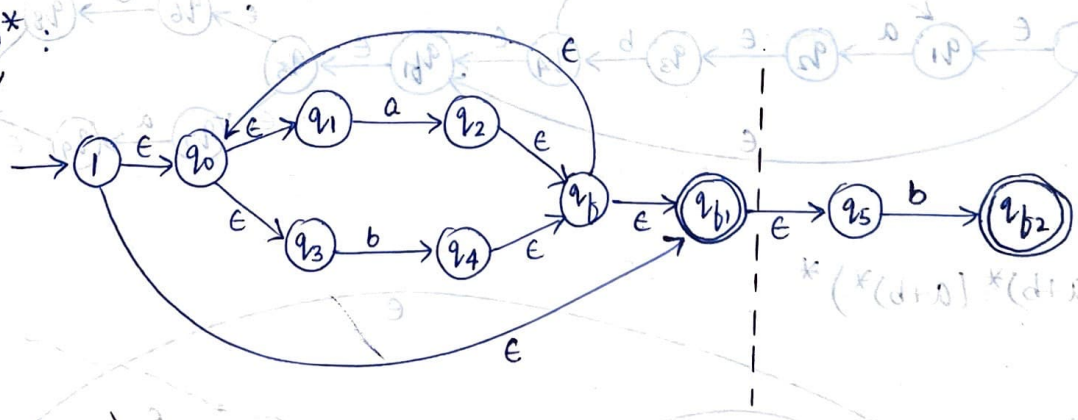
Closure $L(R^*)$



a+b :



$(a+b)^*$:



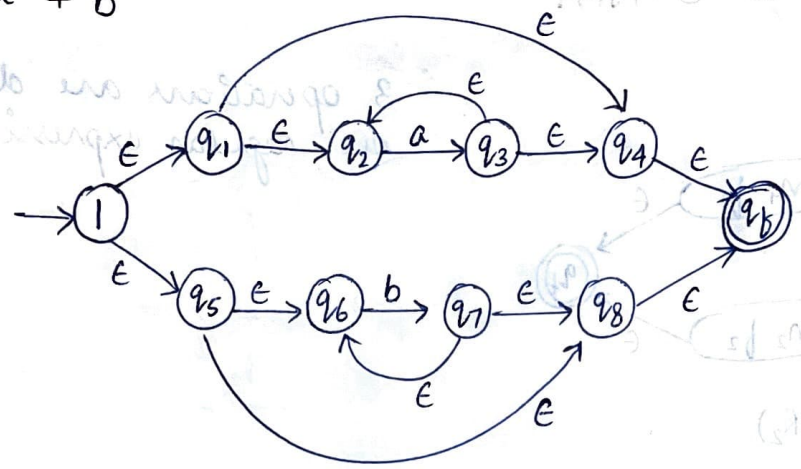
$(a+b)^*$

$(a+b)^*.b.\epsilon$

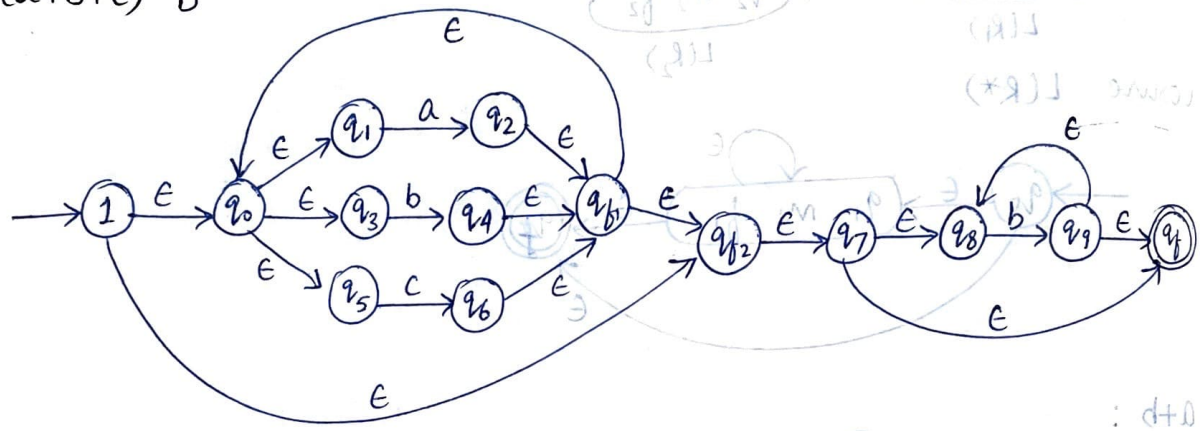
(concatenation)

$(a+b)^+ = (a+b)^*(a+b)$ or $(a+b)(a+b)^*$

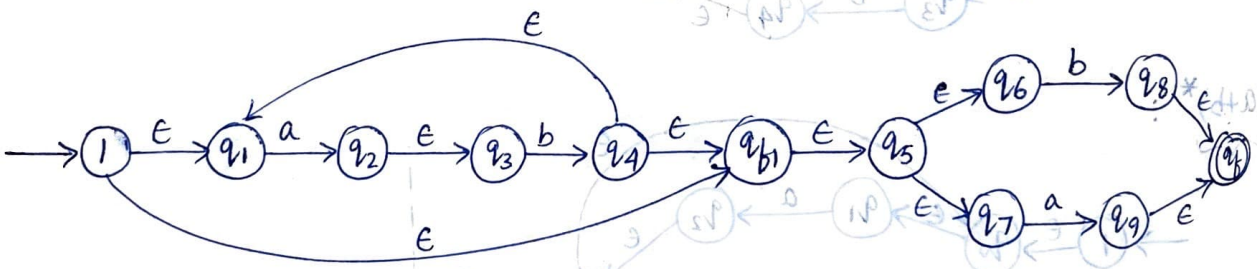
(1) $a^* + b^*$



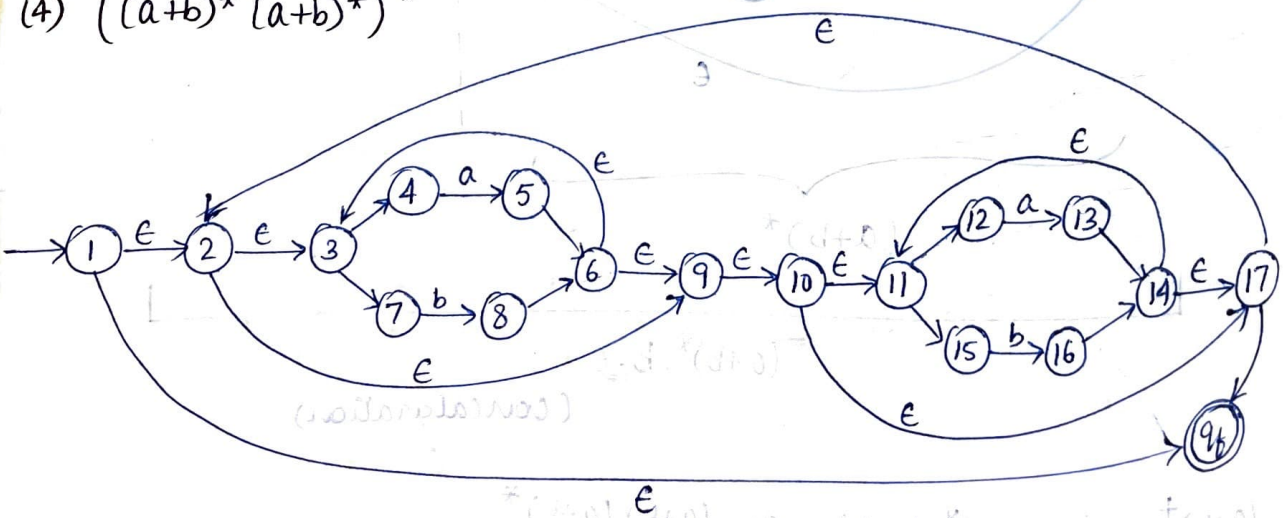
(2) $(a+b+c)^* b^*$



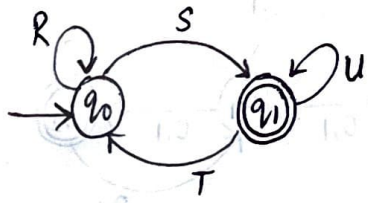
(3) $(ab)^* (b+a)$



(4) $((a+b)^* (a+b)^*)^*$



Conversion from FA to RE

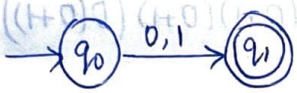


$$(R + SU^*T)^* SU^* \Leftarrow$$

2 techniques
- $R1^+$ → (1) State elimination method
- → (2) Kleen's theorem

State elimination method

Eg: ①

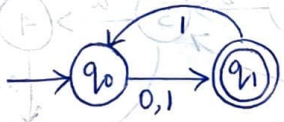


here, $R = \epsilon$
$S = 0+1$
$T = \phi$
$U = \epsilon$

Thus,

$$RE = (\epsilon + (0+1) \cdot \epsilon^* \cdot \phi)(0+1)\epsilon^*$$
$$= \epsilon(0+1)\epsilon^*$$
$$= \epsilon^*(0+1)\epsilon^* = \epsilon(0+1)\epsilon = (0+1)$$

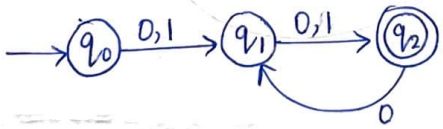
②



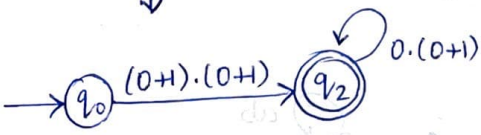
here, $R = \epsilon$, $S = 0+1$, $T = 1$, $U = \epsilon$

$$RE = (\epsilon + (0+1) \cdot \epsilon^* \cdot 1)^* (0+1)\epsilon^*$$
$$= (\epsilon + (0+1) \cdot 1)^* (0+1) = ((0+1) \cdot 1)^* (0+1)$$

③



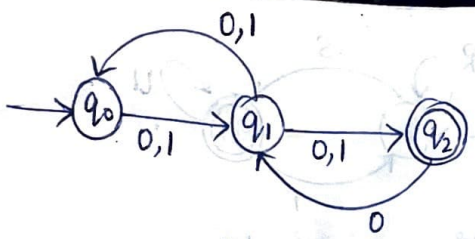
↓

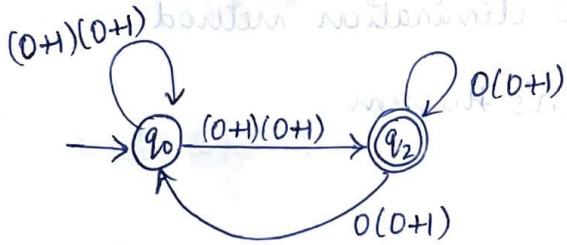


$R = \epsilon$
$S = (0+1) \cdot (0+1)$
$T = \phi$
$U = 0 \cdot (0+1)$

$$(\epsilon + (0+1) \cdot (0+1)(0 \cdot (0+1))^* \phi)^* (0+1)(0+1)(0 \cdot (0+1))^*$$
$$= (0+1) \cdot (0+1) \cdot (0(0+1))^*$$

④

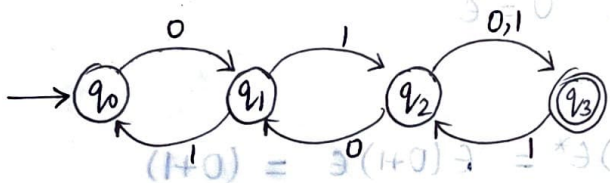




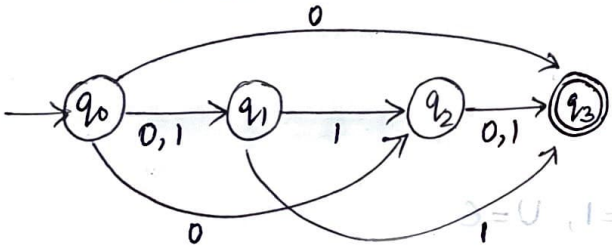
RE : $\left((0+1)(0+1) + (0+1)(0+1)(0\cdot(0+1))^{*}\cdot 0(0+1)\right)^{*}\ (0+1)(0+1)\left(0(0+1)\right)^{*}$

Ques.

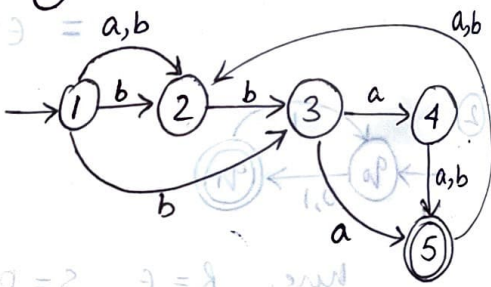
①



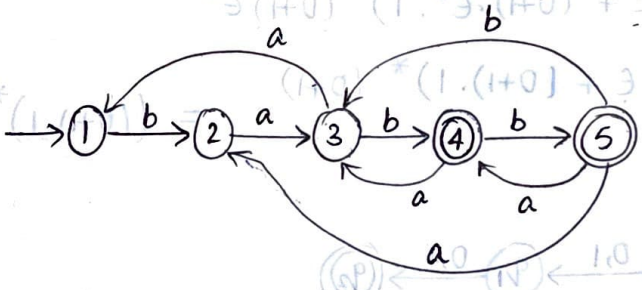
②



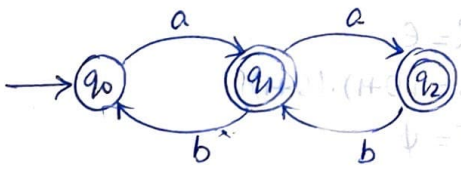
③

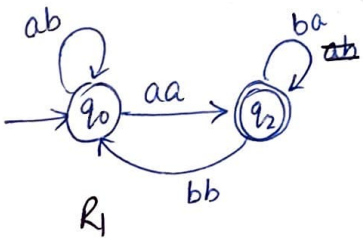


④

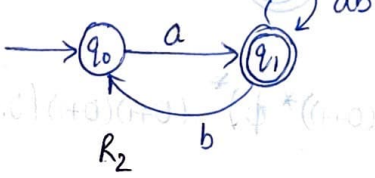


Eg:





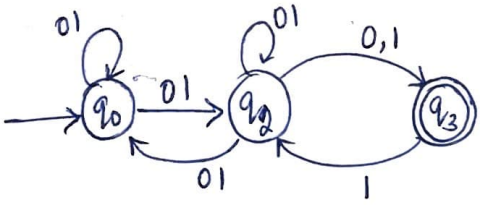
R_1



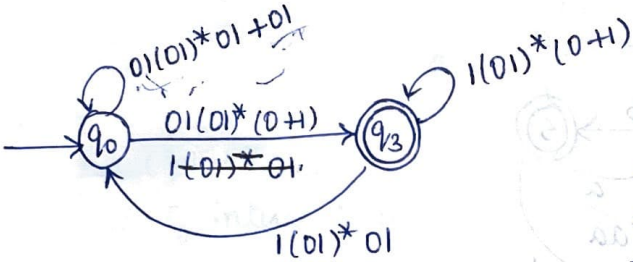
R_2

$R = R_1 + R_2$

① Eliminate q_1



Eliminate q_2



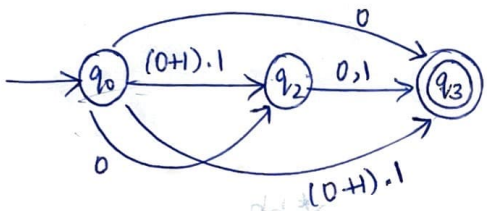
$R = 01(01)^*01 + 01$

$S = 01(01)^*(0+1)$

$T = 1(01)^*(01)$

$U = 1(01)^*(0+1)$

② Eliminate q_1



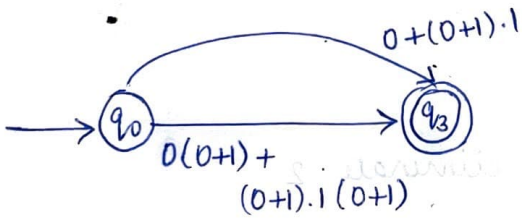
$R = \epsilon$

$S = 0 + (0+1)1 + 0(0+1) + (0+1).1(0+1)$

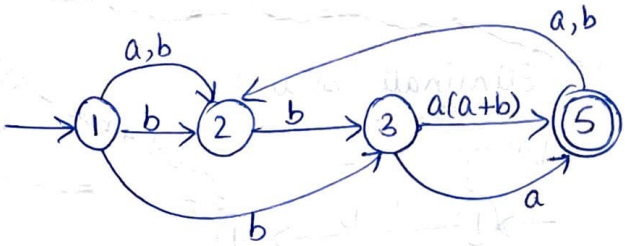
$T = \phi$

$U = \epsilon$

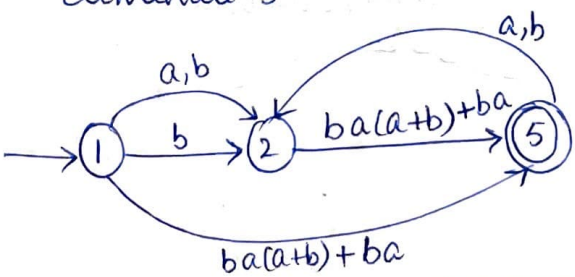
Eliminate q_2



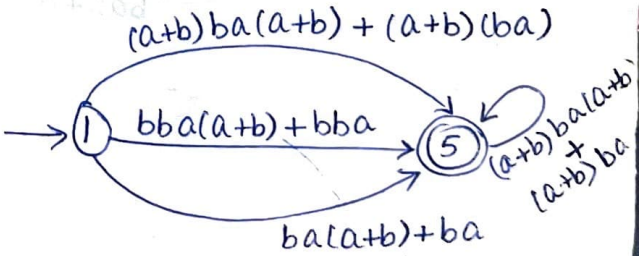
③ Eliminate 4



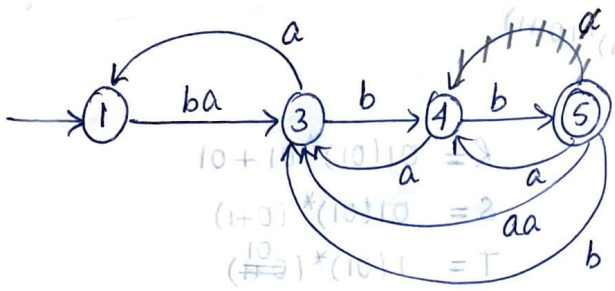
Eliminate 3



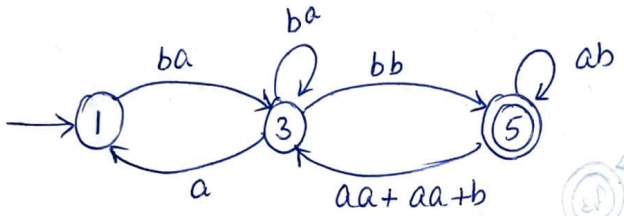
Eliminate 2



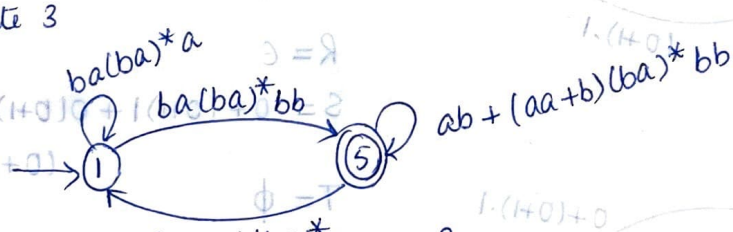
(A)~ (i) Make 4 as non-final state and eliminate 2



Eliminate 4

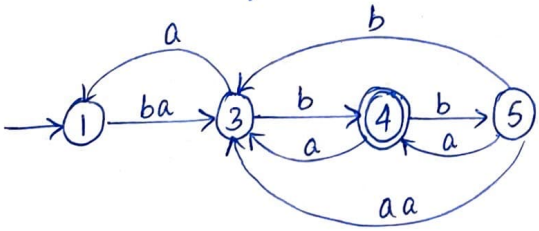


Eliminate 3

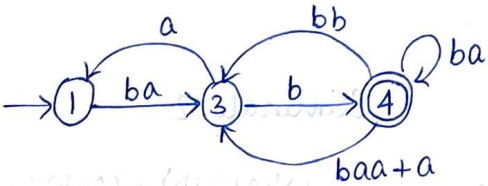


R_1

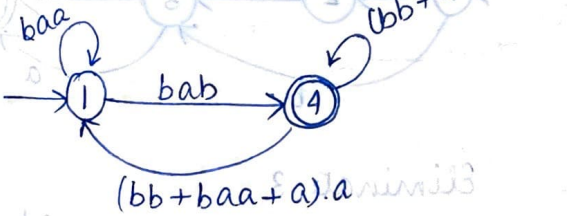
(ii) Make 5 as non-final state and eliminate 2



Eliminate state 5



Eliminate state 3

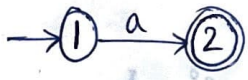


27/02/13

Convert FA to RE (Kleen's theorem)

$$R_{ij}^{(K)} = R_{ij}^{(K-1)} + R_{iK}^{(K-1)} \left(R_{KK}^{(K-1)}\right)^* R_{Kj}^{(K-1)}$$

On direct transition from one state to another : $K=0$



$R_{11}^0 = \epsilon$ $\quad R_{21}^0 = \phi$

$R_{12}^0 = a$ $\quad R_{22}^0 = \epsilon$

$i=1$ (ini. state)

$j=2$ (final state)

K : no. of intermediate states including ini. & final states $= 2$

$\therefore$ ~~R_{22}~~ $R_{12}^2 = R_{12}^1 + R_{12}^1 (R_{22}^1)^* R_{22}^1$

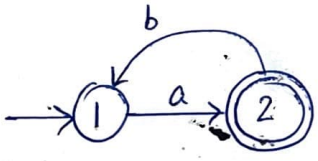
where,

$R_{12}^1 = R_{12}^0 + R_{11}^0 (R_{11}^0)^* R_{12}^0 = a + \epsilon . \epsilon^* . a = a + a = a$

$R_{22}^1 = R_{22}^0 + R_{21}^0 (R_{11}^0)^* R_{12}^0 = \epsilon + \phi = \phi$

Thus,

$R_{12}^2 = a + a(\epsilon)^* \epsilon = a + a = \underline{\underline{a}}$



$R_{11}^0 = \epsilon$ $\quad R_{12}^0 = a$ $\quad R_{21}^0 = b$ $\quad R_{22}^0 = \epsilon$

$R_{12}^2 = R_{12}^1 + R_{12}^1 (R_{22}^1)^* R_{22}^1$

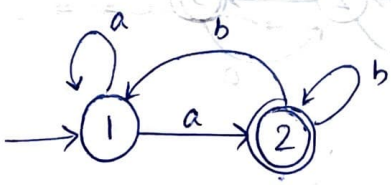
where,

$R_{12}^1 = R_{12}^0 + R_{11}^0 (R_{11}^0)^* R_{12}^0 = a + \epsilon . \epsilon^* . a = a$

$R_{22}^1 = R_{22}^0 + R_{21}^0 (R_{11}^0)^* R_{12}^0 = \epsilon + b . \epsilon^* . a = ba + \epsilon$

so, $R_{12}^2 = a + a(ba+\epsilon)^* (ba+\epsilon)$

$= a + a(ba)^* (ba+\epsilon) = a(ba)^*$



$R_{11}^0 = \epsilon + a$ $\quad R_{12}^0 = a$ $\quad R_{21}^0 = b$ $\quad R_{22}^0 = \cancel{b}\ \epsilon + b$

$R_{12}^1 = R_{12}^0 + R_{11}^0 (R_{11}^0)^* R_{12}^0 = \cancel{a + a.(a)^*.a}$

$a + (\epsilon+a).(\epsilon+a)^*.a = a + aa^* = aa^*$

$R_{22}^1 = R_{22}^0 + R_{21}^0 (R_{11}^1)^* R_{12}^0 = (\epsilon+b) + b.(a+\epsilon)^*.a = (b+\epsilon) + ba^*a$

$= b + \epsilon + ba^*a$

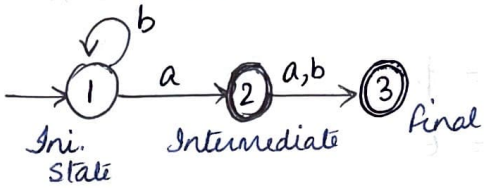
$R_{12}^2 = aa^* + aa^* (b+\epsilon+ba^*a)^* (b+\epsilon+ba^*a)$

or,

$$R_{12}^{2} = aa^* + aa^*(b + ba^*a)^*$$
$$= aa^*(b + ba^*a)^*$$
$$= a^+(b + ba^+)^*$$

(4) Q.

(1)



$R_{11}^{0} = b + \epsilon$ $\quad R_{13}^{0} = \phi$

$R_{12}^{0} = a$ $\quad R_{31}^{0} = \phi$ $\quad R_{33}^{0} =$

$R_{21}^{0} = \phi$ $\quad R_{23}^{0} = a + b$

$R_{22}^{0} = \epsilon$ $\quad R_{32}^{0} = \phi$

$$R_{13}^{3} = R_{13}^{2} + R_{13}^{2}(R_{33}^{2})^* R_{33}^{2}$$

where,

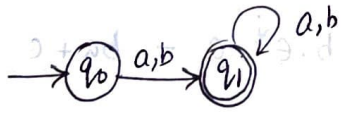
$$R_{13}^{2} = R_{13}^{1} + R_{12}^{1}(R_{22}^{1})^* R_{23}^{1}$$

$$R_{13}^{1} = R_{13}^{0} + R_{11}^{0}(R_{11}^{0})^* R_{13}^{0} = \phi + (b+\epsilon).(b+\epsilon)^*.\phi = \phi$$

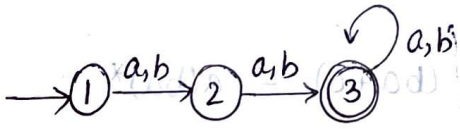
$$R_{12}^{1} = R_{12}^{0} + R_{11}^{0}(R_{11}^{0})^* R_{12}^{0} = a + (b+\epsilon)(b+\epsilon)^*.\phi = a + \phi = a$$

Q. Convert FA to RE using Kleen's theorem.

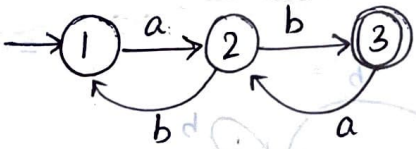
(1)



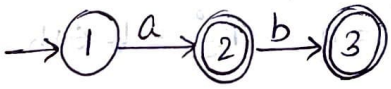
(2)



(5)

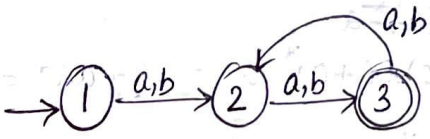


(3)



$R = R_{12}^{2} + R_{13}^{3}$

(4)



R_{13}^{3}

③ $R_{11}^0 = \epsilon$ $\quad R_{32}^0 = \phi$

$R_{12}^0 = a$ $\quad R_{33}^0 = \epsilon$

$R_{21}^0 = \phi$ $\quad R_{13}^0 = \phi$

$R_{22}^0 = \epsilon$ $\quad R_{31}^0 = \phi$ $\qquad R = R_{12}^2 + R_{13}^3$

$R_{23}^0 = b$

$$R_{12}^2 = R_{12}^1 + R_{12}^1 (R_{22}^1)^* R_{22}^1$$

where,

$$R_{12}^1 = R_{12}^0 + R_{11}^0 (R_{11}^0)^* R_{12}^0$$

$$= a + \epsilon.\epsilon^*.a = a + a = a$$

$$R_{22}^1 = R_{22}^0 + R_{21}^0 (R_{11}^0)^* R_{12}^0 = \epsilon + \phi(\epsilon)^*.a = \epsilon + \phi = \epsilon$$

$$R_{12}^2 = a + a.\epsilon^*.\epsilon = a + a = a$$

$$R_{13}^3 = R_{13}^2 + R_{13}^2 (R_{33}^2)^* (R_{33}^2)$$

where,

$$R_{13}^2 = R_{13}^1 + R_{12}^1 (R_{22}^1)^* R_{23}^1$$

$$= \phi + a(\epsilon)^* b = ab$$

$$R_{13}^3 = ab + ab(\epsilon)^* \epsilon = ab$$

$$R = a + ab$$

① $R_{11}^0 = \epsilon$ $\quad R_{22}^0 = a+b$

$R_{12}^0 = a+b$ $\quad R_{21} = \phi$

$$R = (a+b)(a+b)^*$$

$R_{p1}^0 = \epsilon$

$R_{12}^0 = a+b$

$R_{21}^0 = \phi$

$R_{22} = a+b+\epsilon$

$R_{12}^{(2)} = R_{ij}^{(k-1)} + R_{ik}^{(k-1)} R_{kk}^{(k-1)} R_{kj}^{(k-1)}$

Properties of regular languages:

(1) Union (2) Intersection (3) Complementation

(4) Reversal ×(5) Difference (6) Homomorphism

(7) Inverse homomorphism

(For a regular language, we can write a FA or a RE or both.)

Suppose $M_1 = (Q_1, \Sigma, q_1, A_1, \delta_1)$ and $M_2 = (Q_2, \Sigma, q_2, A_2, \delta_2)$ accept the language L_1 and L_2 respectively.

Let M be a finite automata defined by $M = (Q, \Sigma, q_0, A, \delta)$ where $Q = Q_1 \times Q_2$ and $q_0 = (q_1, q_2)$ and the transition func. δ is defined by the formula,

$$\delta((p,q), a) = (\delta_1(p,a), \delta_2(q,a)) \text{ for any } p \in Q_1, q \in Q_2 \text{ and } a \in \Sigma$$

then,

Union

(i) If $A = \{(p,q) \mid p \in A_1 \text{ or } q \in A_2\}$, then M accepts lang. $L_1 \cup L_2$

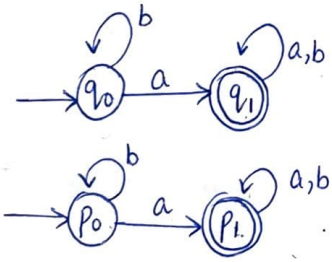
Intersection

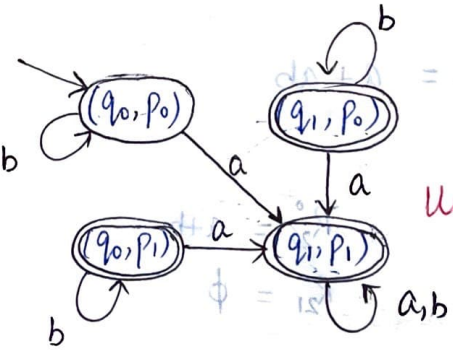
(ii) If $A = \{(p,q) \mid p \in A_1 \text{ and } q \in A_2\}$, then M accepts the lang. $L_1 \cap L_2$

Complement

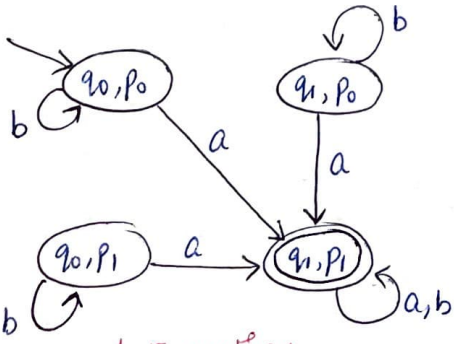
(iii) If $A = \{(p,q) \mid p \in A_1 \text{ and } q \notin A_2\}$, then M accepts language $L_1 - L_2$

Eg.

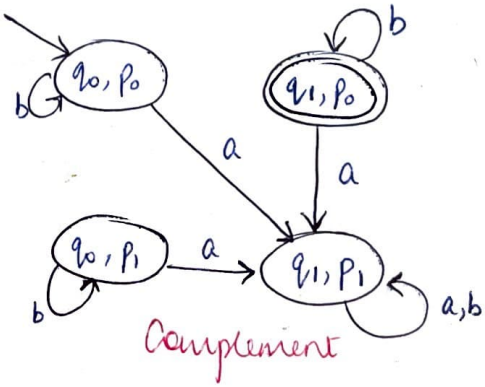




Union



Intersection



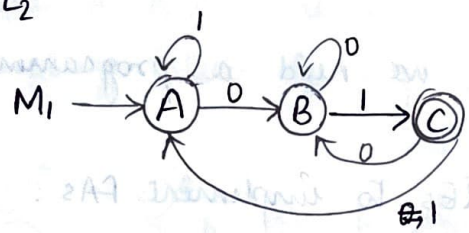
Complement

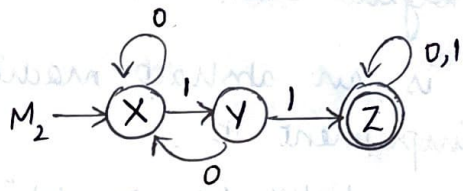
Q. Let M_1, M_2, M_3 be FAs recognizing languages L_1, L_2, L_3 respectively. Draw FAs recognizing following languages:

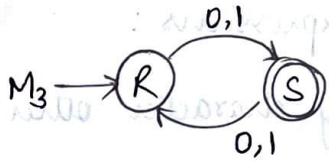
(1) $L_1 \cup L_2$ (4) $L_1 \cap L_3$

(2) $L_1 \cap L_2$ (5) $L_3 - L_2$

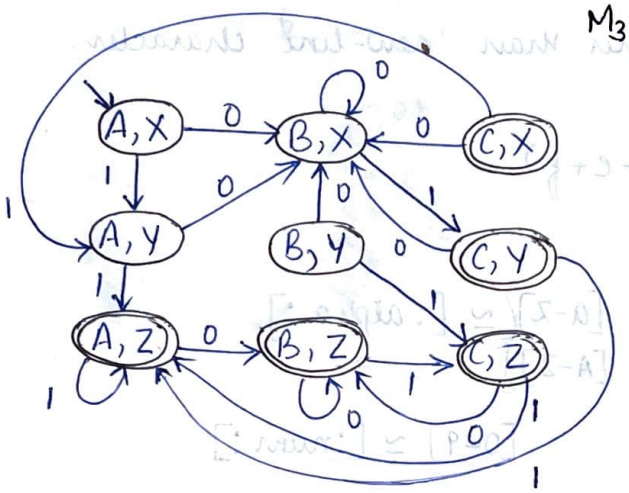
(3) $L_1 - L_2$



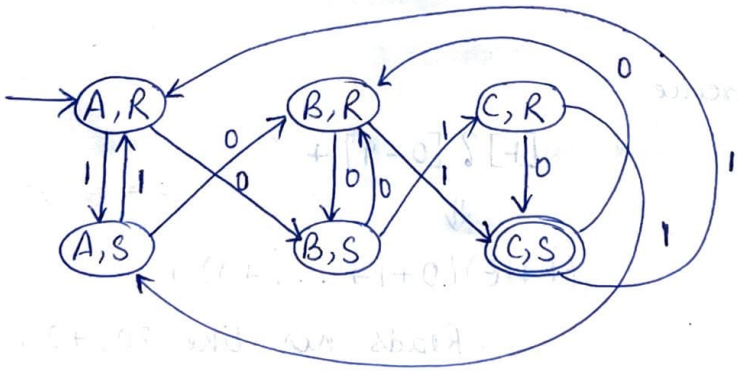




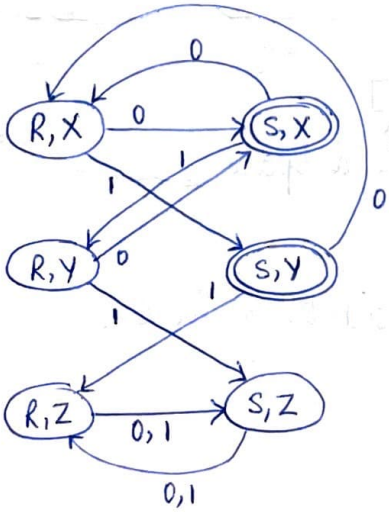
(1)~



(4)~



(5)~



* Applications of FA and RE :

FA

(1) String matching
(2) Compiler design
(3) Digital ckts.

∵ FA is an abstract machine, we need a programming lang. to implement it.

UNIX has pre-defined REs to implement FAs.

(1) UNIX regular expressions :

. → Accepts any character other than 'new-line' character.

[ ]

[abcdef] = (a+b+c+d+e+f)

[a-z]
[A-Z]
[0-9]
} special REs → [:alnum:]

[a-z] / [A-Z] } ≃ [:alpha:]

[0-9] ≃ [:num:]

[a|b] or [ab] ⇒ (a+b)

? → zero or one occurence
* → zero or more "
+ → one or more "

[+]?[0-9]+ ⇓ (+ + ∈)(0+1+ . . . +9)+

unsigned +ve integer

↑ OR

Reads nos like 20, +2, +20 etc.

(2) Lexical analysis

RE to accept keyword int : int[ ]

↑ for a space (dont use ∈)

(3) String matching

strstr() → inbuilt func. to find a substring in C.

Reversal

If L is a regular lang., then L^R is also regular.

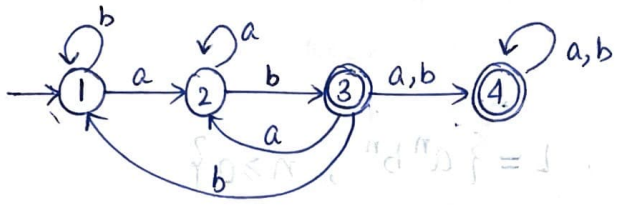
Steps

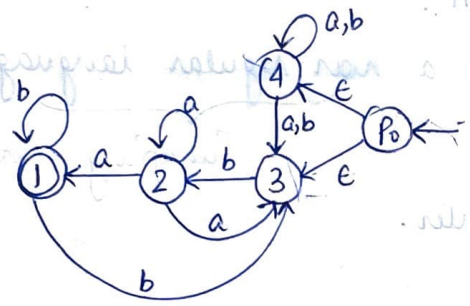
Step(1) : Reverse all the transitions

Step(2) : Swap initial and final states.

Step(3) : Create a new start state p_0 with ϵ-transitions to all accepting states of original DFA. (only when 2 or more final states)

Eg :





Homomorphism

If Σ and Γ are set of alphabets, $h: \Sigma \to \Gamma^*$

If L is a RL over alphabet Σ and h is homomorphism on Σ, then $h(L)$ is also regular.

* If $h(0) = ab$ and $h(1) = c$, then $h(00110) = ?$

$h(00110) = ababccab$

Inverse homomorphism

If h is a homomorphism for alphabet Σ to alphabet Γ and L is a language over Γ, then $h^{-1}(L)$ is also regular.

Eg: let h be a homomorphism defined by, $h(a) = 01$ & $h(b) = 10$

then $h^{-1}(\overset{1001}{\cancel{ba}}) = ?$

Ans : ~~1001~~ ba

Now, consider homomorphism h from alphabet $\{0,1,2\}$ to $\{a,b\}$ defined by, $h(0) = ab$, $h(1) = b$ and $h(2) = aa$

Find $h(0210)$, $h(2201)$ ~~and~~ $h(1^*02^*)$, $h(1+02)^*$, $h^{-1}(ab+baa)^*$ and $h^{-1}(ababb)$

$h(0210) = abaabab$

$h(2201) = aaaaabb$

$h(1^*02^*) = b^*ab(\cancel{ab}\,aa)^*$

$h(1+02)^* = (b+abaa)^*$

$h(ab+baa)^* = (0+12)^*$

$h^{-1}(ababb) = 001$

$h^{-1}(abab) = 00$ or 021 or 210 or 2121

$\left\{ \begin{array}{l} h(0) = ab \\ h(1) = b \\ h(2) = a \end{array} \right\}$ List all the options

Q. Draw DFA for the RE, $L = \{a^n b^n \,;\, n \geq 0\}$

This DFA cannot be drawn.

The above expression belongs to a $\underbrace{\text{non regular language}}_{\text{Pumping lemma}}$.

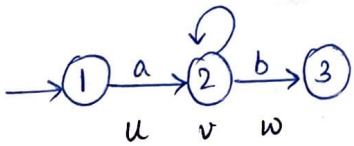
$a^*b^* \neq a^n b^n$

FA cannot act as a counter.

Pigeonhole principle

If length of string = 2, min. 3 states are reqd.

If length ↑ss and states = 3, then use self loops, and this string can be split into 3 pieces:



$\equiv uv^iw$ where $i = 0, 1, 2, \ldots$

Pumping lemma

Let $M = (Q, \Sigma, \delta, q, F)$ be a finite automata and has 'n' no. of states. Let L be a language accepted by M and assume that it is regular. Let $X \in L$ and length(X) or $|X| \geq n$, then X can be decomposed into $X = uvw$ such that, $|uv| \leq n$, $|v| \geq 0$, then $uv^iw \in L$ for $i = 0, 1, 2, \ldots$

Proof by contradiction

Let L be regular.

$|a^n b^n| = 2n > n$

$x = uv^i w$

$u = a^{n-1}$

$v = a$

$w = b^n$ here, $|uv| \leq n$ and $|v| \neq \epsilon$

$\left.\begin{array}{l} u = a^n \\ v = b \\ w = b^{n-1} \end{array}\right\}$ X

when $i = 0$,

$x = a^{n-1} . b^n \notin a^n b^n$

$\therefore$ L is non regular.

Q. Show that the following languages are non regular:

(1) $L = \{ ww^R \mid w \in (a,b)^* \}$

Let $w = a^n b^n$

Then $w^R = b^n a^n$

$\therefore |ww^R| = 4n > n$

Now, $x = uv^i w$ where, $u = a^{n-1}$, $v = a$ and $w = b^n b^n a^n$

Put $i = 0$, $x = a^{n-1} b^n b^n a^n \notin L$

(2) ~~La*~~ $L = \{ a^n b^l c^{n+l} \mid n, l \geq 0 \}$

$|a^n b^l c^{n+l}| = 2(n+l) > n$

$x = uv^i w$ where, $u = a^{n-1}$, $v = a$ and $w = b^l c^{n+l}$

Put $i = 0$, $x = a^{n-1} b^l c^{n+l} \notin L$

(3) $L = \{ ww \mid w \in (0,1)^* \}$

Let $w = 0^n 1^n$

$ww = 0^n 1^n 0^n 1^n$ and $|ww| = 4n > n$

$x = uv^i w$ where, $u = 0^{n-1}$, $v = 0$, $w = 1^n 0^n 1^n$

Put $i = 0$,

$x = 0^{n-1} 1^n 0^n 1^n \notin L$

(4) $L = \{a^n b^l \mid l > n\}$

Let $L = a^n b^{n+1}$

$|L| = n + n + 1 = 2n + 1 > n$

$x = uv^i w$ where $u = a^{n-1}$, $v = a$, $w = b^{n+1}$

Put $i = 0$,

$x = a^{n-1} b^{n+1} \notin L$

(5) $L = \{a^n b^{n+m} c^m \; ; \; n, m \geq 0\}$

$|L| = n + n + m + m = 2(n+m) > n$

$x = uv^i w$ where, $u = a^{n-1}$, $v = a$, $w = b^{n+m} c^m$

Put $i = 0$,

$x = a^{n-1} b^{n+m} c^m \notin L$