REGULAR SETS AND REGULAR GRAMMARS

Finite Automata i.e the way to represent a language by a machine or model. In this chapter we will discuss how these languages can be converted to regular expressions. The languages accepted by a finite automata are easily described by simple expressions called regular expressions. A language is regular if and only if it can be accepted by a finite automaton. In other words, a language can be generated in a simple way, from simple primitive languages, if and only if it can be recognized in a simple way, by a device with finite number of states and no auxiliary memory.

Regular Expressions

A regular expression is an exp that describes a whole set of strings, by following certain syntax rules. Regular expn are used by many text editors and utilitie to search a block of text for certain pattern eg:- to replace the found strings with some other strings.

Regular expressions consists of constants and operators that denote a set of strings and operations over these sets respectively. For finite alphabet Σ, the following constants are defined-

1) Empty set :- ϕ $\{\phi\}$

2) Empty string :- $\wedge$ $\{\wedge\}$

3) Literal character :- 'a' in Σ denoting the set $\{a\}$.

and the following operations:-

1) Concatenation :- PQ denoting the set $\{\alpha\beta : \alpha \in P, \beta \in Q\}$

for eg:- $\{a, bc\}\{de, f\}$ ~~$= \{a, bc, de, f\}$~~

$= \{ade, af, bcde, bcf\}$

2) Union :- $P \cup Q$

3) Kleene Star :- P^* denoting the smallest superset of P that contains $\wedge$ and is closed under string concatenation.

for eg:

i) $a \cup b^*$ denotes $\{a, \Lambda, b, bb, bbb, \ldots\}$

ii) $(a \cup b)^*$ denotes the set of all strings consisting of any occurence of a's and b's including empty string Λ.

Recursive Definition of Regular Expressions

1) A terminal symbol (i.e $x \in \Sigma$) is a regular expression.

2) The union of two different regular expressions R_1 & R_2 written as $R_1 + R_2$ is also a regular expression.

eg).

$R_1 = (ab)^*$

$R_2 = (a+b)^*$

$R_3 = R_1 + R_2 = (ab)^* + (a+b)^*$

Simplification of Regular Expressions (Algebraic laws)

Let us assume P, Q, R be different regular expressions. The identities are:-

1) $\phi + R = R$ eg:- $\phi + (ab)^* = (ab)^*$

2) $\phi R = R$ eg:- $\phi (ab)^* = (ab)^*$

3) $\Lambda R = R \Lambda = R$ eg $\Lambda (ab)^* = (ab)^* \Lambda = (ab)^*$

4) $\Lambda^* = \Lambda$ & $\phi^* = \Lambda$

5) $R + R = R$

6) $(R^*)^* = R^*$

7) $R^* R^* = R^*$

8) $RR^* = R^* R = \cancel{R^*} R^+$

9) $(PQ)^* P = P(QP)^*$

10) $(P+Q)^* = (P^* Q^*)^* = (P^* + Q^*)^*$

11) $(P+Q) R = PR + QR$

$R(P+Q) = RP + RQ$

12) $\Lambda + RR^* = R^* = \Lambda + R^* . R$

Algebraic laws for Regular Expression

1) **Associative law:-**

$$(x+y)+z = x+(y+z)$$

$$(xy)z = (xy)z$$

2) **Commutative law:-**

$$r_1 + r_2 = r_2 + r_1$$

3) **Identity law:-**

$$x \times 1 = 1 \times x = x$$

$$0 + x = x + 0 = x$$

i) $\phi + R = R + \phi = R$

ϕ is the identity for union.

ii) $\wedge R = R \wedge = R$

$\wedge$ is the identity for concatenation.

iii) $\phi R = R \phi = \phi$

ϕ is the annihilator for cancellation.

Annihilators :- For a operator this is the value for which when it is applied to some operator and other value, it produces annihilator. for eg:-

$$x \times 0 = 0 \times x = 0$$

0 is the annihilator for multiplication.

4) **Distributive law:-**

$$x \times (y+z) = xy + xz$$

Idempotent law:-

we call a operator idempotent, if we apply this operator to two same values and the result is that value when it given.

$a + a \neq a$ so $+$ is not idempotent.

$a \times a \neq a$

① $(0+1)^* 0 (0+01)^*$

$1 \{000, 100\}$, from 2

$2 \begin{cases} 000 \\ 010 \\ 100 \\ 110 \end{cases}$ null from 2^{nd}

001

000

② $0^*(10^+1)^* 0^4 10^*$

10001000

$(0+1)^* 001^* \Rightarrow$ ~~1000~~ 11001 (above it: 11000)

$110(1+0)^* \Rightarrow$ 11001

11000

Arden's Theorem :-

Let P & Q be two regular expressions over Σ. If P does not contain Λ, then the following equation in R i.e

$R = Q + RP$ has a unique solution given by $R = QP^*$.

Q: Prove that: i) $a^*(ab)^*(a^*(ab)^*)^* = (a+ab)^*$

$LHS = a^*(ab)^*(a^*(ab)^*)^*$

Let $P = a^*(ab)^*$

$\therefore LHS = P(P)^* = P^* = (a^*(ab)^*)^*$

$= (a+ab)^*$

$= RHS$

ii) $(1+00^*1)+(1+00^*1)(0+10^*1)^*(0+10^*1)$

$= 0^*1(0+10^*1)^*$

$LHS = (1+00^*1)+(1+00^*1)(0+10^*1)^*(0+10^*1)$

$= (1+00^*1)(\Lambda+(0+10^*1)^*(0+10^*1))$

$= (1+00^*1)(0+10^*1)^* \quad (\because \Lambda+R^*R = R^*)$

$= 1(\Lambda+00^*)(0+10^*1)^*$

$= (1\,0^*)(0+10^*1)^* \quad (\because \Lambda+RR^* = R^*)$

$= 0^*1(0+10^*1)^*$

Q:- $(1+00^*1) + (1+00^*1)(0+10^*1)^*(0+10^*1) = 0^*1(0+10^*1)^*$

Soln: $LHS = (1+00^*1) + (1+00^*1)(0+10^*1)^*(0+10^*1)$

$= (1+00^*1)(\Lambda + (0+10^*1)^*(0+10^*1))$

$= (1+00^*1)(0+10^*1)^*$ $(\because \Lambda + R^*R = R^*)$

$= 1(\Lambda + 00^*)(0+10^*1)^*$

$= 0^*1(0+10^*1)^* = RHS$

Q:- Show that $(ab)^* \neq a^*b^*$

Soln: $LHS = (ab)^* = \{\Lambda, ab, abab, ababab, \ldots\}$

$RHS = a^*b^* = \{\Lambda, a, b, ab, aa, bb, abb, aab, \ldots\}$

LHS is going to give us the string combination of ab but in RHS there is possibility of ab but not abab.

Q:- Show that $(0^*1^*)^* = (0+1)^*$

$LHS = (0^*1^*)^* =$

Q1: Exhibit the language $L(a^*(a+b))$ in set notations.

Soln: $L(a^*(a+b)) = L(a^*) \cdot L(a+b)$

$= (L(a))^* \cdot L(a+b)$

$= (L(a))^* \cdot (L(a) \cup L(b))$

$= (\Lambda, a, aa, aaa, \ldots)\{a, b\}$

$= \{a, aa, aaa, aaaa, \ldots, b, ab, aab, \ldots\}$

Q2: Exhibit the language $L((a+b)^*(a+bb))$ in set notation.

Soln: $L((a+b)^*(a+bb)) = L(a+b)^* \cdot L(a+bb)$

$= (L(a+b))^* \cdot L(a+bb)$

$= (L(a+b))^* \cdot (L(a) \cup L(bb))$

$= (L(a) \cup L(b))^* \cdot (L(a) \cup L(bb))$

$= \{\Lambda, a, b, ab, aab, abb, \ldots\}\{a, bb\}$

$= \{a, aa, ba, aba, bb, bbb, abbb, \ldots\}$

Q3: Represent the following sets by regular expressions.

i) $\{0, 1, 2\}$ ii) $\{101\}$ iii) $\{10, 01\}$

iv) $\{1^{2n+1} : n \geq 0\}$ v) $\{a^2, a^5, a^8, \ldots\}$ vi) $\{1, 11, 111, \ldots\}$

i) $\{0, 1, 2\}$ $0+1+2$

ii) $\{101\}$ 101

iii) $\{10, 01\}$ $10+01$

iv) $\{1^{2n+1} : n \geq 0\}$ $\{1, 111, 11111, \ldots\}$ $1(11)^*$

v) $\{a^2, a^5, a^8, \ldots\}$ $\{aa, aaaaa, aaaaaaaa \ldots\}$ $aa(aaa)^*$

vi) $\{1, 11, 111, \ldots\}$ $\{1\}^+$ or $1\{1\}^*$

Q4: Find the regular expression represen[ting the] set of all strings of the form:-

a) $a^m b^n c^p$ where $m, n, p \geq 1$

$m=n=p=1$ abc

$m=1$ $n=1$ $p=2$ abcc

R.E $a(a)^* b(b)^* c(c)^*$

b) $a^m b^{2n} c^{3p}$ where $m, n, p \geq 1$

$m=n=p=1$ abbccc

$m=1$ $n=2$ $p=3$ abbbbccccccccc

R.E $a(a)^* (bb)(bb)^* ccc(ccc)^*$

c) $a^n b a^{2m} b^2$ where $m \geq 0$, $n \geq 1$

$m=0$ $n=1$ abbb

$m=1$ $n=1$ abaabb

$aa^* b(aa)^* bb$

Q5: Find the sets represented by following regular expressions:-

a) $(a+b)^*(aa+bb+ab+ba)^*$ $\{a,b\}^*$

b) $(aa)^* + (aaa)^*$ $\{x \in \{a\}^*$ where x is divisible by 2 or 3$\}$

c) $(1+01+001)^*(\Lambda+0+00)$

d) $a+b(a+b)^*$ $\{a, b, ba, bb, baa, bab, bba, bbb \ldots\}$

Write ... strings ou[t] ... adjacent ... adjac[ent]

Write regular expression for the set of all strings over $\{0,1\}$ in which every pair of adjacent zero's appears before any pair of adjacent ones.

$$(01)^*(0011)^*(0\phi)^*$$

Q7: Let $\Sigma = \{a,b\}$ for each of the following languages over Σ, find a regular expression representing the following:-

a) all strings that contain exactly one 'a's.

b) all strings beginning with 'ab'.

c) all strings that contain neither the substring 'aaa' or 'bbb'.

a) b^*ab^* b) $ab(a+b)^*$ c) $(a+b)^*(aaa+bbb)(a+b)^*$

Q8: Describe the following sets by regular expression R over $\{0,1\}$ or $\{a,b\}$.

i) The set of all strings which consists of all words with exactly two a's.

$\cancel{(a+b)^* a(a+b)^* a(a+b)^*}$ or b^*ab^*ab

ii) $L_2 = \{abb, a, b, bba\}$

$R = abb + a + b + bba$

iii) L_3 = A language of all words without 'aa'.

$R = b\,(abb^*)^*\,(\Lambda + a)$

iv) L_4 = { The set of all strings of a's and b's that at some point contains a dou

$RE = [aa + bb + (ab+ba)(aa+bb)^*(ab+ba)]$

v) L_5 = The set of all strings over {0,1} having almost two 0's or almost one pair of 1's.

RE =

vi) L_6 = The set of all strings over {0,1} which has almost two 0's.

vii) L_7 = The set of all strings that ~~contain~~ consists of alternating 0's and 1's.

$RE = (01)^* + (10)^* + 1(01)^* + 0(10)^*$

viii) L_8 = The set of all strings with even no. of a's followed by an odd no. of b's.

$RE = (aa)^*(bb)^*b$

ix) L_9 = The set of all strings that has atleast one 'a' and atleast one 'b'.

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

$10 = \{ a$ double

$RE = (a$

10 = { the set of all words ending either double 'b' or single 'a'.

RE = $(a+b)^*(a+bb)$

Q9: Find the shortest string that is not in the language represented by the regular expression $a^*(ab)^*b^*$.

Length 0 = ϕ

Length 1 = a, b

Length 2 = ab, aa, bb but not ba

So ba is not in the language.

Q: Consider the set of all binary strings wherein the fifth last symbol is 0. $\Sigma = \{0,1\}$.

Solⁿ: According to the specified condition the string can contain either 0 or 1 as last four symbols. Compulsorily, we should have 0 as the fifth last symbol.

$$(1+0)^{*}.0.(0+1)(0+1)(0+1)(0+1)$$

Q: Construct Regular Expression for binary strings containing no ~~more~~ maximum than two 0's.

Solⁿ:- No more than two 0's means -

1) Having no 0 at all. — 1^{*}
2) Having exactly one 0. — $1^{*}.0.1^{*}$
3) Having exactly two 0's. — $1^{*}01^{*}01^{*}$

RegEx - $\boxed{1^{*} + 1^{*}.0.1^{*} + 1^{*}01^{*}01^{*}}$

Q: Write Regular ~~language~~ E to denote a language L which accepts all the strings which begin or end with either 00 or 11.

Solⁿ:- L_1 = the strings which begin with 00 or 11.
L_2 = the strings which end with 00 or 11.

$L_1 = (00+11)(0+1)^{*}$

$L_2 = (0+1)^{*}(00+11)$

$$R = (00+11)(0+1)^{*} + (1+0)^{*}(00+11)$$

Q: write RE which contains L having s... which should have atleast one 0 and atleast one 1.

Sol^n: The possible strings are 01, 10, 001, ..

$$RE = [(0+1)^* 0 (0+1)^* 1 (0+1)^*] + [(0+1)^* 1 (0+1)^* 0 (0+1)^*]$$

Q: Describe the language denoted by the following R.E

$r.e = (b^* (aaa)^* b^*)^*$

Sol^n: The possible strings of given RE are -

ϵ, b, aaa, b, bb aaab

L = The language consists of the strings in which a's appear tripled, there is no restriction on the no. of b's.

Constructing Regular Expressions

Q: Construct RE for all even binary numbers.

Solⁿ: For even numbers, binary representation will terminate with 0. Therefore, RegEx must have a 0 at the end, before that any combination of 0's and 1's can appear in any order.

$\therefore$ $(0+1)^*.0$

Q:- Construct RE for set of strings that begin with atleast two a's and end with an even no. of b's.

Solⁿ: RegEx for two a's — aa

RegEx for strings that begin with atleast two a's ~~and~~

$aa.(a)^*$

RegEx for strings that end with bb — bb

" " " " " even no. b's — $(bb)^*$

$\therefore$ Required RegEx $aa.(a)^*(bb)^*$

Q:- Construct RE for the language of all strings over {a,b} that begin with a's and end with b's (with nothing else in between).

Solⁿ: For the strings that have even no. of both a's & b's.

$(aa)^*.(bb)^*$

for strings that have odd no. of both a's & b's. $\boxed{aa^*bb^*}$

$a.(aa)^*\ b.(bb)^*$

Combining these two — $(aa)^*(bb)^* + a.(aa)^*\ b.(bb)^*$

②

Imp Q: Construct the regular expression for all strings with exactly one pair of consecutive 0's in them, over $\Sigma\{0,1\}$.

Constr...

ⓐ NFA

ⓑ N...

Soln: Strings accepted in this language
00, 100, 001, 101010011,

Strings not accepted by this RE are,
0, 10, 01, 010, 000 ....

00 – exactly one pair of consecutive 0's.

Anything can be present before or after consecutive 0's – as ~~(1.(00) 1)~~ long as it is not giving consecutive 0's.

RegEx for binary strings not containing consecutive 0's – $(1+(1.0))^*$

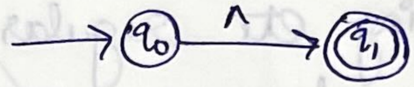
– Means we can repeat either 1 or 10 any number of times. Every 0 must be preceded by 1.

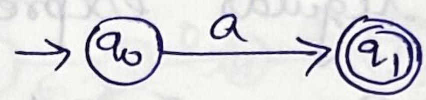
Final RE –

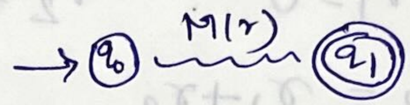
$(1+(0\,))^* . 00 . (1+(10))^*$

Construction of NFA for L(r) :-

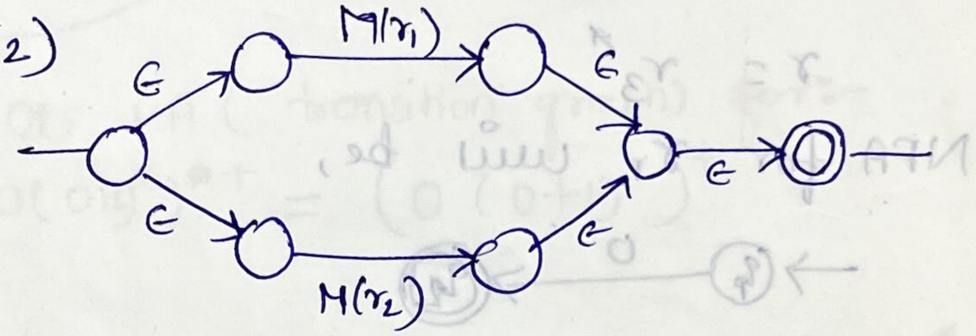
(a) NFA accepts ϕ 

(b) NFA accepts $\wedge$ 

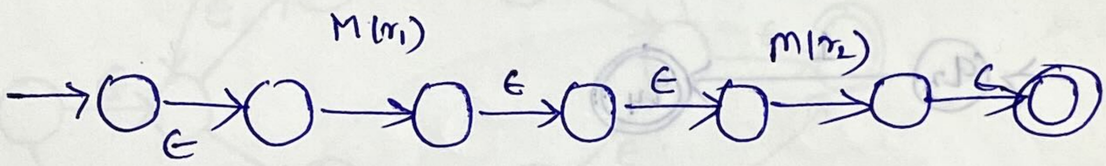
(c) NFA accepts a 

(d) NFA accepting L(r) 

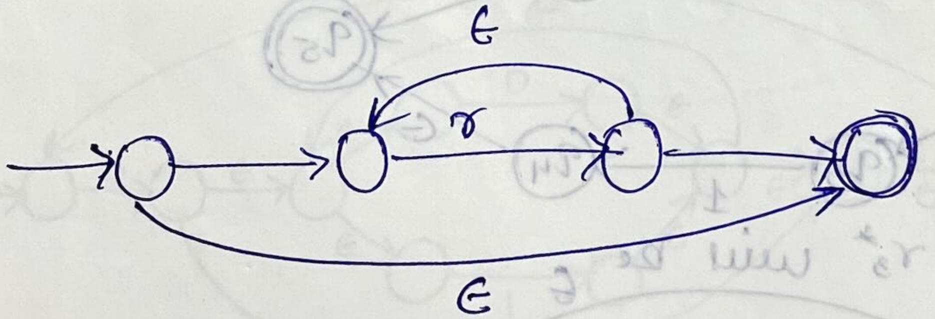
(e) $L(r_1 + r_2)$



(f) $L(r_1 . r_2)$



(g) $L(r^*)$



Q:- Construct NFA with ϵ-moves for the regular expression $(0+1)^*$.

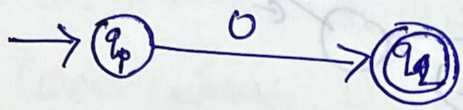
Solⁿ: The NFA will be constructed step by step by breaking the regular expression into smaller regular expressions.

Let $r_1 = 0 \qquad r_2 = 1$

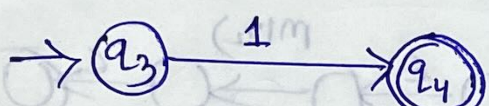
$r_3 = r_1 + r_2$

$r = r_3^*$

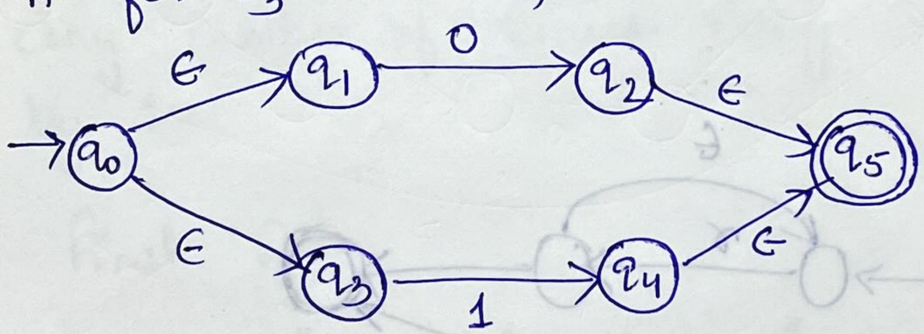
NFA for r_1 will be,



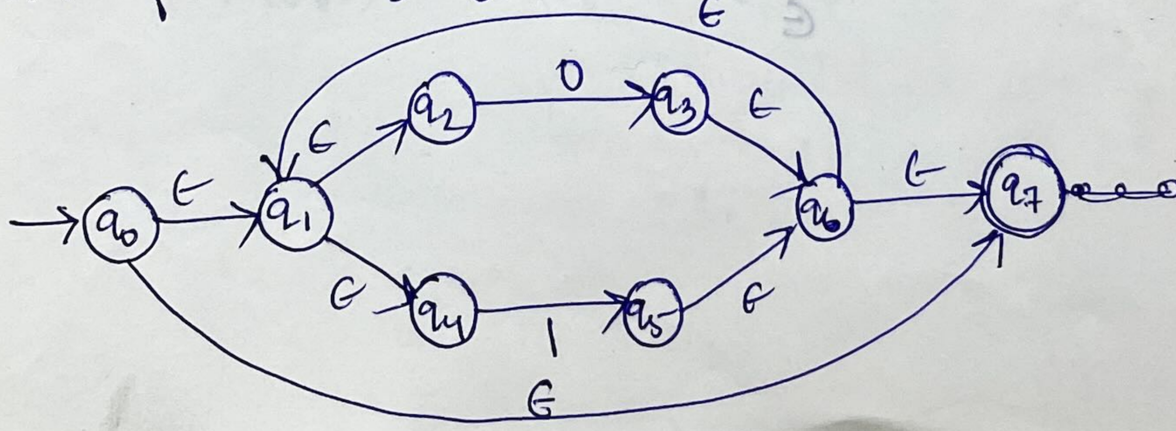
NFA for r_2 will be,



NFA for r_3 will be,

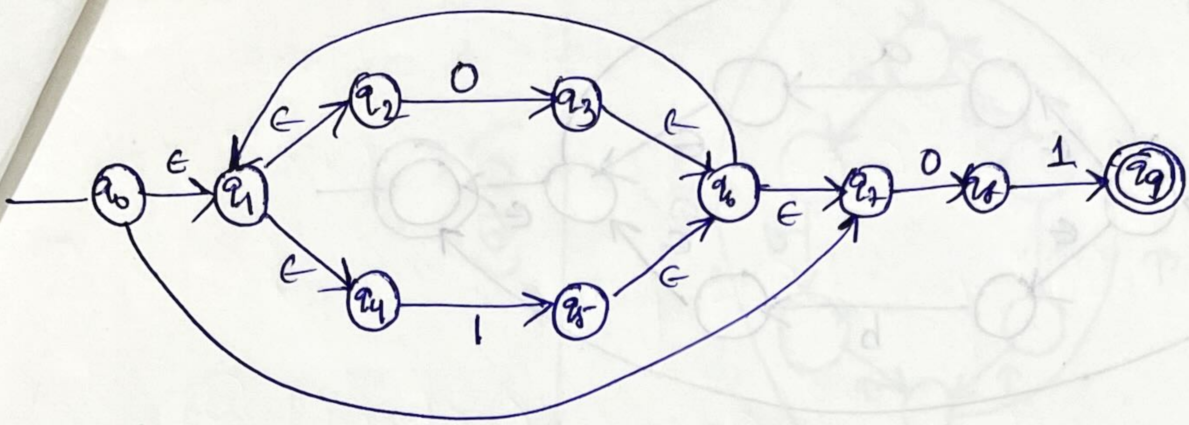


NFA for $r = r_3^*$ will be



Q:- Convert the following regular expression to FA
$(0|1)^* 01$.

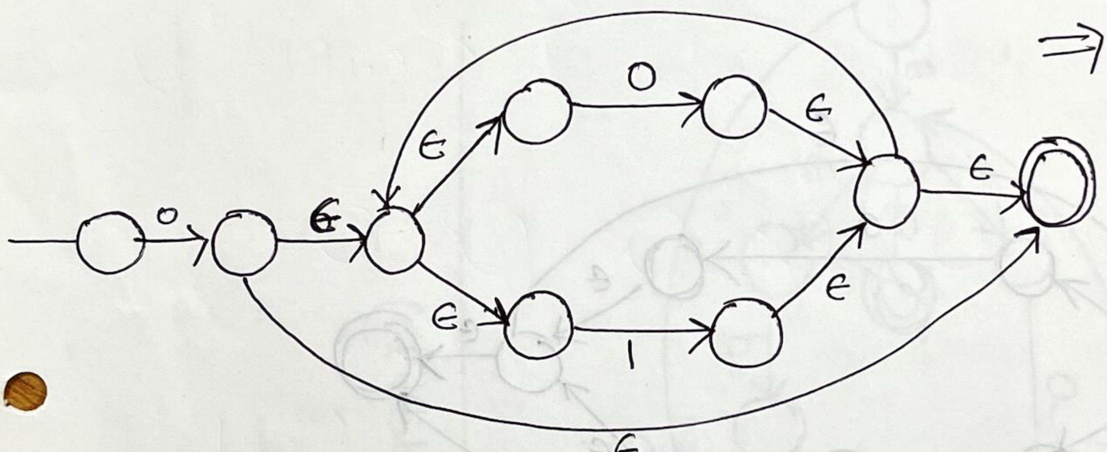
Soln:- $(0+1)^* 01$



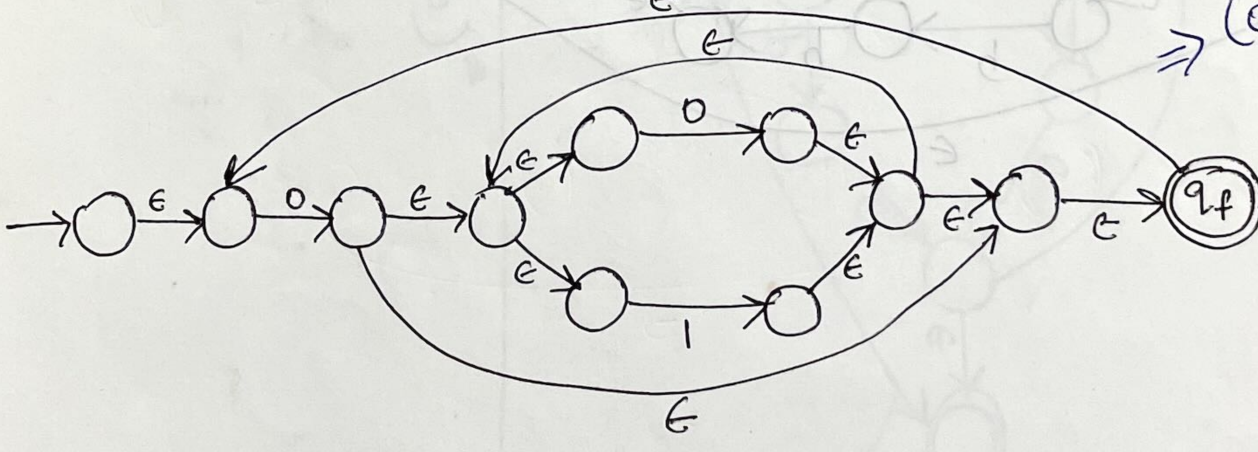
Q:- Construct the FA (transition graph) for:-

i) $R = (0(0|1)^*)^+ = (0(0+1)^*)^+$

$\Rightarrow 0(0+1)^*$



$\Rightarrow (0(0+1)^*)^+$

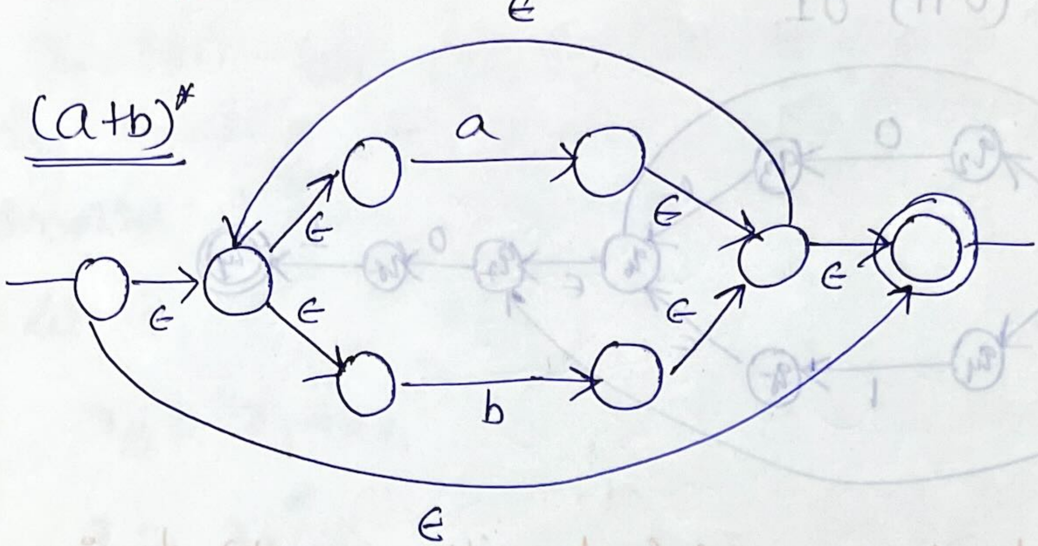


17/2/2022

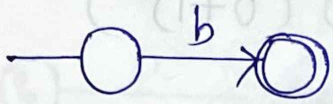
Q:- Construct a NFA for language:

$L = (a+b)^* b (a+bb)^*$

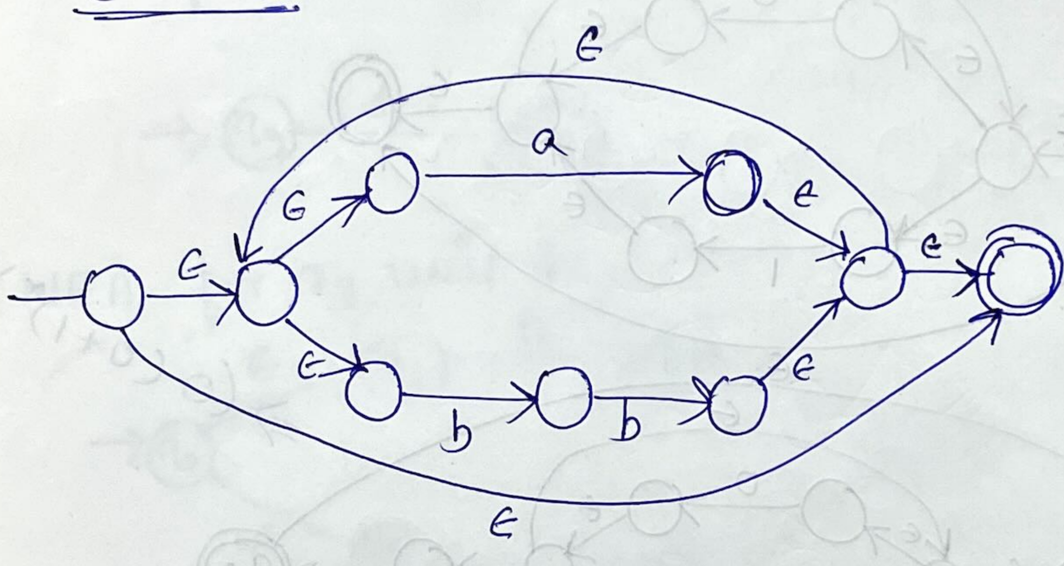
$(a+b)^*$



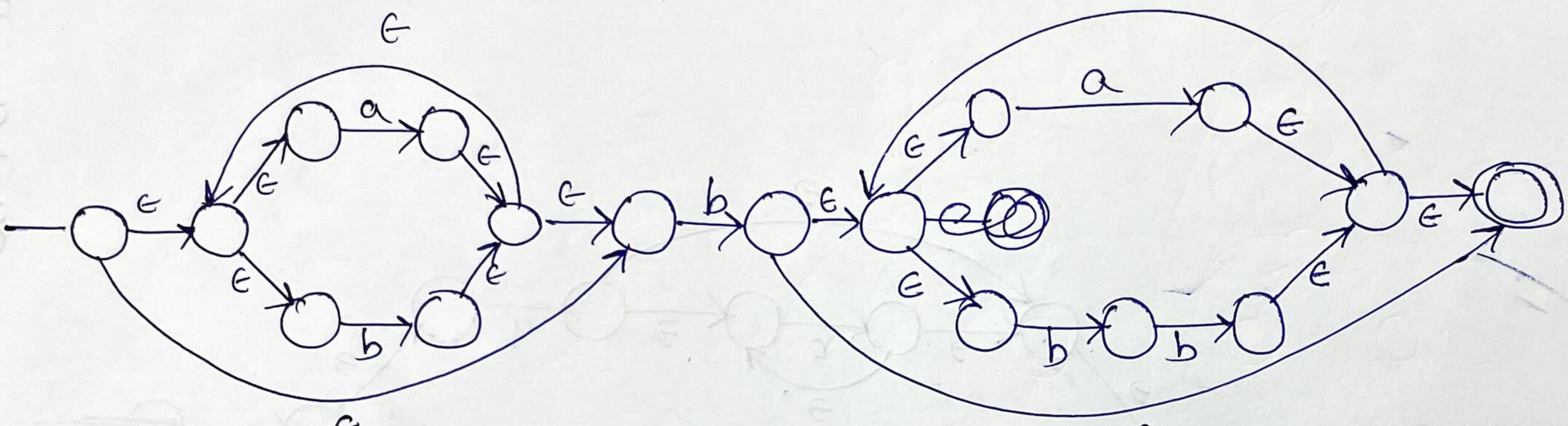
b



$(a+bb)^*$

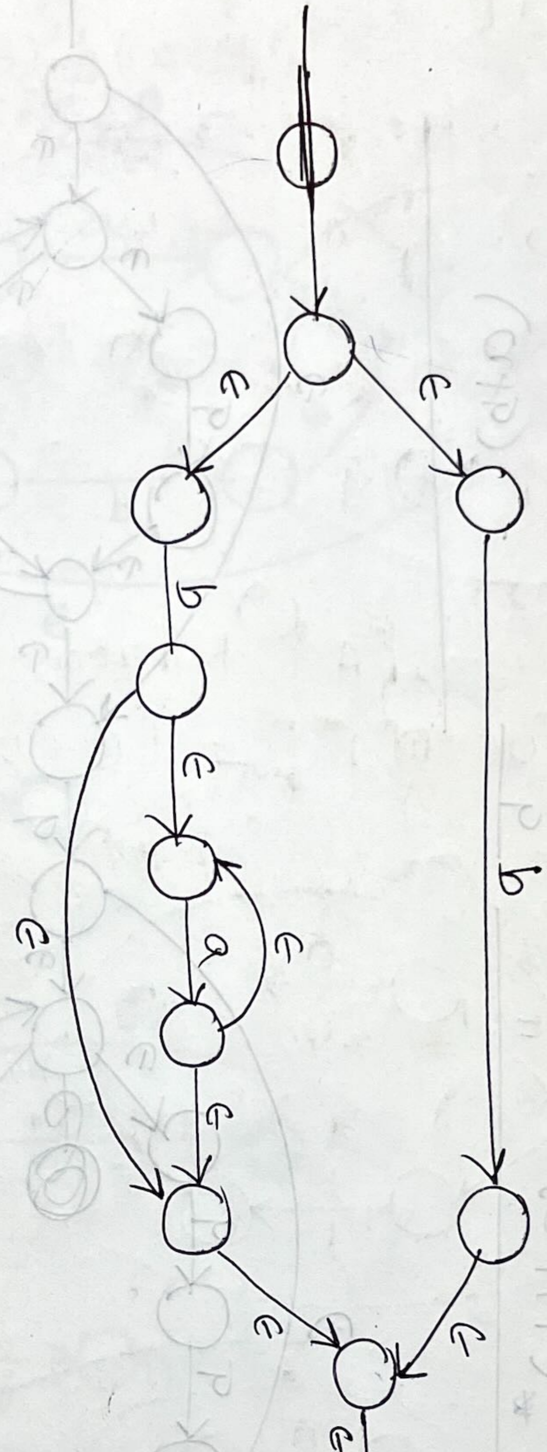


$(a+b)^* \, b \, (a+bb)^*$



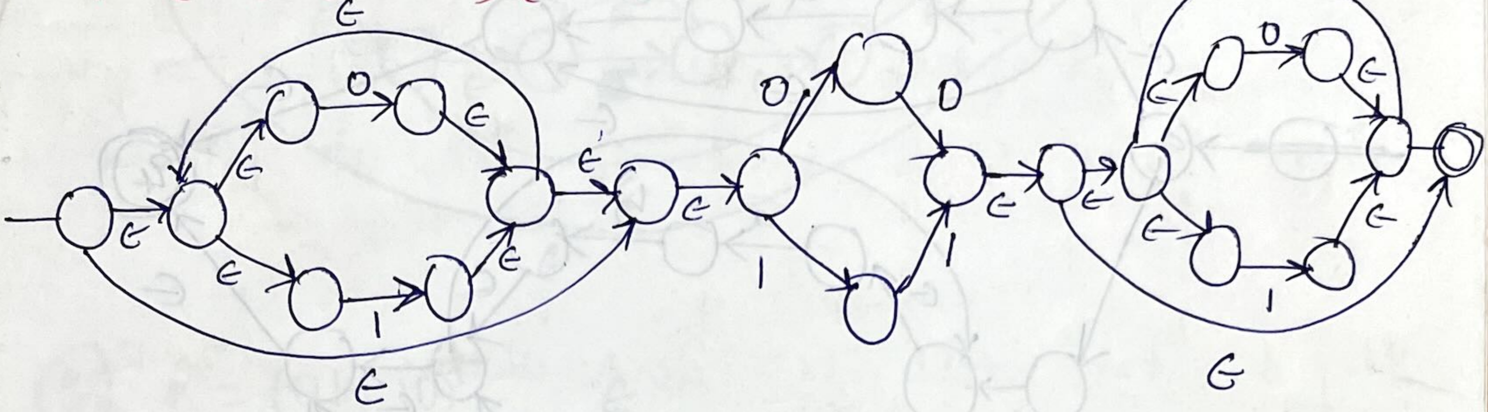
$(a+b)^*$ $\quad b \quad$ $(a+bb)^*$

Q: Construct b+ ba*

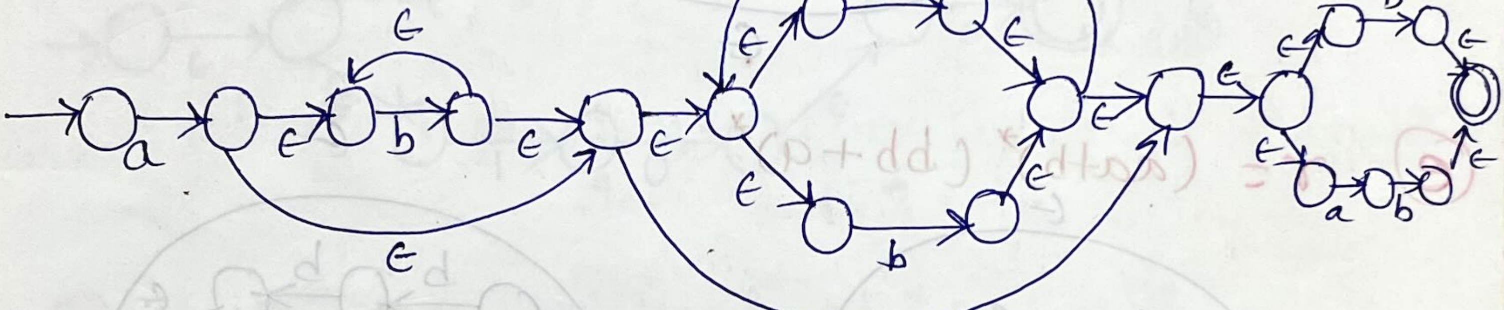


Q. Construct an NFA equivalent to the following regular expression:-

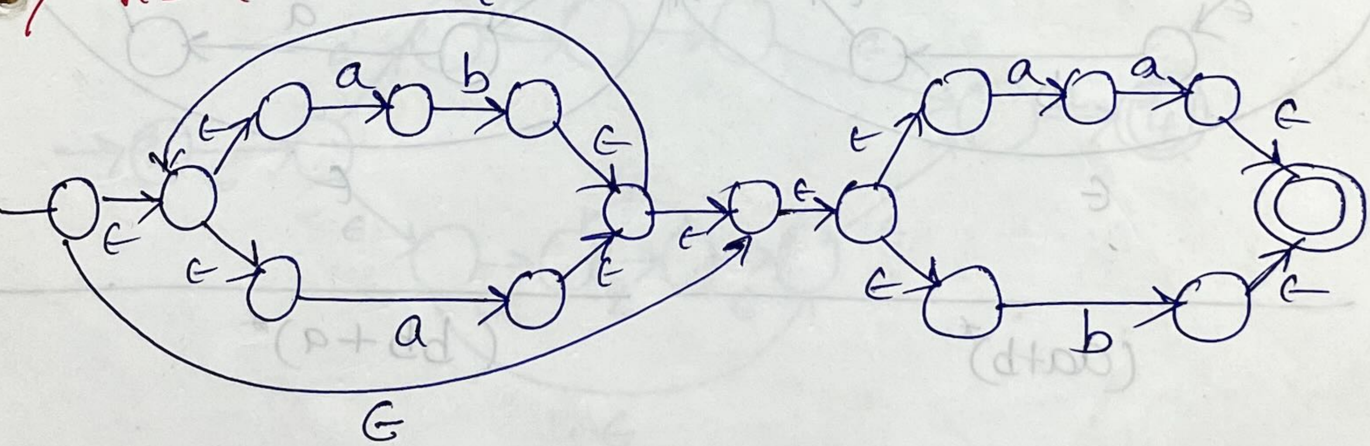
a) $(0+1)^*(00+11)(0+1)^*$



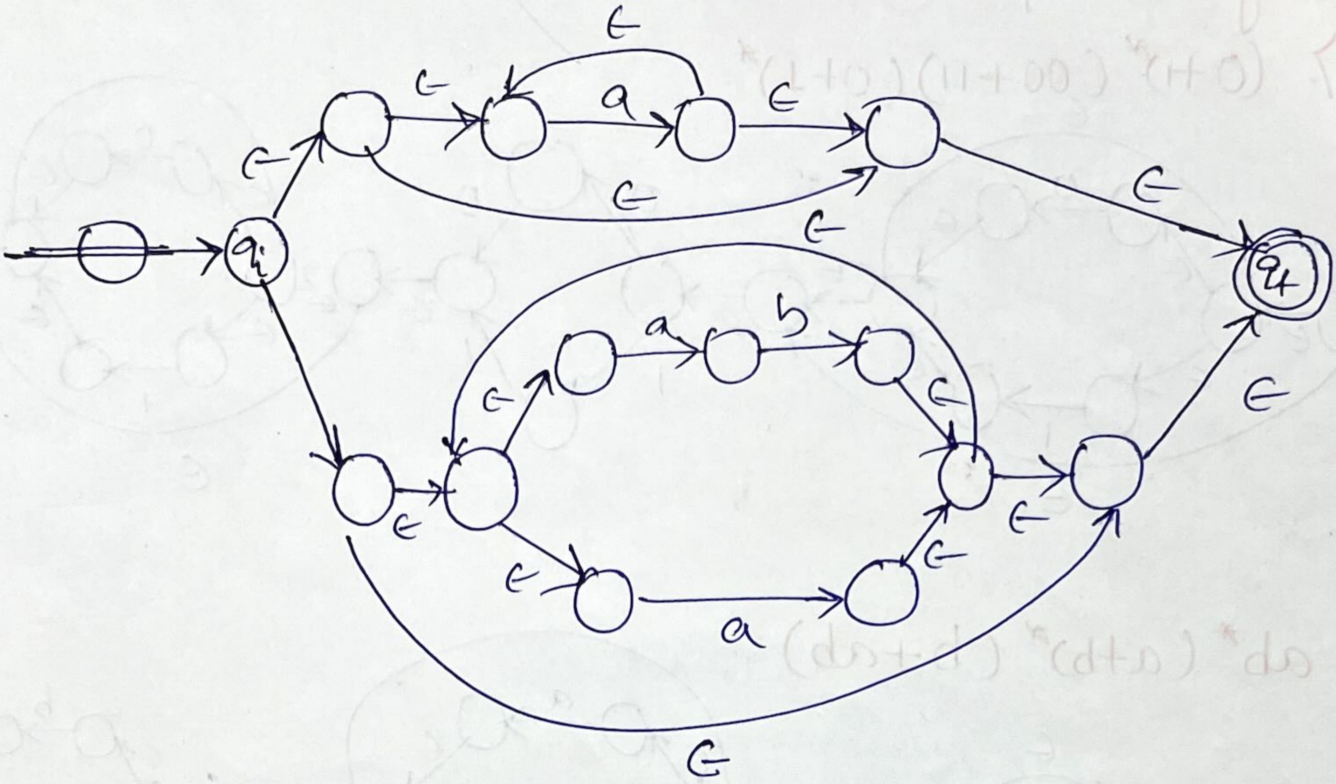
b) $ab^*(a+b)^*(b+ab)$



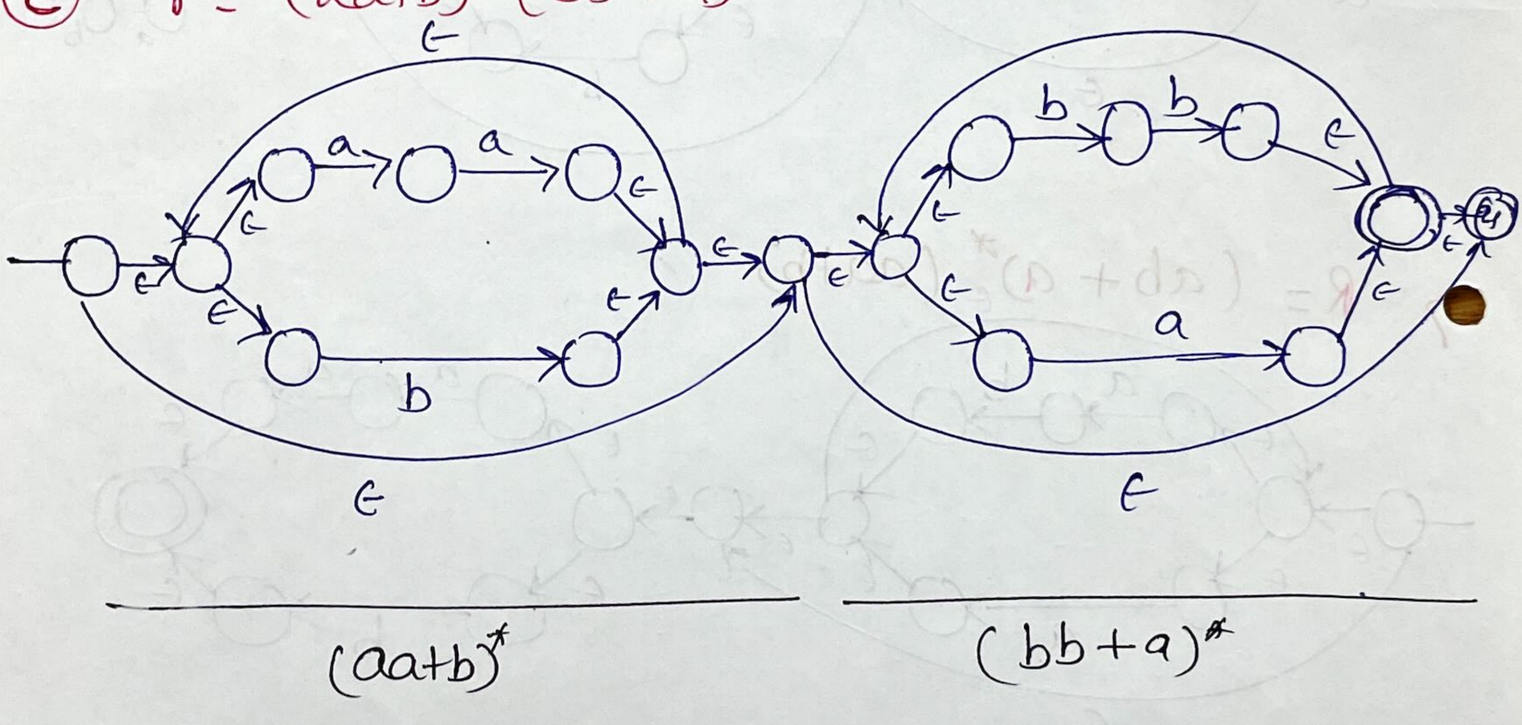
c) $R = (ab+a)^*(aa+b)$



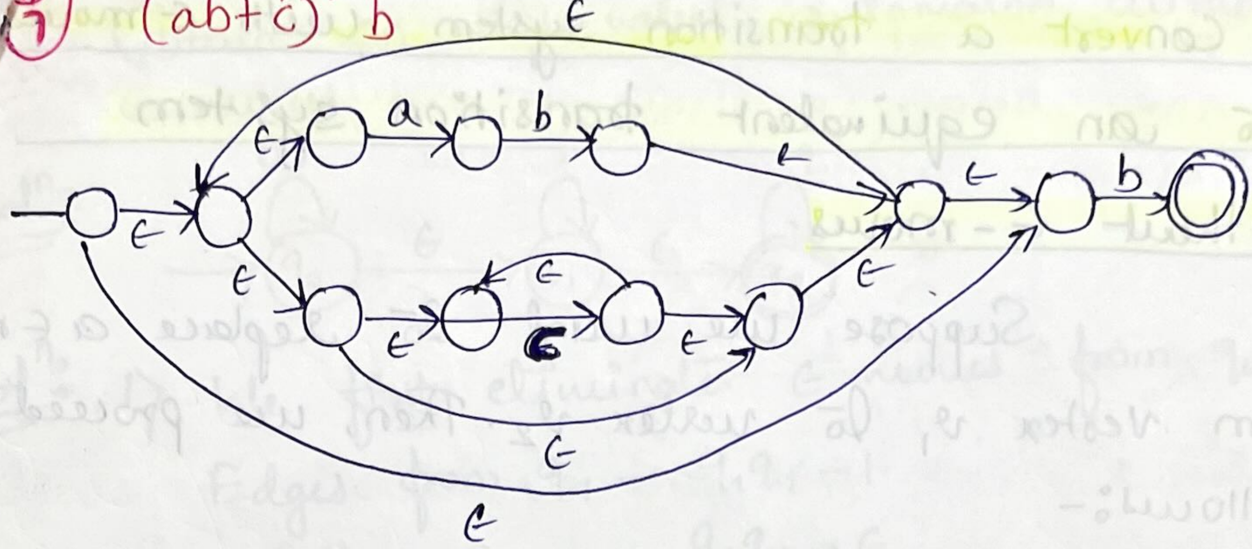
(d) $r = a^* + (ab+a)^*$



(e) $r = (aa+b)^* (bb+a)^*$

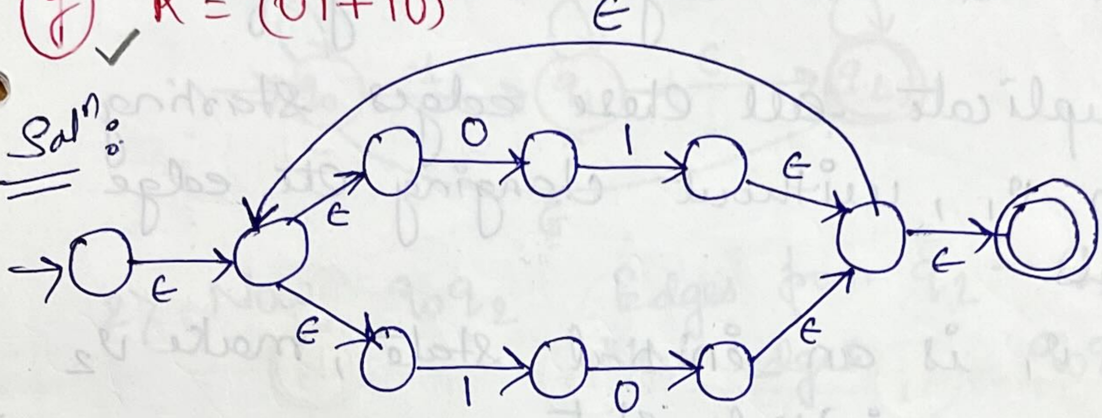


Q $(ab+c^*)^* b$

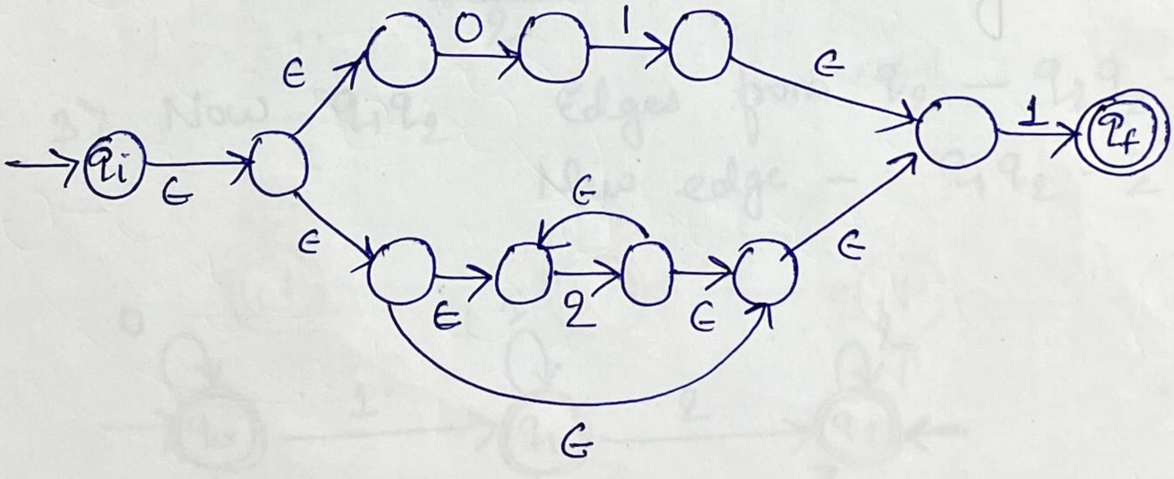


Q $R = (01+10)^*$

Sol^n:



Q $(01+2^*)\,1$



To convert a transition system with ϵ moves into an equivalent transition system without ϵ-moves

Suppose we want to replace a ϵ move from vertex v_1 to vertex v_2. Then, we proceed as follows:-

Step 1: Find all the edges starting from v_2.

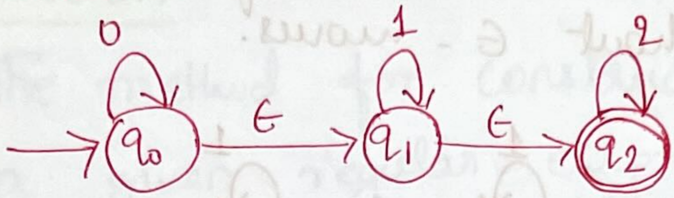
Step 2:- Duplicate all these edges starting from v_1, without changing the edge labels.

Step 3:- If v_1 is an initial state, make v_2 also an initial state.

Step 4: If v_2 is final state, make v_1 also as final state.

Q: Obtain an equivalent automaton without ϵ-moves.

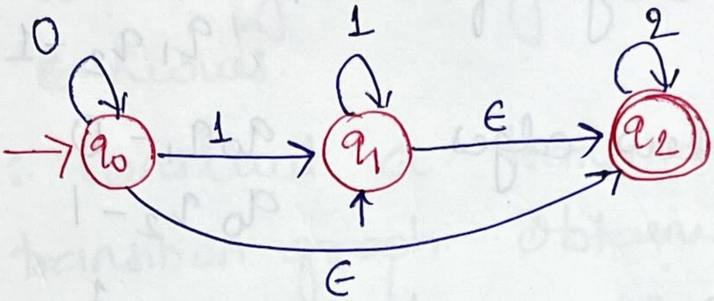
Soln:



Soln: 1) We first eliminate ϵ moves from q_0 to q_1.

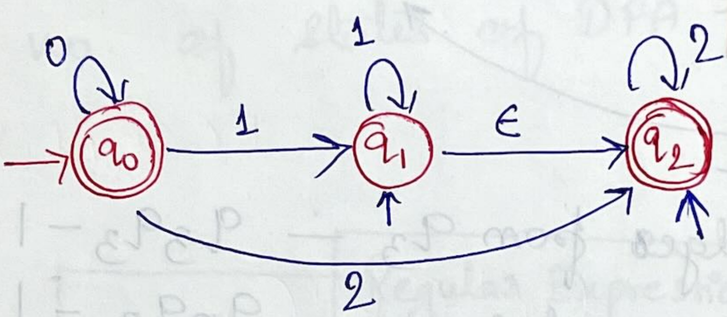
Edges from q_1 – q_1q_1 – 1

q_1q_2 – ϵ.



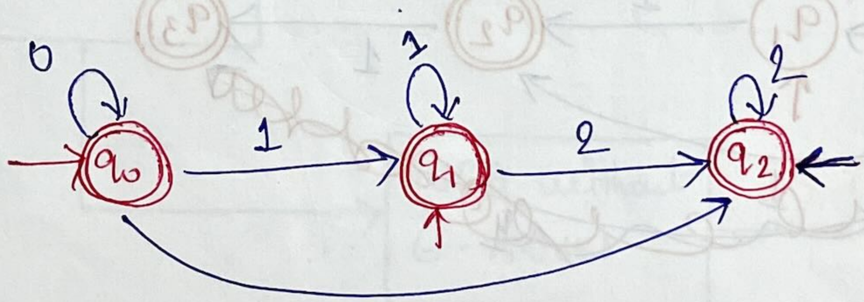
2) Now q_0q_2 Edges from q_2 – q_2q_2 – 2

New edges q_0q_2 – 2



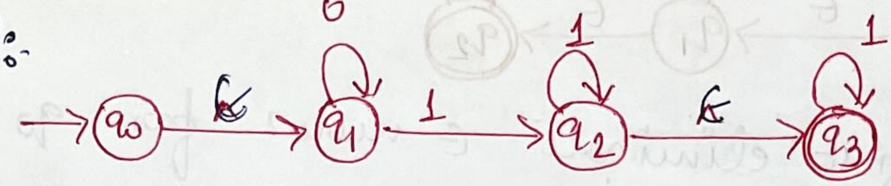
3) Now q_1q_2 Edges from q_2 – q_2q_2 – 2

New edge – q_1q_2 – 2



Q:- A transition diagram with ϵ moves is given below. Construct an equivalent transition diagram without ϵ-moves.

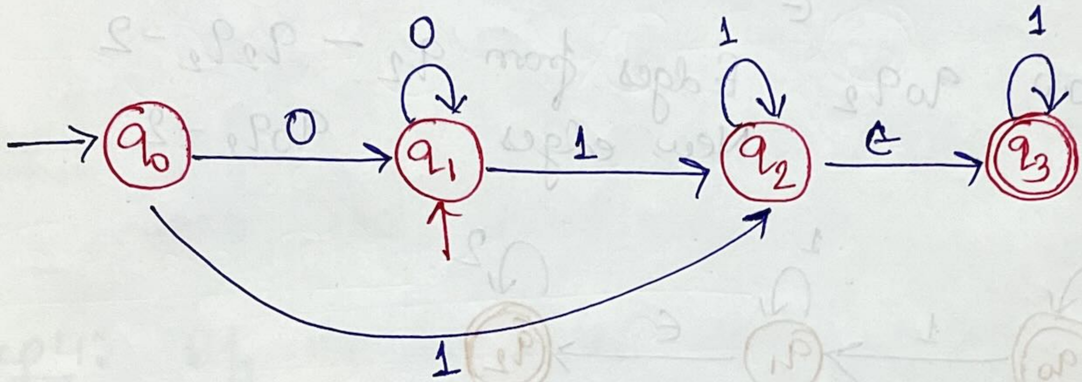
Sol^n:



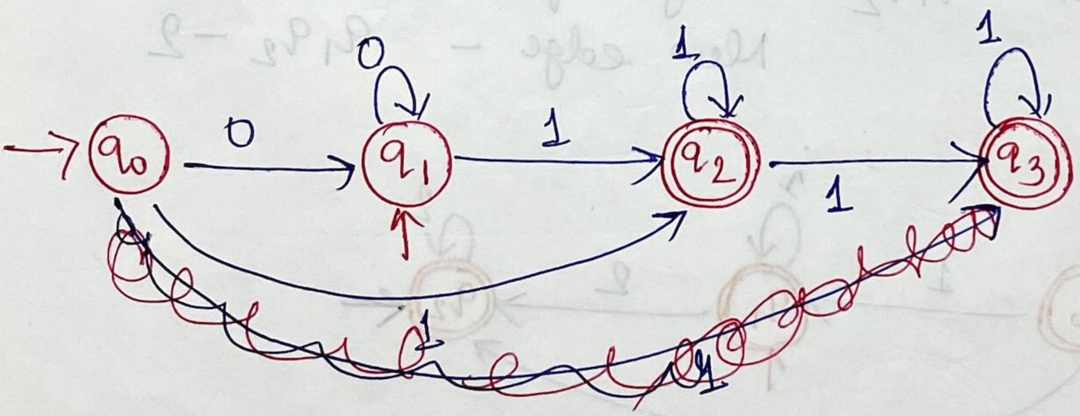
ϵ moves edges are q_0q_1, q_2q_3

1) q_0q_1 — Edges from q_1 – q_1q_1 – 0
q_1q_2 – 1

New edges — q_0q_1 – 0
q_0q_2 – 1



2) q_2q_3 — Edges from q_3 – q_3q_3 – 1
New edge — q_2q_3 – 1

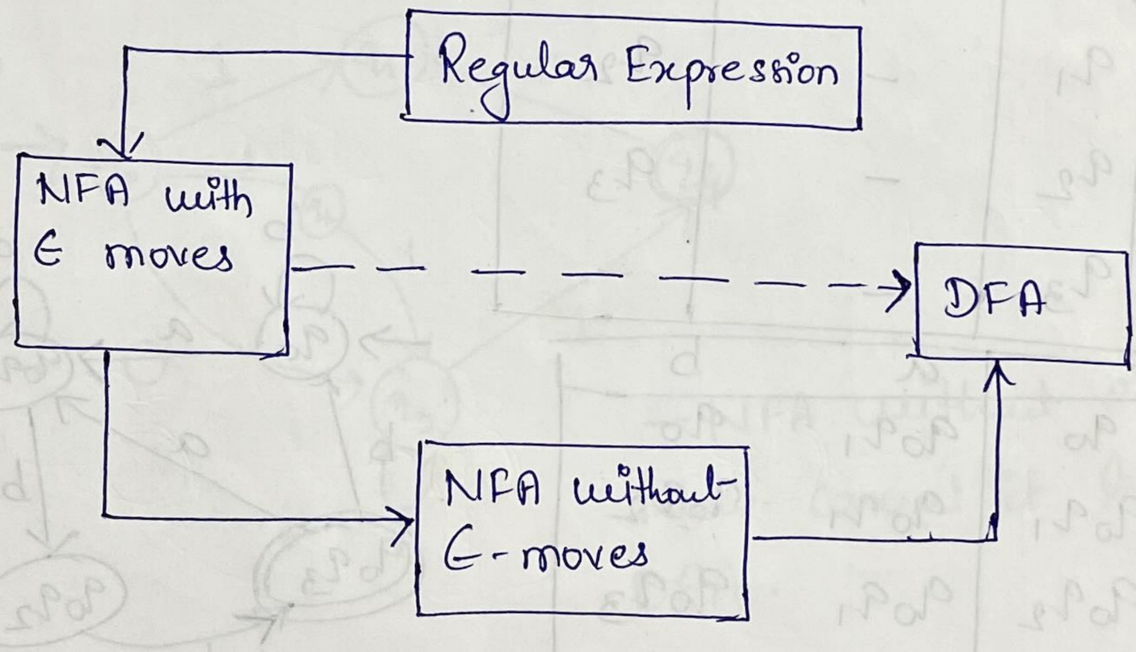


Construction of DFA Equivalent to a Regular Expression

The method for constructing a DFA equivalent to a given regular expression is called Subset method. which involves two steps:-

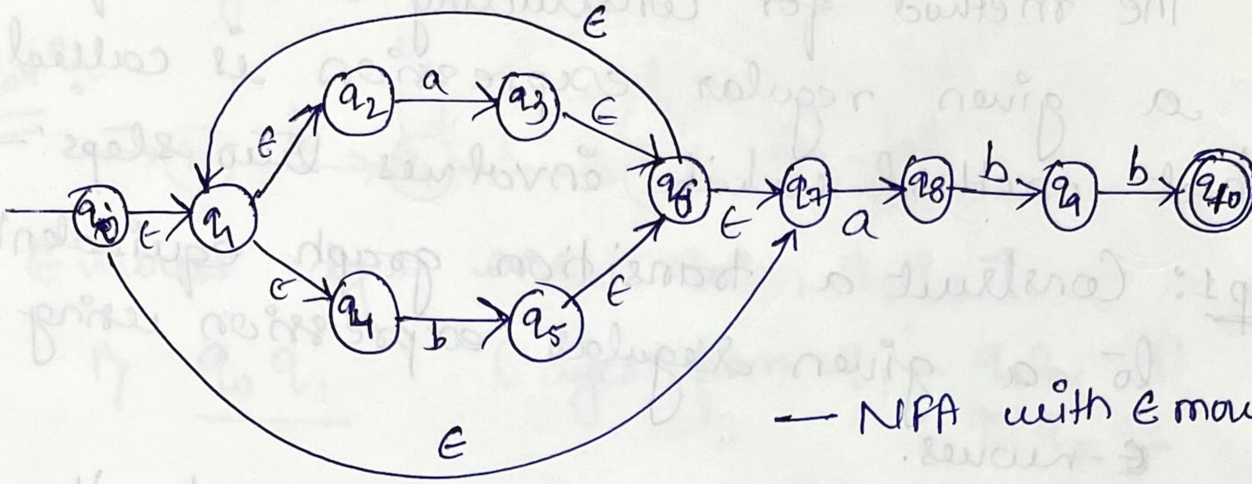
Step 1: Construct a transition graph equivalent to a given regular expression using ϵ-moves.

Step 2: Construct a transition table for the transition graph obtained in step (1). Then, construct the DFA using the method of conversion NFA to DFA and reduce the no. of states of DFA if possible.

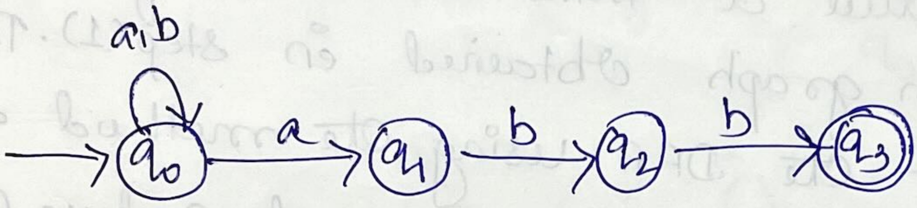


Q: Construct a finite automaton equivalent regular expression

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



— NFA with ε moves

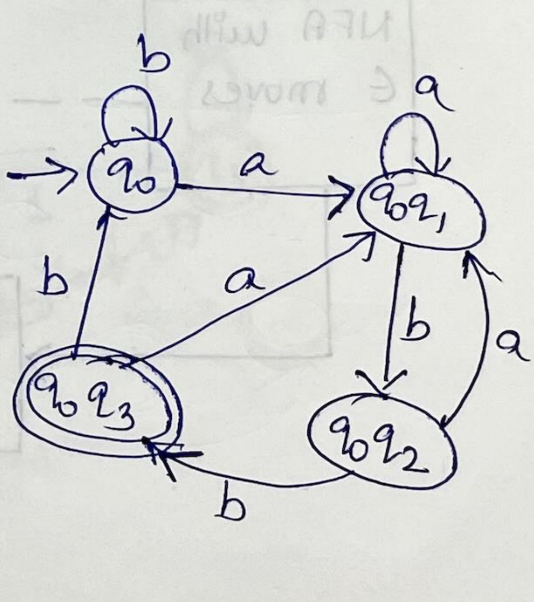


— NFA without ε moves.

Transition Table :-

	a	b
→ q_0	q_0, q_1	q_0
q_1	—	q_2
q_2	—	q_3
* q_3	—	—

	a	b
→ q_0	q_0q_1	q_0
q_0q_1	q_0q_1	q_0q_2
q_0q_2	q_0q_1	q_0q_3
* q_0q_3	q_0q_1	q_0



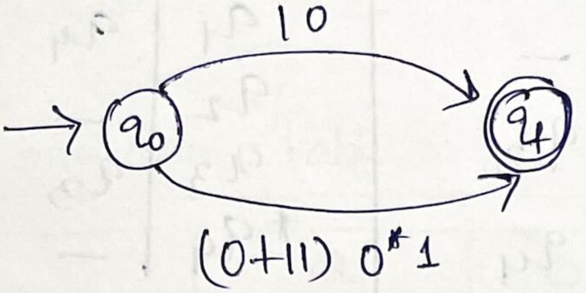
Equivalent DFA

Design a DFA from given Regular Expression

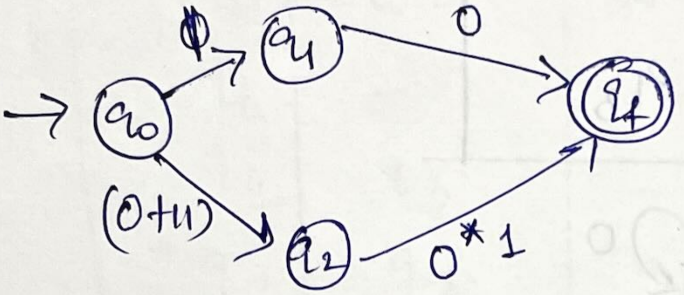
$10 + (0+11)\,0^*1.$

Solⁿ:

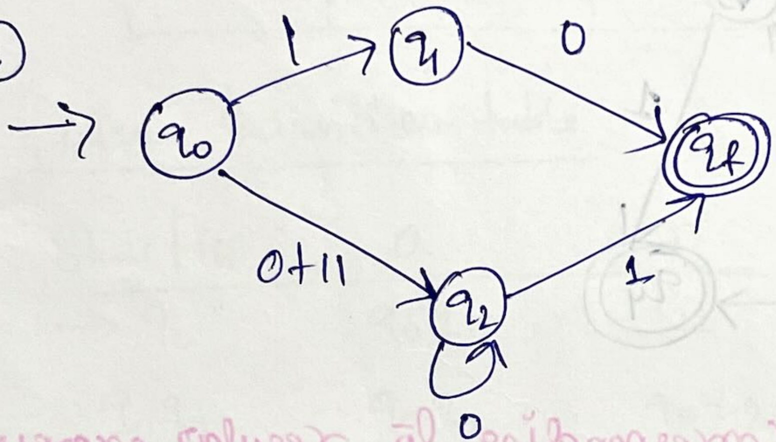
(a)



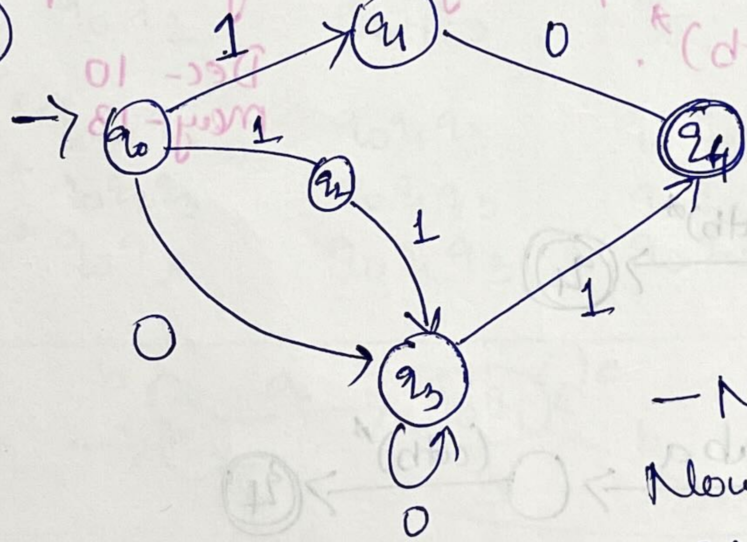
(b)



(c)



(d)

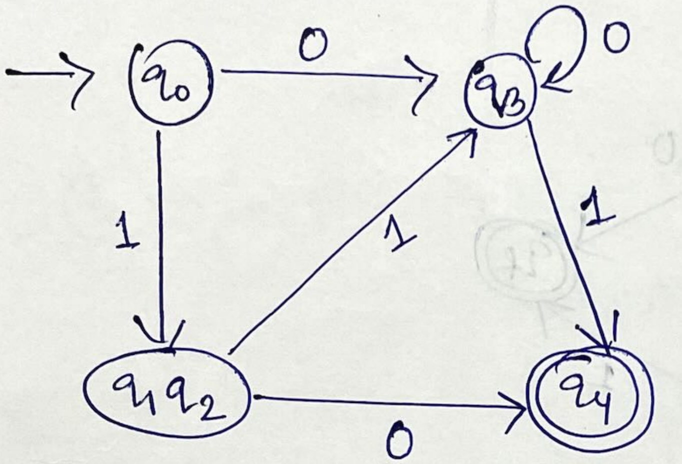


— NFA without ε moves.

Now convert it to required DFA.

State i/p	0	1
$\rightarrow q_0$ ✓	q_3	$q_1 q_2$
q_1	q_4	—
q_2	—	q_3
q_3 ✓	q_3	q_4
$*q_4$ ✓	—	—
$q_1 q_2$ ✓	q_4	q_3

	0	
$\rightarrow q_0$	q_3	$q_1 q$
q_1	q_4	—
q_2	—	q_3
q_3	q_3	q_4
$*q_4$	—	—

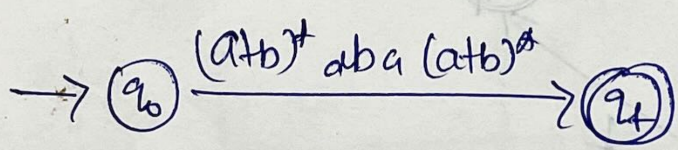


Q: Design a DFA corresponding to regular expression $(a+b)^* aba (a+b)^*$.

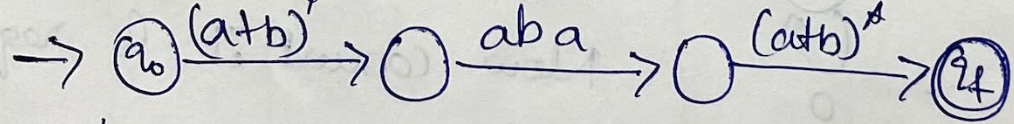
Dec-10
May-13

Sol^n:

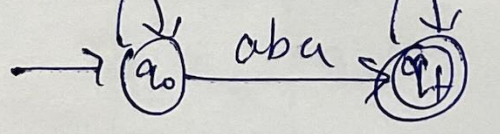
(a)

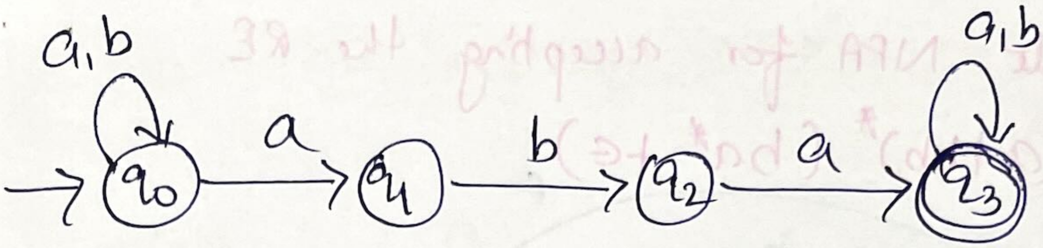


(b)



(c)





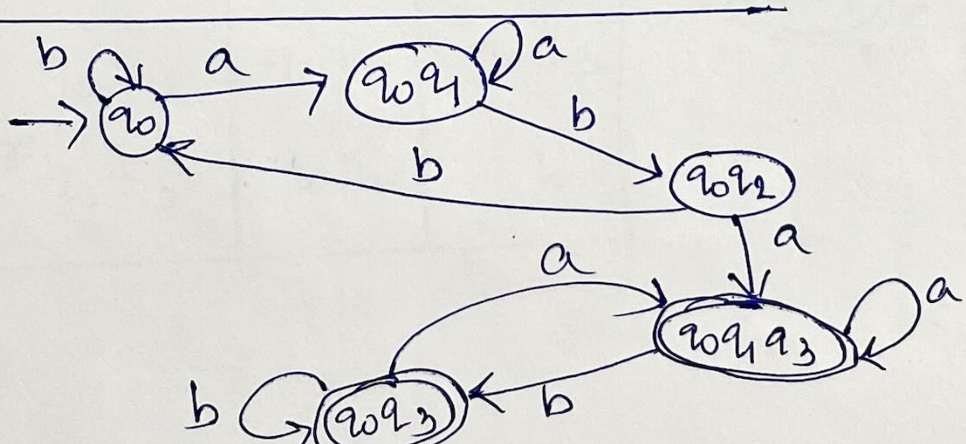
Transition Table for above FA:-

	a	b
$\rightarrow q_0$	$q_0 q_1$	q_0
q_1	–	q_2
q_2	q_3	–
$*q_3$	q_3	q_3

New transition table

State / i/p	a	b
$\rightarrow q_0$	$q_0 q_1$	q_0
$q_0 q_1$	$q_0 q_1$	$q_0 q_2$
$q_0 q_2$	$q_0 q_1 q_3$	q_0
$*q_0 q_1 q_3$	$q_0 q_1 q_3$	$q_0 q_2 q_3$ / $q_0 q_3$
$*q_0 q_2 q_3$	$q_0 q_1 q_3$	$q_0 q_3$
$*q_0 q_3$	$q_0 q_1 q_3$	$q_0 q_3$

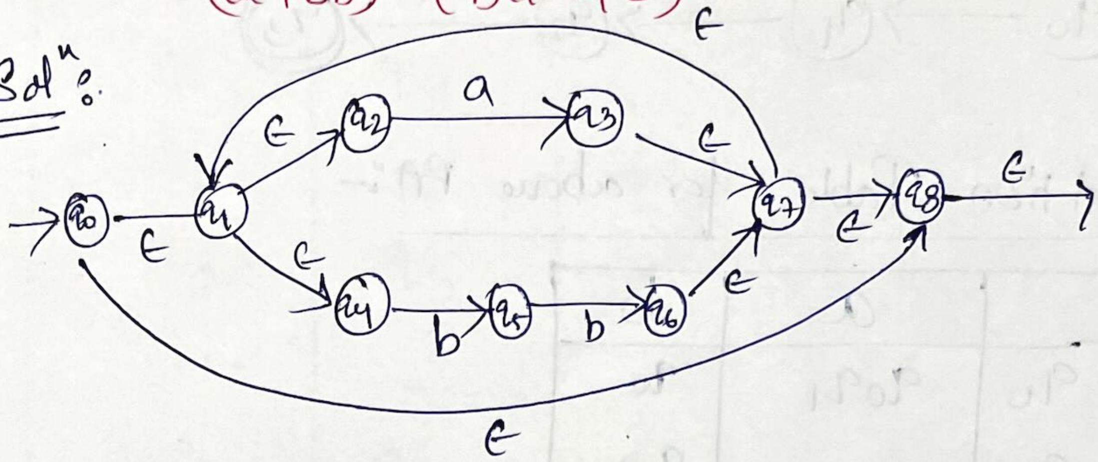
} Replace $q_0 q_2 q_3$ by $q_0 q_3$

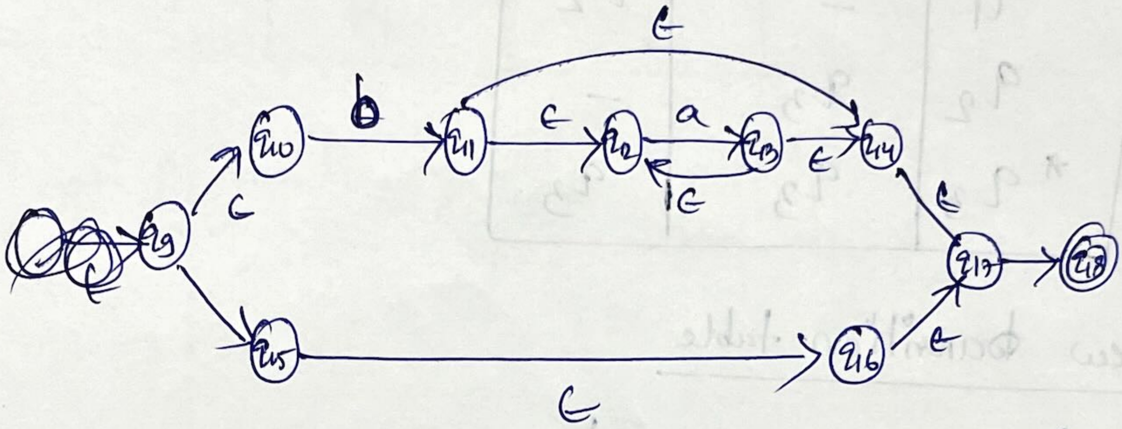


Q: Write NFA-ϵ for accepting the RE

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

Sol^n:





$\rightarrow q_0$ - Initial state

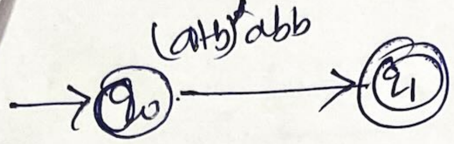
q_{18} - final State

~~Design NFA for re (a+b)* a(b)~~

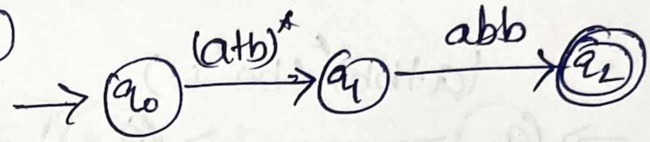
Repeat

$(a+b)^* abb$

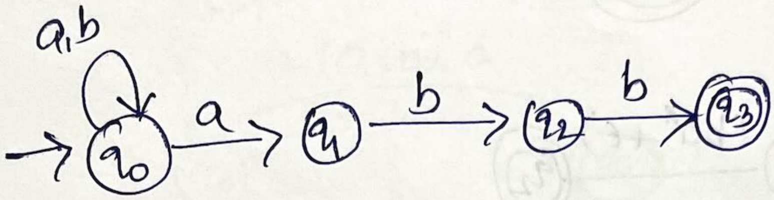
(a)



(b)



(c)

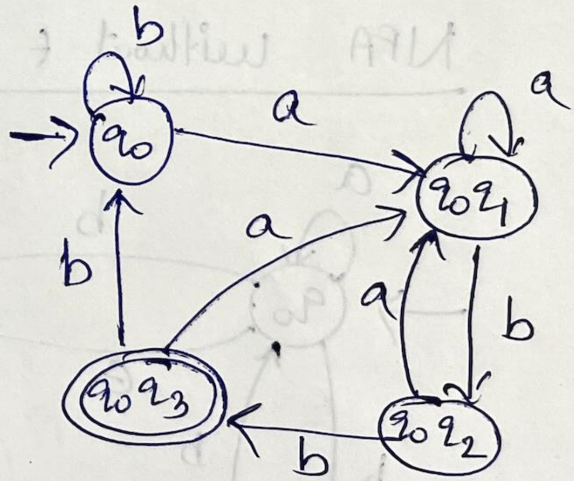


Transition table for NFA

State / i/p	a	b
$\rightarrow q_0$	$q_0 q_1$	q_0
q_1	—	q_2
q_2	—	q_3
$* q_3$	—	—

Transition table for DFA

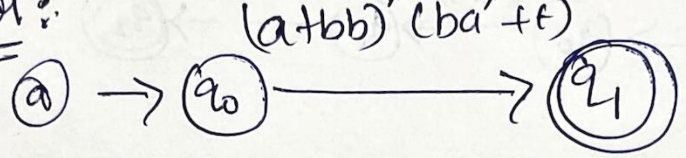
State / i/p	a	b
$\rightarrow q_0$	$q_0 q_1$	q_0
$q_0 q_1$	$q_0 q_1$	$q_0 q_2$
$q_0 q_2$	$q_0 q_1$	$q_0 q_3$
$* q_0 q_3$	$q_0 q_1$	q_0



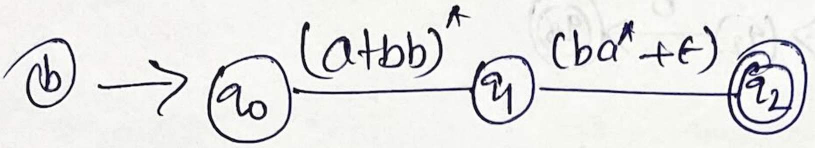
Q: Write NFA for accepting the following $(a+bb)^*(ba^*+\epsilon)$ [May-13]

Sol^n:

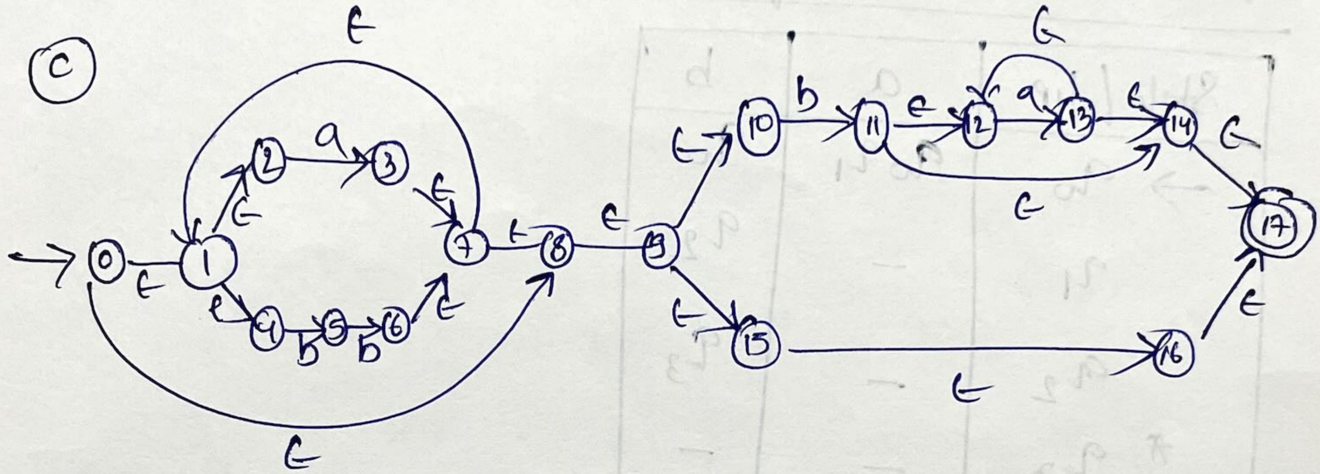
(a)



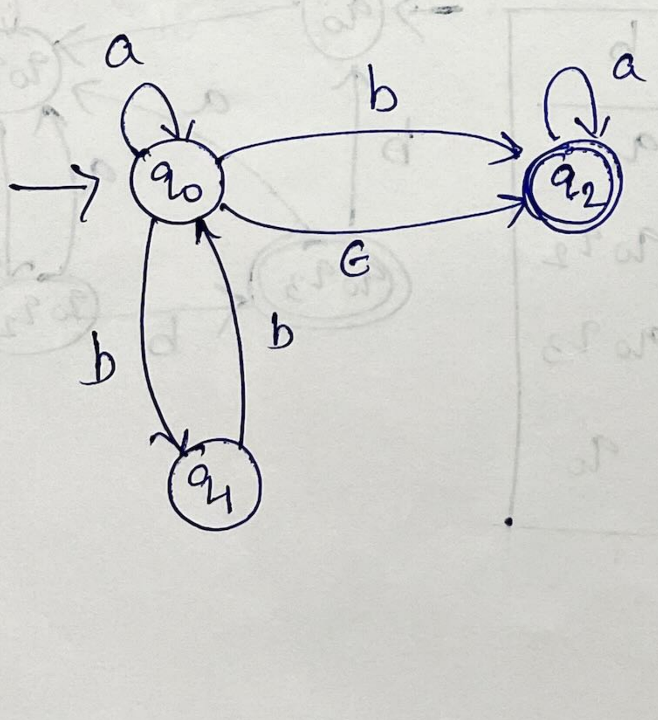
(b)



(c)



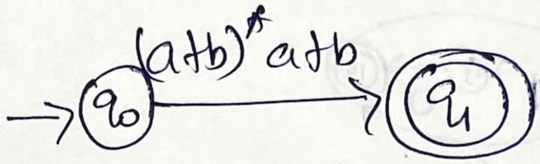
NFA without ε moves

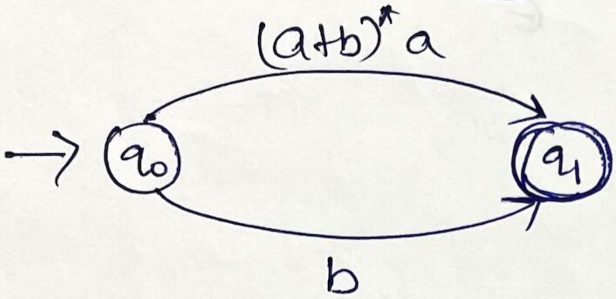


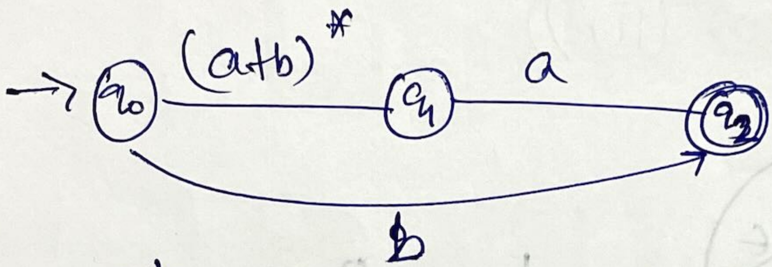
13 Construct an NFA for the following regular expression

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

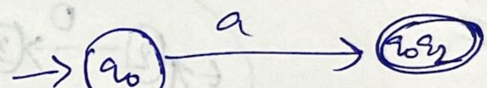
Solⁿ:

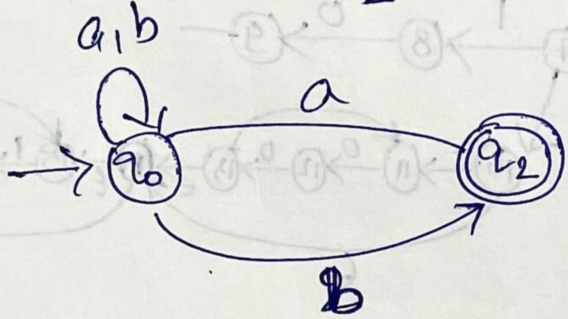






	a	b
→ q_0	q_0q_2	q_0
* q_2	-	-
* q_0q_2	q_0q_2	q_0

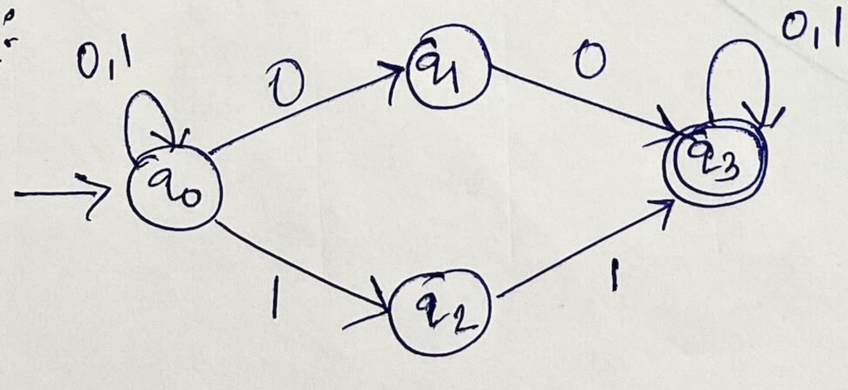




Q:- Construct the finite automaton to accept the regular expression

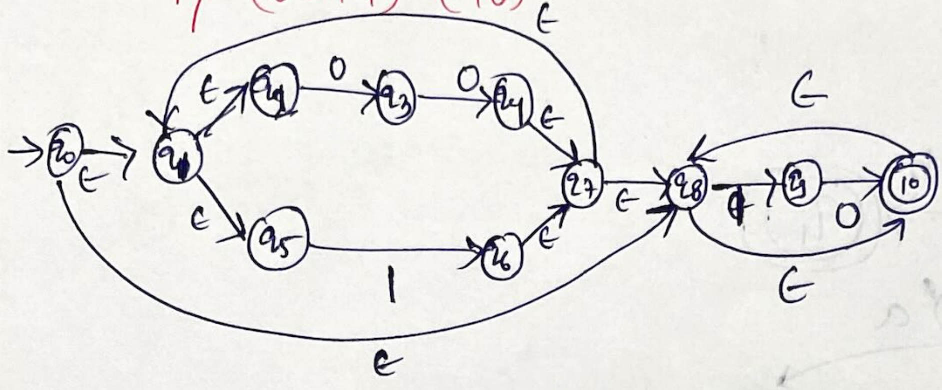
$(0+1)^* (00+11) (0+1)^*$

Solⁿ:



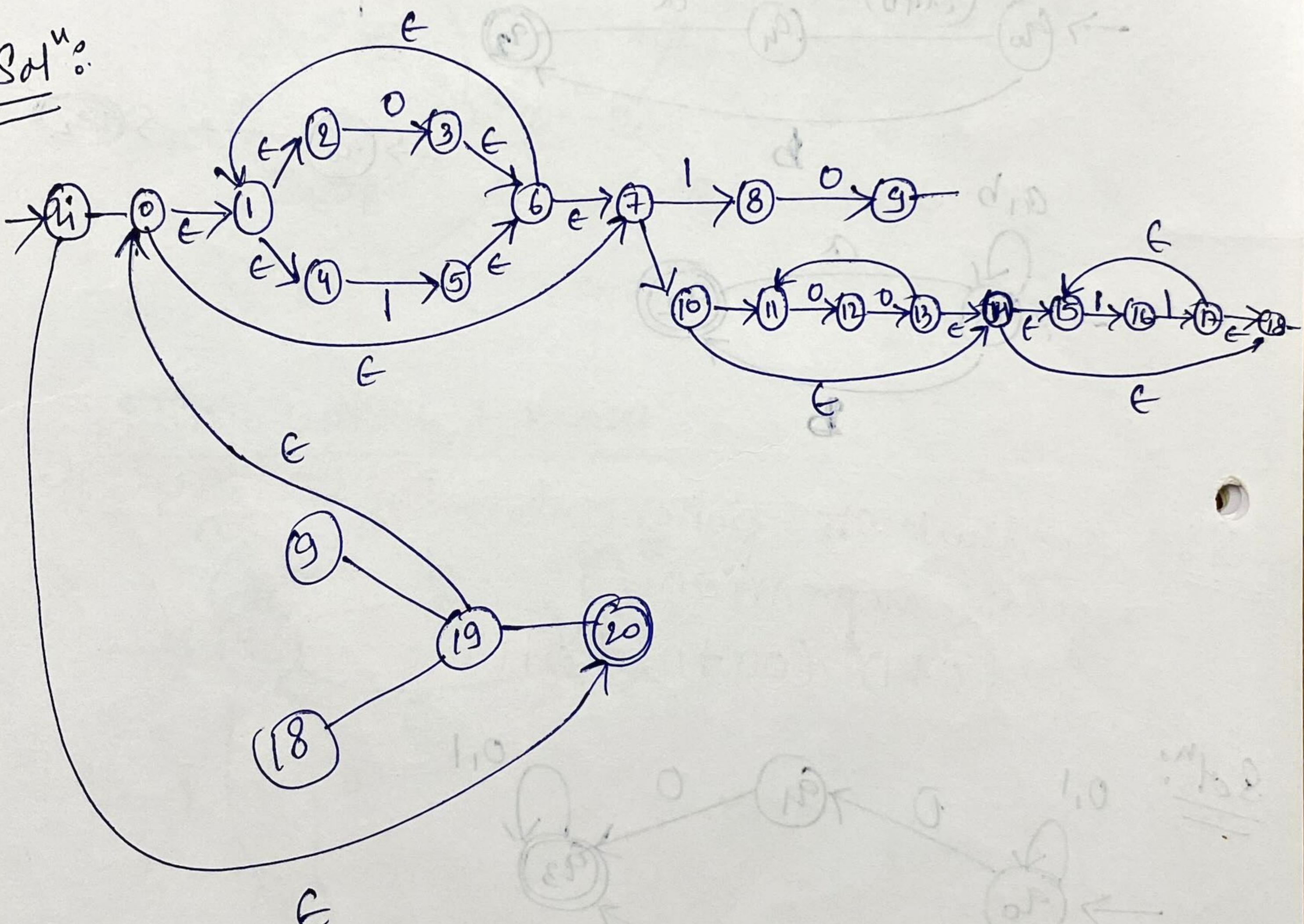
Q: Construct NFA with ϵ moves for [Dec-12]

i) $(00+1)^*(10)^*$



ii) $((0+1)^*(10 + (00)^*(11)^*))^*$

Sol^n:



SC-12) Q: Construct DFA that recognizes the set:-

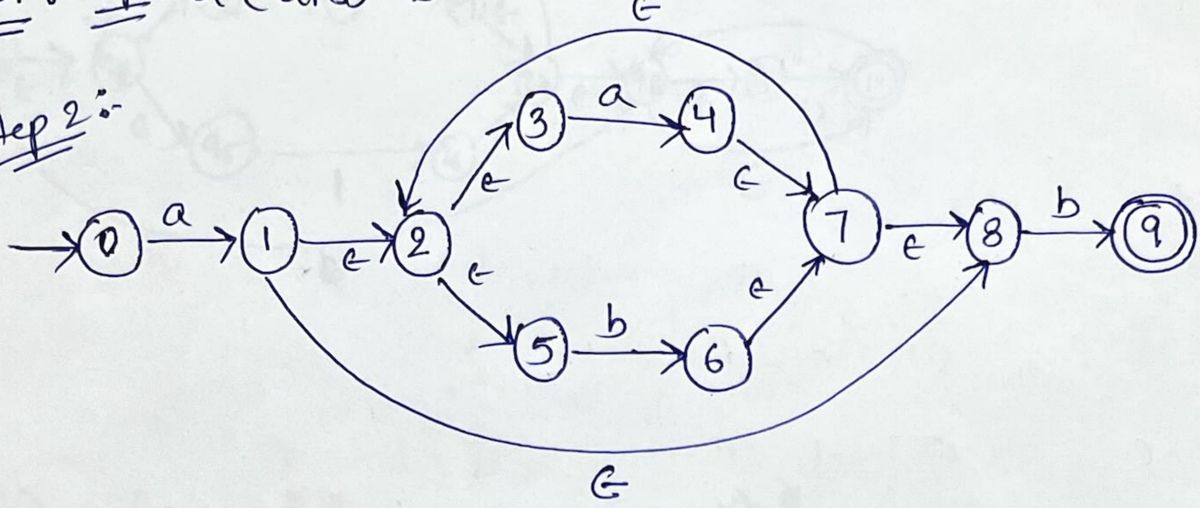
$$R = (1(00)^{*}1 + 01^{*}0)^{*}$$

SUBSET CONSTRUCTION METHOD

Q: Construct minimized DFA to recognized strings starting with a and ending with b over $\Sigma = \{a, b\}$

Soln: **Step 1:-** $a(a+b)^* b$

Step 2:-

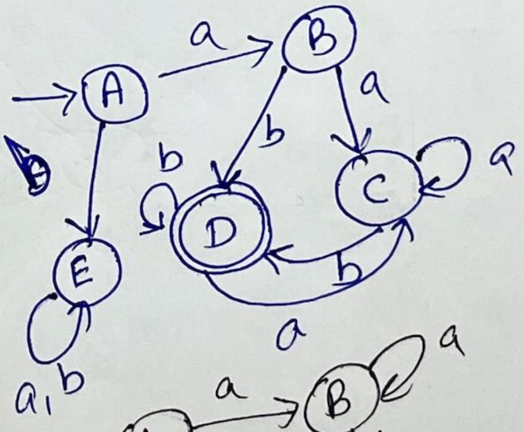


NFA for $a(a+b)^* b$

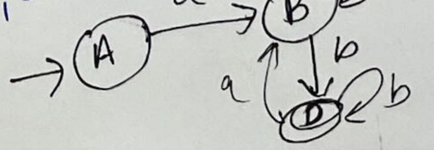
Step 3:- NFA to DFA

r	$y = \epsilon$ closure (r)	$\delta(y, a)$	$\delta(y, b)$
{0} A	{0} A	{1} B	{ }
{1} B	{1, 2, 3, 5, 8} B	{4} C	{6, 9} D
{4} C	{4, 7, 2, 3, 5, 8} C	{4} C	{6, 9} D
* {6, 9} D	{6, 7, 8, 2, 3, 5, 9} D	{4} C	{6, 9} D
{∅} E	{∅} E	{ }	{ }

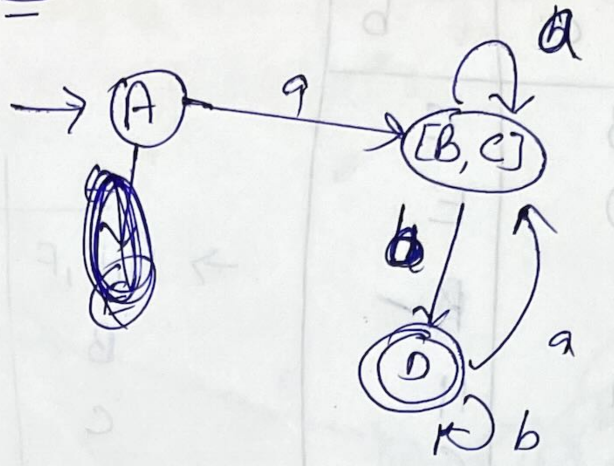
	a	b
→ A	B	—
B	C	D
C	C	D
* D	C	D
E	—	—



→ A	B	—
B	B	D
* D	B	D



Q\Σ	a	b
→A	B	–

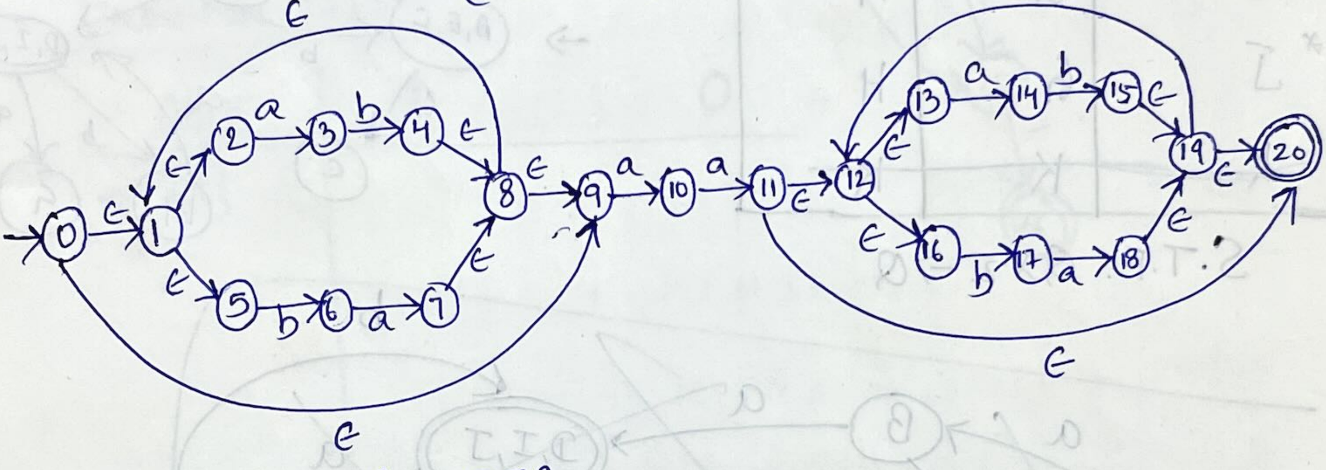


Q: Construct minimized DFA for

$r = (ab/ba)^* \, aa \, (ab/ba)^*$

Solⁿ: Step 1: NFA with ε-moves for r.

$r = (ab + ba)^* \, aa \, (ab + ba)^*$



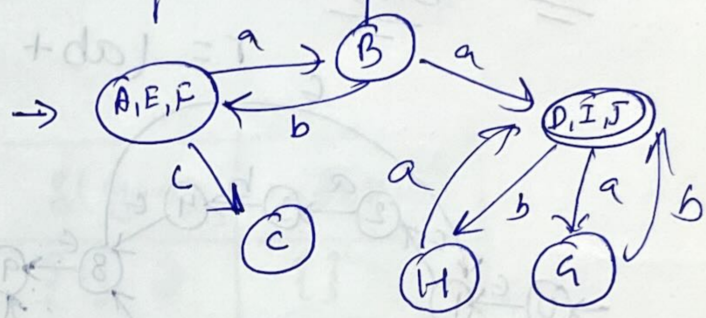
Step 2:- NFA to DFA

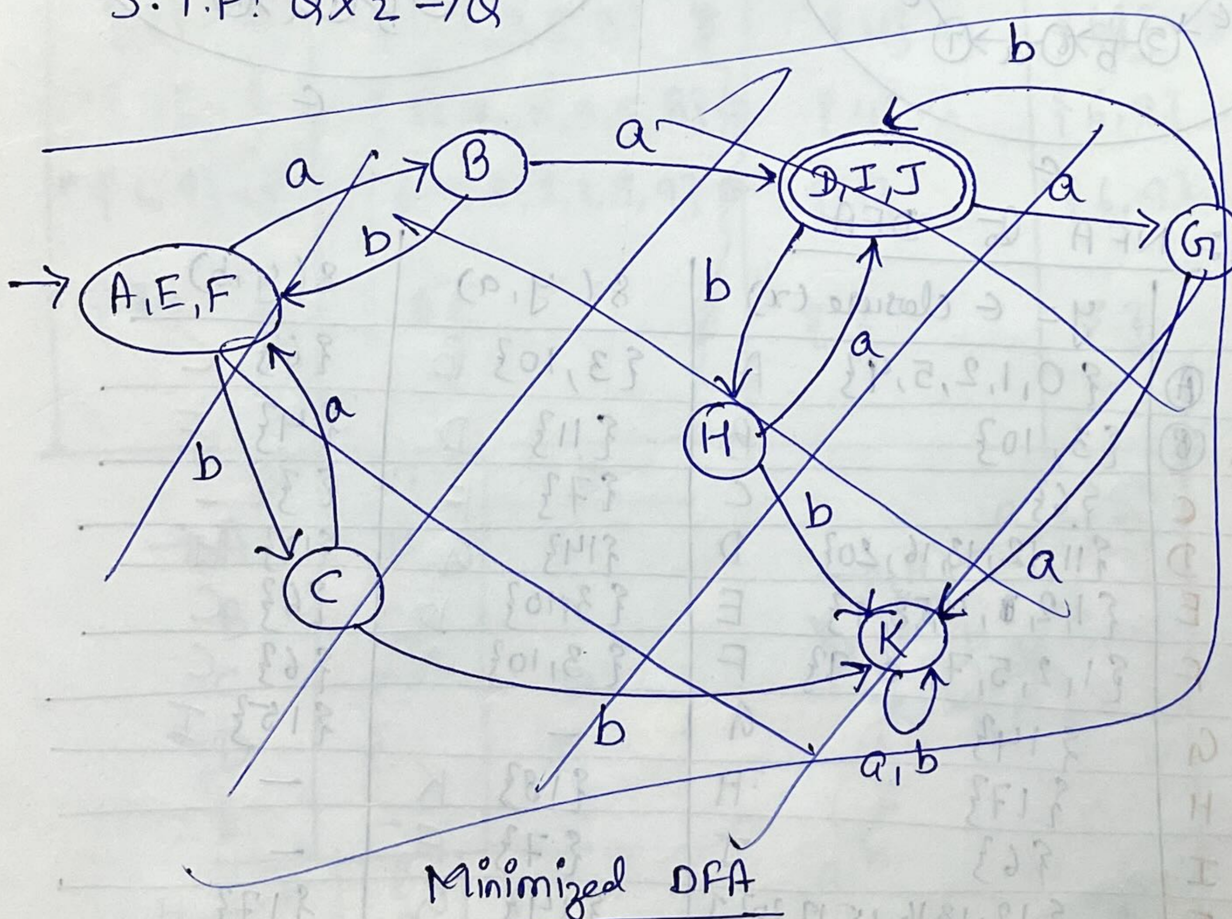
	x		y = ε closure (x)		δ(y, a)	δ(y, b)
A	{0}	A	{0, 1, 2, 5, 9}	A	{3, 10} B	{6} C
B	{3, 10}	B	{3, 10}	B	{11} D	{4} E
C	{6}	C	{6}	C	{7} F	{ } –
D	{11}	D	{11, 12, 13, 16, 20}	D	{14} G	{17} H
E	{4}	E	{1, 2, 3, 4, 5, 8, 9}	E	{3, 10} B	{6} C
F	{7}	F	{1, 2, 5, 7, 8, 9}	F	{3, 10} B	{6} C
G	{14}	G	{14}	G	–	{15} I
H	{17}	H	{17}	H	{18} K	–
	{6}	I	{6}	I	{7} F	–
I	{15}	J	* {12, 13, 16, 15, 19, 20}	J	{14} G	{17} H
J	{18}	K	* {12, 13, 16, 18, 19, 20}	K	{14} G	{17} H

Q/Σ	a	b
→A	B	C
B	D	E
C	F	~~B~~ —
*D	G	H
E	B	C
F	B	C
G	~~B~~	I
H	J	K
*I	G	H
*J	G	H
~~K~~	~~K~~	~~K~~

S.T.F: $Q \times \Sigma \rightarrow Q$

	a	b
→ A,E,F	B	C
B	D,I,J	A,E,F
C	F	—
* D,I,J	G	H
G	—	D,I,J
H	D,I,J	∅

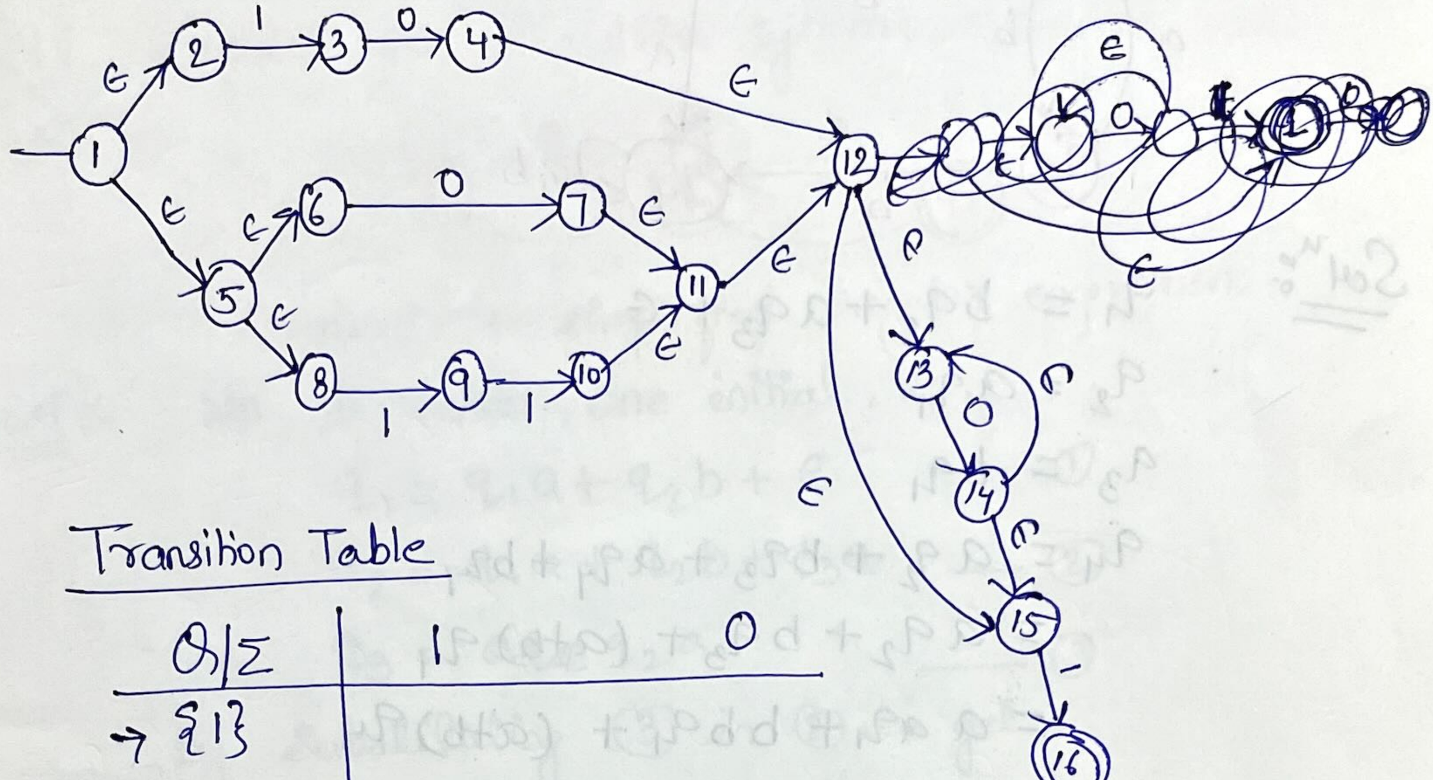




Minimized DFA

Q. Construct minimized DFA for regular expression,

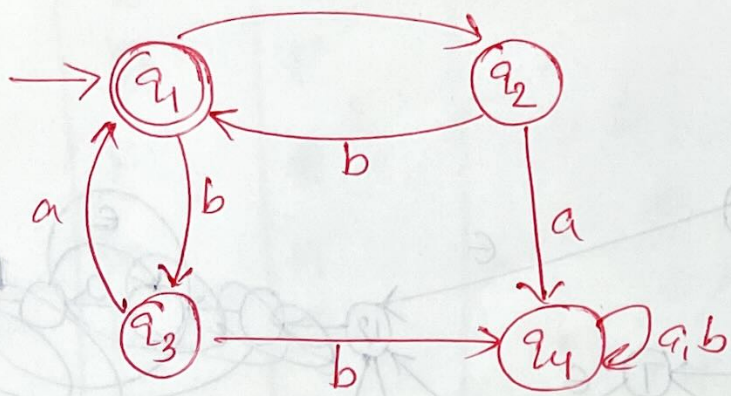
$r = (10 + (0 + 11))0^* 1$.



Transition Table

Q/Σ	1	0
→ {1}		

Q: Find the regular expression from the following DFA.



Soln:

$$q_1 = bq_2 + aq_3 + \epsilon$$
$$q_2 = aq_1$$
$$q_3 = bq_1$$
$$q_4 = aq_2 + bq_3 + aq_4 + bq_4$$
$$= aq_2 + bq_3 + (a+b)q_4$$
$$= aaq_1 + bbq_1 + (a+b)q_4$$
$$= (aa+bb)q_1 + (a+b)q_4$$

$$q_1 = baq_1 + abq_1 + \epsilon$$
$$= (ba+ab)q_1 + \epsilon$$

Final RE — $q_1 = \epsilon(ba+ab)^* = (ba+ab)^*$ ($\because R = Q + RP$, $R = QP^*$)

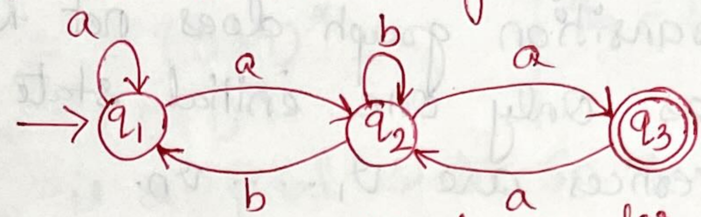
$$q_2 = a(ba+ab)^*$$
$$q_3 = b(ba+ab)^*$$
$$q_4 = (aa+bb)(ba+ab)^* + (a+b)q_4$$
$$q_4 = (aa+bb)(ba+ab)^*(a+b)^*$$

$q_1 = (ba+ab)^*$ RE for given DFA.

Arden's Theorem:-

$$R = Q + PR$$
$$\Rightarrow R = QP^*$$

Q: Consider the transition system,



Construct a simplified regular expression.

Soln:- No, ∈-moves, one initial, so,

$$q_1 = q_1 a + q_2 b + \epsilon \quad \text{—— (1)}$$
$$q_2 = q_1 a + q_2 b + q_3 a \quad \text{—— (2)}$$
$$q_3 = q_2 a \quad \text{—— (3)}$$

By substituting (3) in (2) we get,

$$q_2 = q_1 a + q_2 b + q_2 aa$$
$$\Rightarrow q_2 = q_1 a + q_2 (b + aa)$$
$$\Rightarrow q_2 = q_1 a (b + aa)^* \quad (\because R = Q + PR,\ R = QP^*)$$
$$\text{—— (4)}$$

Substitute (4) in (1) we get,

$$q_1 = q_1 a + q_1 a (b + aa)^* b + \Lambda$$
$$\Rightarrow q_1 = q_1 (a + a(b+aa)^* b) + \Lambda$$
$$q_1 = \Lambda (a + a(b+aa)^* b)^*$$

$$\therefore q_1 = \Lambda (a + a(b+aa)^* b)^*$$
$$q_2 = (a + a(b+aa)^* b)^* a (b+aa)^*$$
$$q_3 = (a + a(b+aa)^* b)^* a (b+aa)^* a$$

— String recognized by graph

Algebraic Method using Arden's Theorem

This is used to find the regular expression recognized by a transition system.

The following assumptions are made regarding the transition system:-

1) The transition graph does not have ϵ-moves.
2) It has only one initial state, say v_1.
3) Its vertices are $v_1 \ldots v_n$.
4) v_i is the regular expression representing the set of strings accepted by the system even though v_i is the final state.
5) α_{ij} denotes the regular expression representing the set of labels of edges from v_i to v_j. When there is no such edge then

$$\alpha_{ij} = \phi .$$

We can get the following set of equation in $v_1 \ldots\ldots v_n$.

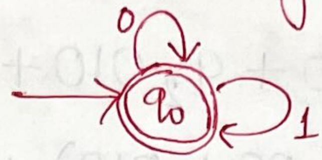
$$v_1 = v_1\alpha_{11} + v_2\alpha_{21} + \ldots v_n\alpha_{n1} + \Lambda$$

$$v_2 = v_1\alpha_{12} + v_2\alpha_{22} + \ldots v_n\alpha_{n2}$$

$$\vdots$$

$$v_n = v_1\alpha_{1n} + v_2\alpha_{2n} + \ldots v_n\alpha_{nn}.$$

Q: Represent the language accepted by the DFA shown:



$R = Q + PR$

$R = QP^*$

Sol^n:

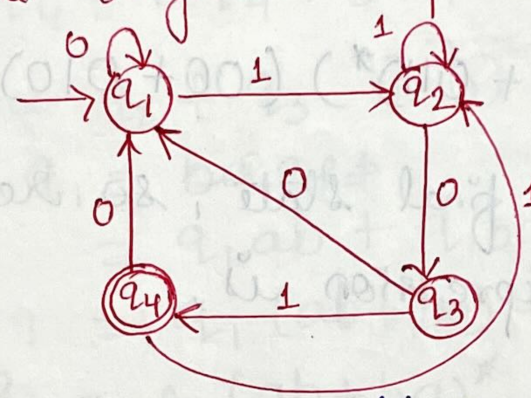
$$q_0 = 0q_0 + 1q_0 + \epsilon$$
$$= (0+1)q_0 + \epsilon$$
$$= \epsilon(0+1)^*$$
$$q_0 = (0+1)^*$$

$$q_0 = q_0 0 + q_0 1 + \epsilon$$
$$= q_0(0+1) + \epsilon$$
$$q_0 = (0+1)^*$$

Since, q_0 is a final state. So q_0, represents the final regular expression as

$$R = (0+1)^*$$

Q:- Find the regular expression corresponding to



Sol^n:- We form the equations,

$$q_1 = q_1 0 + q_3 0 + q_4 0 + \epsilon \quad \text{—— (1)}$$
$$q_2 = q_1 1 + q_2 1 + q_4 1 \quad \text{—— (2)}$$
$$q_3 = q_2 0 \quad \text{—— (3)}$$
$$q_4 = q_3 1 \quad \text{—— (4)}$$

Putting the value of q_3 in (4)

$$q_4 = q_2 01$$

Substituting q_4 in (2)

$$q_2 = q_1 1 + q_2 1 + q_2 011$$
$$= q_1 1 + q_2(1 + 011)$$
$$q_2 = q_1 1(1+011)^*$$

Putting q_3 & q_4 in eq. ① we get,

$$q_1 = q_1 0 + q_2 00 + q_2 010 + \Lambda$$
$$= q_1 0 + q_2 (00 + 010) + \Lambda$$
$$= q_1 0 + q_1 (1(1+011)^*)(00+010) + \Lambda$$
$$= q_1 (0 + (1(1+011)^*)(00+010)) + \Lambda$$

$$q_1 = \Lambda (0 + (1(1+011)^*)(00+010))^*$$ ✓

Put q_2 in q_4

$$q_4 = q_3 1 = q_2 01 = q_1 (1(1+011)^*)01$$

$$= (0 + (1(1+011)^*)(00+010))^* (1(1+011)^*)01$$

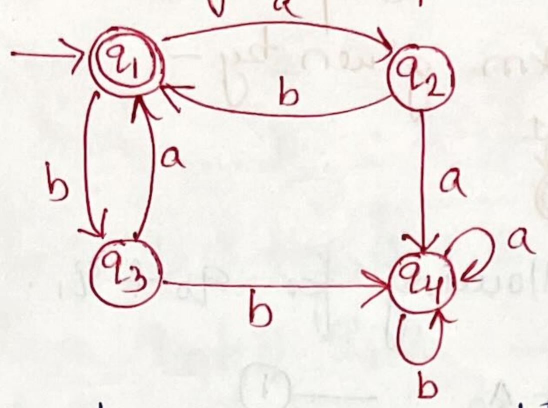
q_4 is the final state, so required regular expression is,

$$q_4 = (0 + (1(1+011)^*)(00+010))^* (1(1+011)^*)01$$

$\cancel{q_1 = (0 + 1(1+011)^* 0 + 1(1+011)^* 011)^*}$

$$q_2 = q_1 \, 1(1+011)^*$$
$$q_3 = q_1 \, 1(1+011)^* 0$$
$$q_4 = q_1 \, 1(1+011)^* 01$$

Find the regular expression for,



Sol^n :- We form the equation,

$$q_1 = bq_2 + aq_3 + \Lambda \quad \text{--- (1)}$$

$$q_2 = aq_1 \quad \text{--- (2)}$$

$$q_3 = bq_1 \quad \text{--- (3)}$$

$$q_4 = aq_4 + bq_4 + aq_2 + bq_3 \quad \text{--- (4)}$$

Replace q_2 & q_3 in (1)

$$q_1 = \cancel{baq_1 +}$$

$$= q_1 ab + q_1 ba + \Lambda$$

$$= q_1 (ab + ba) + \Lambda$$

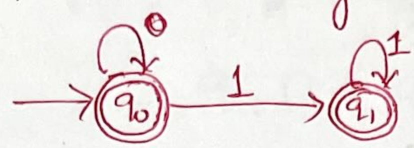
$$q_1 = \Lambda (ab + ba)^*$$

q_1 is the initial & final state both

So, ~~the~~ required regular expression is,

$$(ab + ba)^*$$

Q8- Construct a regular expression corresponding to transition system given by -



Soln: We have the following for q_0 & q_1.

$$q_0 = q_0 0 + \Lambda \quad \text{---} (1)$$

$$q_1 = q_0 1 + q_1 1 \quad \text{---} (2)$$

~~Substitute (1) in (2)~~

By applying Arden's Theorem to eq.n (1)

$$q_0 = \Lambda + q_0 0$$

$$q_0 = 0^*$$

Put the value of q_0 in eq.n (2) we get,

$$q_1 = 0^* 1 + q_1 1$$

$$q_1 = (0^* 1) 1^* \quad \left(\because P = Q + RP,\ P = QP^* \right)$$

As the transition system has both q_0 & q_1 as final states,

then

$$q_0 + q_1 = 0^* + 0^* 1 1^*$$

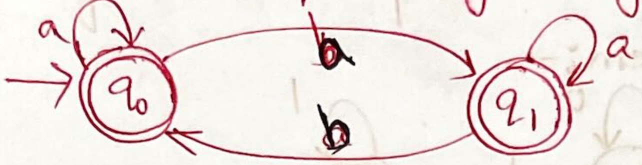
$$= 0^* (\Lambda + 1 . 1^*)$$

$$= 0^* (\Lambda + 1^*)$$

$$= 0^* 1^*$$

orresponde:

Find corresponding regular expression,



Solⁿ:

$$q_0 = q_0 a + q_1 b + \epsilon \quad \text{— (1)}$$

$$q_1 = q_0 b + q_1 a \quad \text{— (2)}$$

By Arden's Theorem on eqⁿ (2) we get,

$$q_1 = q_0 b a^*$$

Put the value of q_1 in (1)

$$q_0 = q_0 a + q_0 ba^* b + \epsilon$$

$$q_0 = q_0 a + q_0 ba^* b + \epsilon$$

$$q_0 = q_0 (a + ba^* b) + \epsilon$$

$$q_0 = \epsilon (a + ba^* b)^* \quad \text{(By Arden's Theorem)}$$

$$q_0 = (a + ba^* b)^*$$

$$\therefore q_1 = (a + ba^* b)^* ba^*$$

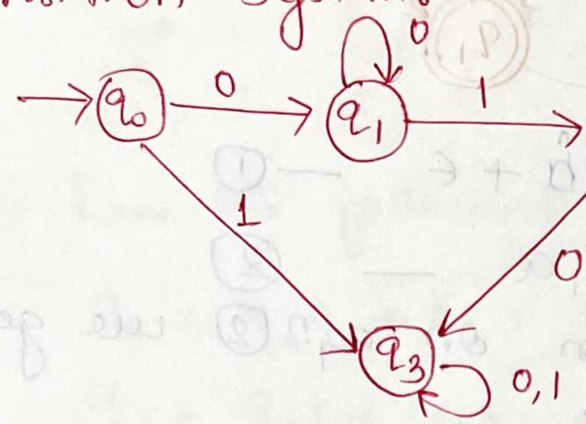
q_0 & q_1 both are final states.

$$\therefore q_0 + q_1 = (a + ba^* b)^* + (a + ba^* b)^* ba^*$$

$$= (a + ba^* b)^* (\Lambda + ba^*)$$

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

Q:- Find the regular expression for given transition system:-



Soln:-

$q_0 = \Lambda$ ——— (1)

$q_1 = q_0 0 + q_1 0$ ——— (2)

$q_2 = q_1 1 + q_2 1$ ——— (3)

$q_3 = q_0 1 + q_2 0 + q_3 0 + q_3 1$ ——— (4)

Put the value of q_0 in eq. (2)

$q_1 = \Lambda 0 + q_1 0$

$q_1 = 0 + q_1 0$

$q_1 = q_1 0 + 0$ (By Arden's rule)

$q_1 = 00^*$

Put the value of q_1 in eqn (3)

$q_2 = 00^* 1 + q_2 1$

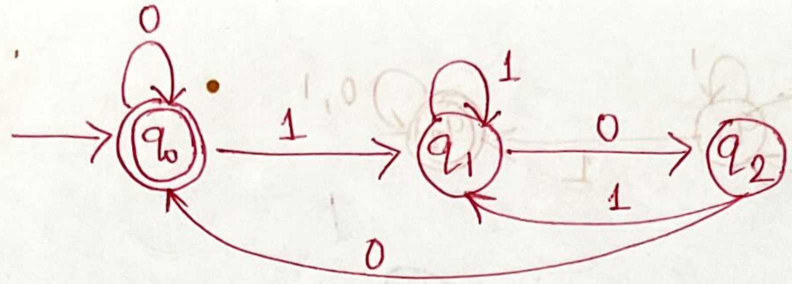
$q_2 = 00^* 1 1^*$ (By Arden's rule)

Since q_2 is final state.

So, ~~String~~ r.e for the given FA is,

$\boxed{r.e = 00^* 1 1^*}$

given:



Sol^n:

$$q_0 = q_0 0 + q_2 0 + \epsilon \quad \text{--- (1)}$$

$$q_1 = q_0 1 + q_1 1 + q_2 1 \quad \text{--- (2)}$$

$$q_2 = q_1 0 \quad \text{--- (3)}$$

Put the value of q_2 in eq^n (2)

$$q_1 = q_0 1 + q_1 1 + q_1 01$$

$$q_1 = q_0 1 + q_1 (1 + 01)$$

$$q_1 = q_0 1 (1 + 01)^* \quad \text{(By Arden's Rule)} \quad \text{--- (4)}$$

Now, put the value of q_2 in eq^n (1)

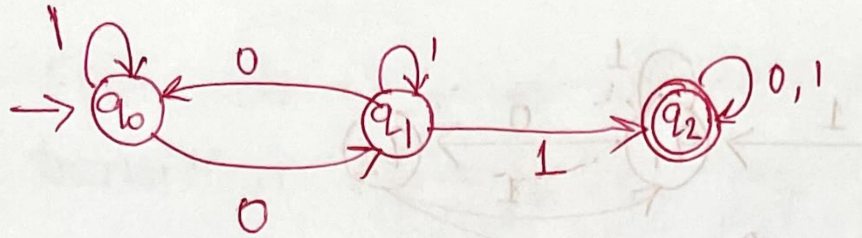
$$q_0 = q_0 0 + q_1 00 + \epsilon$$

$$= q_0 0 + q_0 1 (1 + 01)^* 00 + \epsilon \quad \text{(from eq}^n \text{ (4))}$$

$$= q_0 (0 + 1 (1 + 01)^* 00) + \epsilon$$

$$q_0 = (0 + 1 (1 + 01)^* 00)^* \quad \text{(By Arden's Theorem)}$$

Q:



Solⁿ:

$q_0 = q_0 1 + q_1 0 + \epsilon$ — (1)

$q_1 = q_0 0 + q_1 1$ — (2) $\Rightarrow q_1 = q_0 0 1^*$ — (4)

$q_2 = q_1 1 + q_2 0 + q_2 1$ — (3)

$\Rightarrow q_2 = q_1 1 + q_2 (0+1)$

$\Rightarrow q_2 = q_1 1 (0+1)^*$ — (5)

~~Substitute (4) in~~ ~~— final Regular Expression~~

Substitute (4) in (1)

~~Regular Expression accepted by given FA is~~,

$q_0 = q_0 1 + q_0 0 1^* + \epsilon$

~~$q_2 = q_1$~~

$q_0 = \epsilon + q_0 (1 + 01^*)$

$q_0 = (1+01^*)^*$

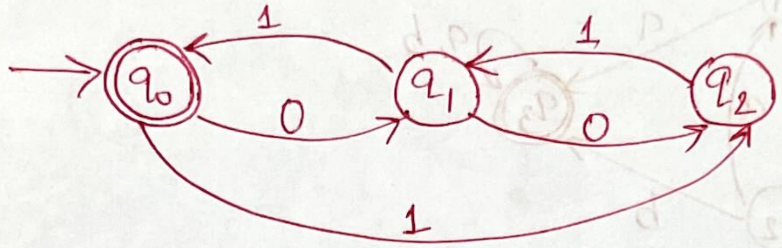
$\therefore q_1 = (1+01^*)^* 01^*$

Hence final regular expression is given by

$q_2 = q_1 1 (0+1)^*$

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

$$\boxed{RE \Rightarrow (1+01^*)^* 01^* 1 (0+1)^*}$$



9

$q_0 = q_1 1 + \epsilon$ — (1)

$q_1 = q_0 0 + q_2 1$ — (2)

$q_2 = q_0 1 + q_1 0$ — (3)

Substitute (2) in (3) we get,

$q_2 = q_0 1 + q_0 00 + q_2 10$

$q_2 = q_0 (1+00)(10)^*$ — (4)

Substitute (2) in (1)

$q_0 = (q_0 0 + q_2 1).1 + \epsilon$ — (5)

Substitute (4) in (5) we get,

$q_0 = [q_0 0 + [q_0 (1+00)(10)^*].1]\,1 + \epsilon$

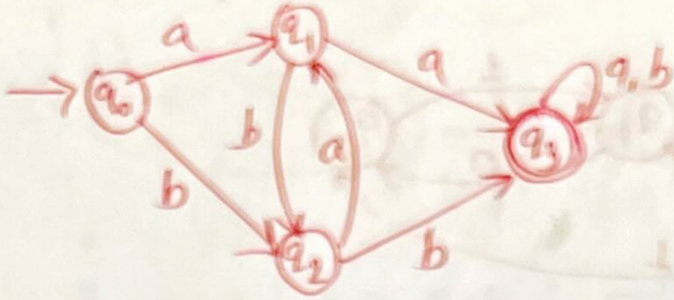
$q_0 = q_0 [0 + (1+00)(10)^* 1]\,1 + \epsilon$

By Arden's theorem,

$q_0 = ((0 + (1+00)(10)^* 1)\,1)^*$

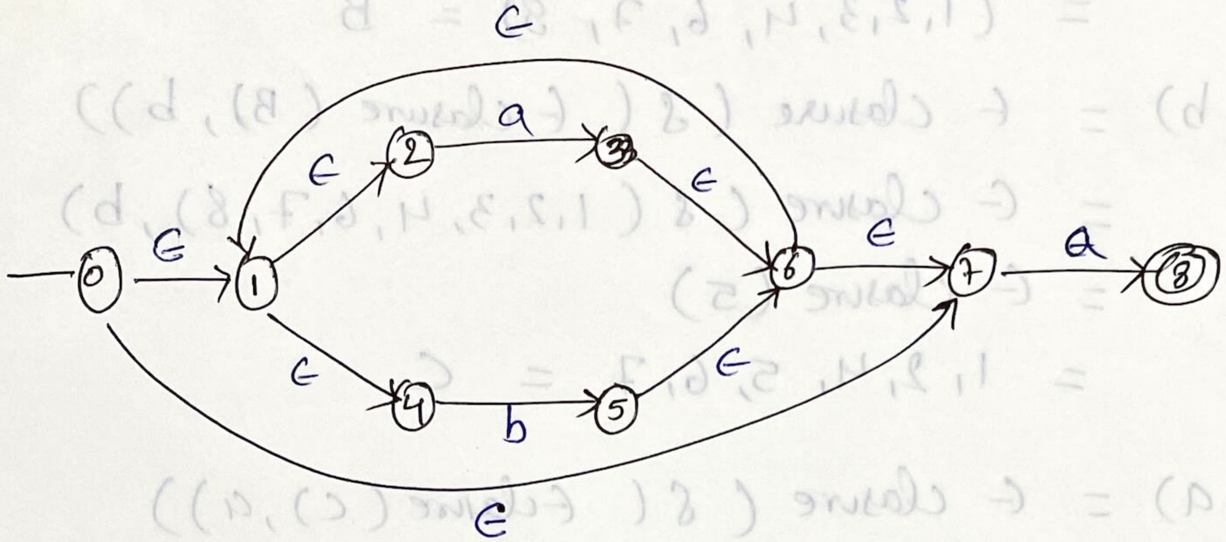
— final RE as q_0 is final state

Q:



Q: Convert the following Regular Expression to DFA.
$(a+b)^* a$.

Soln:
$\Sigma = \{a, b\}$



Subsets are constructed on the basis of ∈-moves.

$(A) = \{0, 1, 2, 4, 7\} = \epsilon\ closure(0)$

$$\delta(A, a) = \epsilon\ closure(\delta(\epsilon\ closure(0) \cancel{1,2,4,7}, a))$$
$$= \epsilon\ closure(\delta(0,1,2,4,7), a)$$
$$= \epsilon\ closure(3, 8)$$
$$= (1, 2, 3, 4, 6, 7, 8) \text{ — } B$$

$$\delta(A, b) = \epsilon\ closure(\delta(\epsilon\ closure(0), b))$$
$$= \epsilon\ closure(\delta(0,1,2,4,7), b)$$
$$= \epsilon\text{-}closure(5)$$
$$= (1, 2, 4, 5, 6, 7) \text{ — } C$$

$\delta(B, a) = \epsilon\text{ closure}(\delta(\epsilon\text{ closure}(B), a))$

$= \epsilon\text{ closure}(\delta(1, 2, 3, 4, 6, 7, 8), a)$

$= \epsilon\text{ - closure}(3, 8)$

$= (1, 2, 3, 4, 6, 7, 8) = B$

$\delta(B, b) = \epsilon\text{ closure}(\delta(\epsilon\text{ closure}(B), b))$

$= \epsilon\text{ closure}(\delta(1, 2, 3, 4, 6, 7, 8), b)$

$= \epsilon\text{ closure}(5)$

$= 1, 2, 4, 5, 6, 7 = C$

$\delta(C, a) = \epsilon\text{ closure}(\delta(\epsilon\text{ closure}(C), a))$

$= \epsilon\text{ closure}(\delta(1, 2, 4, 5, 6, 7), a))$

$= \epsilon\text{ closure}(3, 8)$

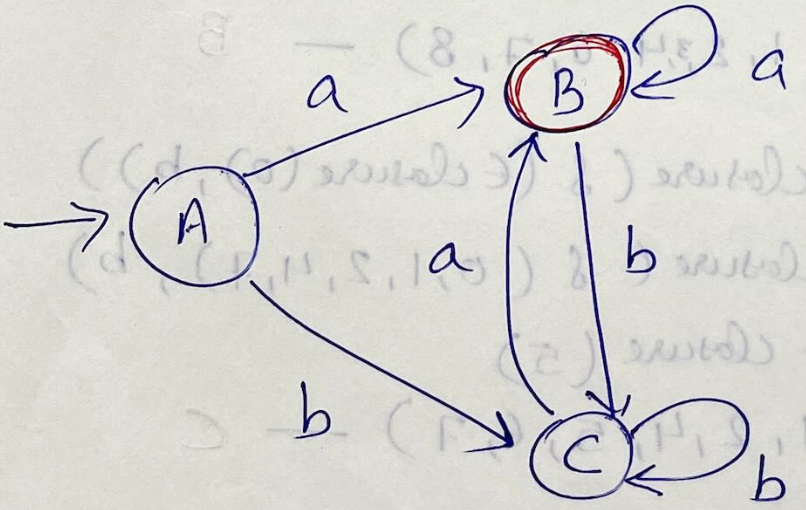
$= (1, 2, 3, 4, 6, 7, 8) = B$

$\delta(C, b) = \epsilon\text{ closure}(\delta(\epsilon\text{ closure}(C), b))$

$= (\epsilon\text{ closure}(\delta(1, 2, 4, 5, 6, 7), b))$

$= \epsilon\text{ closure}(5)$

$= 1, 2, 4, 5, 6, 7 = C$



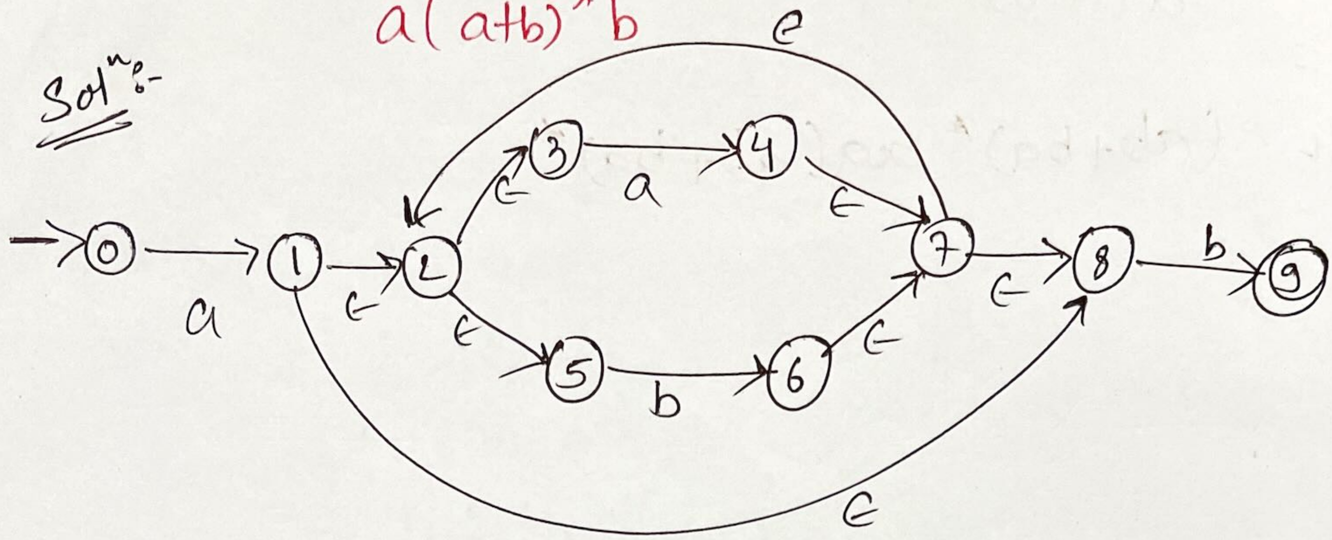
Q: $a(a+b)^*b$

Q: $(ab+ba)^*aa(ab+ba)^*$

Q: Construct minimized DFA.

$a(a+b)^*b$

Sol^n:-



ϵ closure $(0) = 0$ — A

$\delta(A, a) = \epsilon\,closure(\delta(\epsilon\,closure(A), a))$
$= \epsilon\,closure(\delta(0, a))$
$= \epsilon\,closure(1) = \{1, 2, 3, 5, 8\}$ — B

$\delta(A, b) = \epsilon\,closure(\delta(\epsilon\,closure(A), b))$
$= \epsilon\,closure(\delta(0, b))$
$= \epsilon\,closure(\epsilon) = \epsilon$

$\delta(B, a) = \epsilon\,closure(\delta(\epsilon\,closure(B), a))$
$= \epsilon\,closure(\delta(1, 2, 3, 5, 8), a)$
$= \epsilon\,closure(4) = (2, 3, 4, 5, 7, 8)$ — C

$\delta(B, b) = \epsilon\,closure(\delta(\epsilon\,closure(B), b))$
$= \epsilon\,closure(\delta(1, 2, 3, 5, 8)\ b)$
$= \epsilon\,closure(6, 9) = (2, 3, 5, 6, 7, 8, 9)$ — D

~~E-closure~~

ϵ - closure (0) = 0 - A

$\hat{\delta}(A, a) = \epsilon$ closure ($\delta(A, a)$)

= ϵ - closure (1) = 1, 2, 3, 5, 8 - B

$\hat{\delta}(A, b) = \epsilon$ - closure ($\delta(A, b)$)

= ϵ closure (ϕ) = ϕ

$\hat{\delta}(B, a) = \epsilon$ closure ($\delta(B, a)$)

= ϵ closure (4) = 4, 7, 8, 2, 3, 5 - C

$\hat{\delta}(B, b) = \epsilon$ closure (6, 9) = 6, 7, 2, 3, 5, 8, 9 - D

$\hat{\delta}(C, a) = \epsilon$ closure ($\delta(C, a)$)

= ϵ closure (4) = C

$\hat{\delta}(C, b) = \epsilon$ - closure ($\delta(C, b)$)

= ϵ closure (6, 9) = ~~D~~ D

	a	b
A	B	ϕ
B	C	D
C	C	D
D	C	D

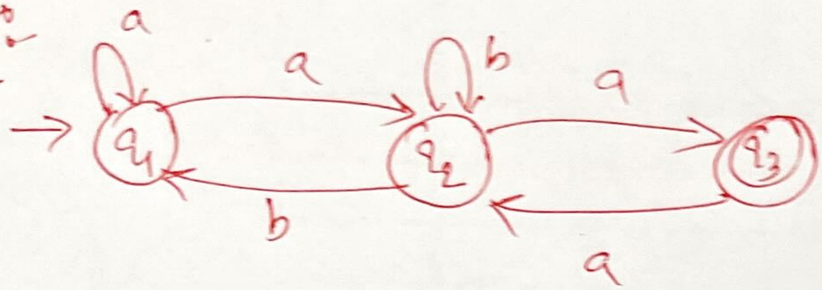
$\hat{\delta}(D, a) = \epsilon$ closure ($\delta(D, a)$)

= ϵ closure (4) = C

$\hat{\delta}(D, b) = \epsilon$ closure ($\delta(D, b)$)

= ϵ closure (6, 9) = D

Q:



$$R = Q + PR$$

$$R = QP^*$$

$$q_1 = \cancel{aq_1 + bq_2 + \epsilon}\ \ q_1 a + q_2 b + \epsilon \quad \text{— (1)}$$

$$q_2 = \cancel{aq_1 + bq_2 + aq_3}\ \ q_1 a + q_2 b + q_3 a \quad \text{— (2)}$$

$$q_3 = \cancel{aq_2}\ q_2 a \quad \text{— (3)}$$

• Substitute eq^n 3 in eq^n (1)

$$q_2 = \cancel{aq_1 + bq_2 + aaq_2}\ q_1 a + q_2 b + q_2 aa$$

$$= \cancel{aq_1 + (b+aa) q_2}\ \ q_1 a + q_2 (b+aa)$$

$$= q_1 a + q_2 (b+aa)$$

$$q_2 = q_1 a (b+aa)^* \quad \text{— (4)}$$

Substitute eq^n 4 in eq^n (1)

$$q_1 = q_1 a + q_1 a (b+aa)^* b + \epsilon$$

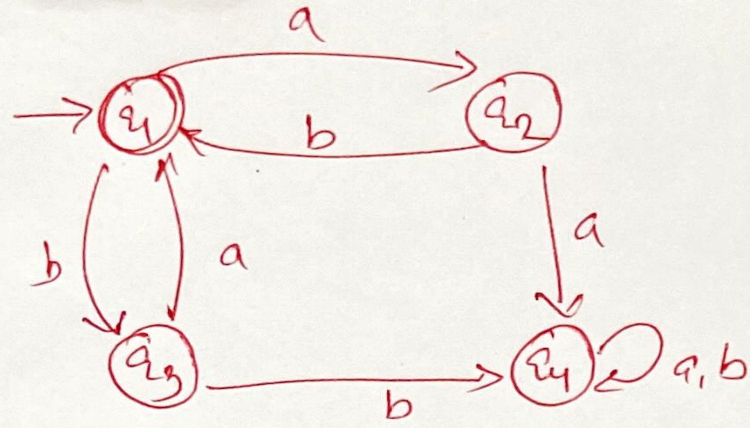
$$= q_1 (a + a(b+aa)^* b)) + \epsilon$$

$$q_1 = (a + a(b+aa)^* b)^*$$

$$q_2 = (a + a(b+aa)^* b)^* . a(b+aa)^*$$

$$q_3 = (a + a(b+aa)^* b)^* a (b+aa)^* a$$

Q:



$q_1 = q_2 b + q_3 a + \epsilon$

$q_2 = q_1 a$

$q_3 = q_1 b$

$q_4 = q_2 a + q_3 b + q_4 a + q_4 b$

$q_1 = q_1 ab + q_1 ba + \epsilon$

$\quad = q_1 (ab + ba) + \epsilon$

$q_1 = (ab + ba)^*$ RE for the given DFA.

Pumping Lemma

Non Regular language:- A language that cannot be defined by regular expression is called a non-regular language. The languages which are not regular are called as non-regular languages.

eg: $L = \{a^n b^n : n \geq 0\}$

$L = \{ww^R : w \in \Sigma^*\}$

$L = \{a^{n!} : n \geq 0\}$

$L = \{0^{n^2} : n \geq 0\}$

Pumping Lemma:- The fundamental tool for proving that a language is

- not regular
- not context-free

is known as pumping lemma.

Steps in Pumping lemma to prove that the given language is not regular:-

Step 1: Assume that the language L is regular.

Step 2: Consider the DFA which has 'm' states.

Step 3: Choose w, such that, $w \in L$ with $|w| \geq m$.

Step 4: Consider $w = xyz$ such that:

i> $|y| \geq 1$

ii> $|xy| \leq m$

iii> Show that $xy^iz \notin L$ ~~we arrive at a contradiction~~ for $i \geq 0$.

Step 5: Thus, by picking i and showing that $xy^iz \notin L$, we arrive at a contradiction and conclude that the assumption in step 1 is false. Hence, the language is not regular.

Q: Show that the language
$L = \{a^n b^n : n \geq 0\}$ is not regular.

Soln: Using the pumping lemma for
$L = \{a^n b^n : n \geq 0\}$ assume that L is regular.

Let m be the integer in the pumping lemma.

Pick a string
$w = a^m b^m$ such that $w \in L$ and $|w| \geq m$.

we can write $w = a^m b^m$ as

$$a^m b^m = xyz.$$

from the pumping lemma, it follows that

Length $|xy| \leq m$, $|y| \geq 1$.

$\therefore a^m b^m = a^{m-k} a^k b^m$ where $k \geq 1$.

$\therefore xyz = a^m b^m = a^{m-k} a^k b^m$

$x = a^{m-k}$ $y = a^k$ $z = b^m$

from the lemma:

$xy^iz \in L$ $i = 0, 1, 2, \ldots$

$w =$

when $n = 0$ $\quad w = 11$

when $n = 1$ $\quad w = ab$

$n = 2$ $\quad w = aabb$

$w = xyz$ $\therefore x = a \quad y = ab \quad z = b$ | $x = aa \quad y = b \quad z = b$

$xy^iz \in L$ for $i = 0, 1, 2, 3$

xy^iz

when $i = 0$ $\quad L_1 = xz = ab$

$i = 1$ $\quad L_2 = aabb$

$i = 2$ $\quad L_3 = aababb$

$L_3 \notin L$.

$i = 1$ $\quad w = aabb \in L$

$i = 2$ $\quad w = aabbb \notin L$

So, our assumption is wrong and the language $L = \{a^n b^n : n \geq 0\}$ is not regular.

Q2: Show that the language $L = \{ww^R : w \in \Sigma^*\}$ is not regular.

Sol:- Let us consider that the language L is regular and let n be any integer. Now, we choose w such that $|w| \geq n$.

Suppose $w = 0^n 1^n 1^n 0^n$

for pumping lemma $|xy| \leq n$.

$\therefore$ Both x & y must be made of all 0's because the first n letters of w are all 0's.

$$\overbrace{000\ldots00}^{n}\ \overbrace{1111\ldots1}^{n}\ \overbrace{1111\ldots1}^{n}\ \overbrace{000\ldots0}^{n}$$

$x \qquad y \qquad z$

Now, when we form the word xyz then it is OK for n 0's n 1's n 1's & n 0's.

For the word xy^2z

i.e $xyyz$, we are adding more 0's in front of the w but we are not adding that much no. of 0's at the end in z part.

This means that $xy^2z \notin L$.

Hence, by pumping lemma

$L = \{ww^R : w \in \Sigma^*\}$ is not regular.

Q3:- Prove that the language $L = \{a^n b a^n$ for $n = 0, 1, \ldots\}$ is not regular.

Soln: Let us assume that the language L is regular. The pumping lemma says that the if the language is regular then there must be strings x, y, z such that all the words of the form xy^iz are in L.

Case 1: If $x = a^n$ $y = b$ $z = a^n$

Then $xy^iz \in L$

for $i = 2$ $= a^n bb a^n$, two b's will be there in the middle.

$a^n bb a^n \notin L$.

Case 2: If y string contain all a's, then b in the middle of the word xyz is in x side or z-side.

If we consider the string $xyyz$ it will increase the no. of a's either in front of b or after b i.e not possible for given L (equal no. of a required.)

So, $L = \{a^n b a^n : n = 0, 1, 2, \ldots\}$ is not regular.

Q: Prove that $L = \{0^n 1^m : n \leq m\}$ is not regular.

Sol^n: Let w be the string of L and

Case 1:- $w = 0^p 1^p$ and is of length at least p.

p when $n = m$.

$w = \underbrace{00 \overbrace{0 \ldots 0}^{p}}_{x} \underbrace{\overbrace{1}_{y} 11 \ldots 1}_{z}$

~~let $x = 00$ $y = 0$~~

$xy^2z = xyyz$ = It will have more 0 compared to 1 so there will not be equal no. of 0's & 1's.

Case 2: when $n < m$.

Suppose $n = 2$ $m = 3$

$w = \underbrace{00}_{x}\underbrace{1}_{y}\underbrace{11}_{z}$

$xyyz = 0010111$ there should not be the case where 1 is ~~preceded~~ followed by 0.

So $L = \{0^n 1^m : n \leq m\}$ is not regular.

Q: Prove that $L = \{0^n 1^m 2^n : n, m \geq 0\}$ is not regular.

Sol^n: **Case 1:** Suppose $n = m = 2$

$w = \underbrace{00}_{x}\underbrace{11}_{y}\underbrace{22}_{z}$

$xy^2z = xyyz = 00111122 \in L.$

$w = \underbrace{001}_{x}\underbrace{12}_{y}\underbrace{2}_{z}$

$xyyz = 00112122 \notin L$. 0 followed by 1 followed 2.

So, given L is not regular.

Case II: $0^n 1^m 2^n \quad n, m \geq 0$

Let us suppose that $n = 2 \quad m = 0$

$w = 0022$

$w = xyz$, since $|xy| \leq n$

Let $x = 0 \quad y =$

$n = 2 \; m = 1 \quad w = \underset{x}{0}\underset{y}{0}\underset{z}{122}$

$xyz = 00122$

$xyyz = 000122$

0 & 2 are not equal in no.

Let us assume that L is regular and try to obtain a contradiction.

Let n be the constant provided by the pumping lemma,

Let $w = 0^n 2^n \; (m = 0)$

The string w is in L and is of length atleast 2n.

So, $w = xyz \quad |xy| \leq n \quad y \neq \wedge$

Clearly, xy will contain only 0's since $w = 0^n 2^n$ and in this, first 'n' places are occupied by the 0's only. So 'y' will be made of 0 only.

$xy^i z$ when $i = 2 \quad xyyz$

the no. of zero will be more but the no. of zero should be equal to total no. of 2.

∴ L is not regular.

Q: Prove that $L = \{0^n : n \text{ is perfect Square}\}$ is not regular.

Soln: Suppose that L is regular.

Values for n = 1, 4, 9, 16 ....

$n=1 \quad w=0$

$n=4 \quad w = 0000 = xyz$

(i) Let $y = 00 \quad z = 0 \quad x = 0$

$w = 0000$
$x = \epsilon \; y = 00 \; z = 00$
$xyz = 0000 \in L$
$xy^2z = 000000 \notin L$

$xy^iz \in L$ — According to Lemma

$i = 1 \quad w = xyz = 0000 \in L$

$i = 2 \quad w = xyyz = 000000 \notin L$

(ii) ∴ Given L is not regular.

Q: Prove that $L = \{0^n 1^{2n}\}$ where $n \geq 1$ is not regular.

Soln: Suppose that L is regular.

for $n=1 \quad L = 011$

Let $y = 1$

$w = xyz = 011 \in L$

$xy^2z = 0111 \notin L$

$x = \epsilon \quad y = 0 \quad z = 11$
$xyz = 011 \in L$
$xyyz = 0011 \notin L.$

Because no. of 1 should be twice as no. of 0's.

$y = 0$

$xyz = 011 \in L$

$xyyz = 0011 \notin L$

∴ L is not regular.

Q: Show that the set

$L = \{a^p : p \text{ is prime number}\}$ is not regular.

Soln: Suppose that L is a regular language. Let n be the constant value by pumping lemma.

Possible values for $n = 2, 3, 5, 7, 11, 13, \ldots$

when $n = 2$

$w = xyz = a^2 = aa$

$y = a \quad z = a$

$xy^2z = aaa \in L$

$xy^3z = aaaa \notin L$

So, the given language L is not regular.

Q: Using pumping lemma show that following sets are not regular:-

ⓐ $\{a^n b^{2n} : n > 0\}$

ⓑ $\{a^n b^m : 0 < n < m\}$

ⓐ Suppose L is regular. Let p be the con'

$w = a^p b^{2p}$

when $p = 2 \quad w = aabbbb$

$w = xyz$

$x = aa \quad y = b \quad z = bbb$

$xy^2z = aabbbbb$

Pumping Lemma

① Prove that the language
$L = \{ a^n b^n \text{ for } n = 0, 1, 2, \ldots \}$ is not regular.

② Prove that $L = \{ a^n b a^n \text{ for } n = 0, 1, 2, \ldots \}$ is not regular.

③ Prove that $L = \{ 0^n 1^m : n \leq m \}$ is not regular

④ Prove that $L = \{ 0^n 1^m 2^n : n, m \geq 0 \}$ is not regular.

⑤ Prove that $L = \{ 0^n : n \text{ is perfect square} \}$ is not regular.

⑥ $\{ a^n b^{2n} : n > 0 \}$

⑦ $\{ a^{n^2} : n \geq 0 \}$

⑧ $\{ a^n b^n c^n : n \geq 1 \}$ not regular

xy^iz $\quad x = \wedge \quad z = \wedge$

$y = a$

$w = \wedge (a)^i \wedge$

$i = 0 \quad w = \wedge$ in L

$i = 1 \quad w = a$ in L

$i = 2 \quad w = aa$ not in L.

Pumping Lemma Theorem :-

Let $M = (Q, \Sigma, \delta, q_0, F)$ be a FA with n states
Let L be the regular set accepted by M.
Let $w \in L$ and the $|w| \geq m$. If $m \geq n$,
then there exists x, y, z such that

$w = xyz$, $y \neq \epsilon$ and $xy^iz \in L$ for each $i \geq 0$.

or

Let L be the regular language, then there exists a constant 'n' such that for every string w in L such that $|w| \geq n$. we can break w into three substrings
$w = xyz$ such that

1) $y \neq \epsilon$ 2) $|xy| \leq n$ 3) for all $i \geq 0$ the string xy^iz is also in L.

Pumping Lemma for Context free language

If L is any Context free language, then we can find a natural no. 'n' such that,

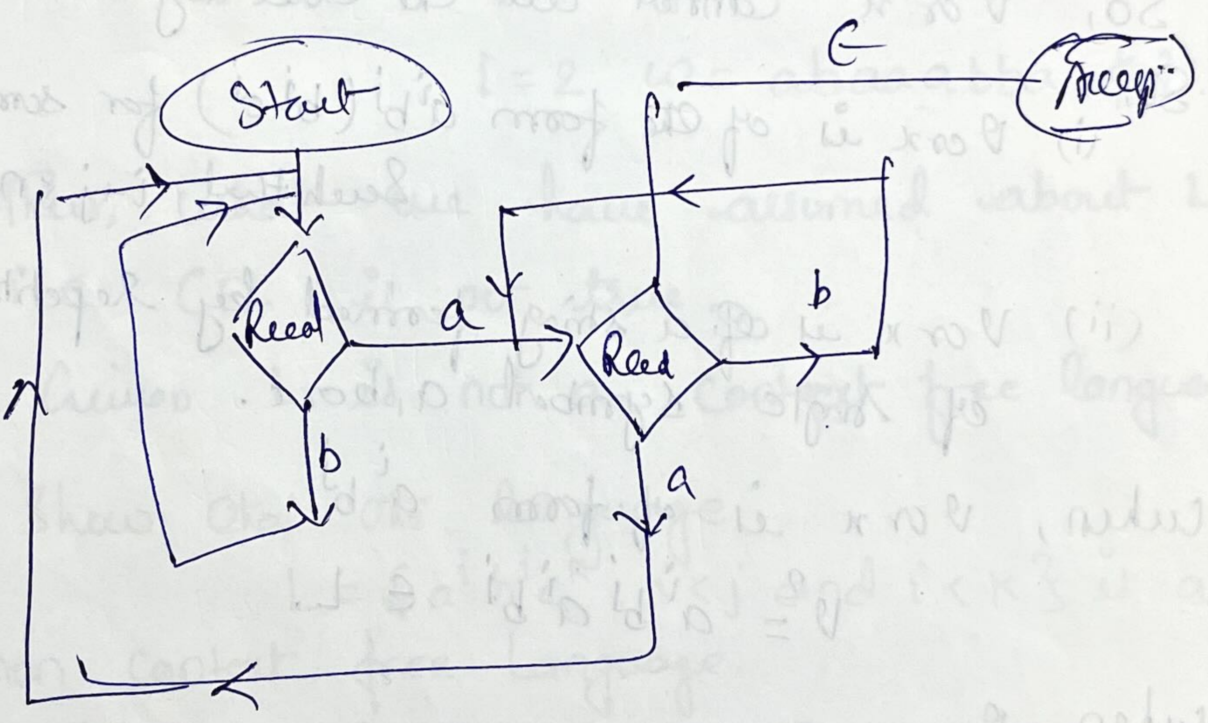
(i) every $z \in L$, where $|z| \geq n$.

(ii) $w = uvwxy$

(iii) $|vx| \geq 1$

(iv) $|vwx| \leq n$

(v) $uv^i wx^i y \in L$, $i \geq 0$.



Q1: Show that the language

$L = \{a^n b^n c^n : n \geq 0\}$ is not a Context free language.

Soln: Assume that L is context free. Let n be the natural no. obtained by pumping lemma.

Let $z = a^n b^n c^n$

$|z| = 3n \geq n$

Write $z =$ ~~uvxy~~ $uvwxy$ where $|vx| \geq 1$.

i.e atleast one of v or x is not ϵ.

$\therefore$ ~~uvxy~~ $uvwxy = a^n b^n c^n$

$|vwx| \leq n$

as $1 \leq |vx| \leq n$

So, v or x cannot all the three symbols a, b, & c.

So,

(i) v or x is of the form $a^i b^j$ ($b^i c^j$) for some i & j.

Such that $i+j \leq n$.

(ii) v or x is ~~of~~ a string formed by repetition of single symbol a, b or c.

when, v or x is of form $a^i b^j$

$v^2 = a^i b^j a^i b^j \notin L$.

when v

Q: Prove that the language $L = \{SS^T : S \in (a,b)^*\}$ is context free or not.

Soln:- Even palindrome $L = SS^T$

$w = pqrst \in L$

According to lemma:-

① $|qs| \geq 1$ ② $|qrs| \leq n$.

③ $w = pq^i rs^i t \in L$

Suppose $w = abaaba$

Lets group the input,

$w = (ab)(a)(a)(b)(ba)$

$pq^i rs^i t$ $i=1$ $w = abaaba \in L$

$i=2$ $w = abaaabba \notin L$.

Thus, what we have assumed about L. i-e CFL is not true.

Hence L is not a context free language.

Q: Show that the language

$L = \{a^i b^j c^k : i<j \text{ and } i<k\}$ is a non context free language.

Soln: Suppose $i=1$ $j=2$ $K=3$

$w = abbccc$

$w = pqrst$

① $|qs| \geq 1$ ② $|qrs| \leq n$

Closure Properties of Regular Languages

1) The set of regular languages are closed under union, ~~Intersection~~ concatenation and Kleene closure.

2) The set of regular languages is closed under complements.

3) The set of regular language is closed under intersection. L_1 & L_2 are R.L

then $L_1 \cap L_2$ is also R.L.

$M_1 = (Q_1, \Sigma, \delta_1, q_1, F_1)$ — DFA for L_1

$M_2 = (Q_2, \Sigma, \delta_2, q_2, F_2)$ — DFA for L_2

Let $M = (Q, \Sigma, \delta, q_0, F)$ be the DFA accepting $L_1 \cap L_2$. M is defined as,

$Q = Q_1 \times Q_2$

$q_0 = \{q_1, q_2\}$

$F = F_1 \times F_2$

and for all p_1 in Q_1 and p_2 in Q_2 and a in Σ

$$\delta((p_1, p_2), a) = (\delta_1(p_1, a), \delta_2(p_2, a))$$

4) L_1 & L_2 are regular then $L_1 - L_2$ is also regular.

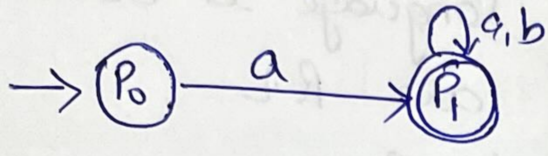
5) L_1 is closed under reversal.

Q: Construct a DFA which accepts over $\Sigma = \{a, b\}$ that accepts the set of strings starts with 'a' and ends with 'b'.

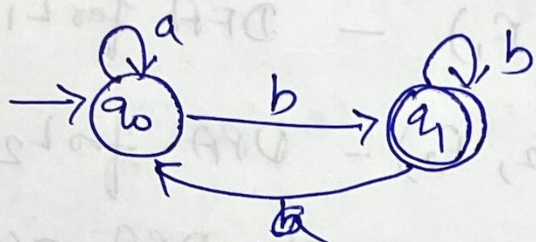
Sol^n: $L_1 = \{w : w \text{ starts with 'a'}\}$

$L_2 = \{w : w \text{ ends with 'b'}\}$

The DFA M_1 which L_1 is



The DFA M_2 which accepts L_2 is -



DFA M which accepts $L_1 \cap L_2$ is constructed as follows:-

Since $Q = Q_1 \times Q_2$

$= \{P_0, P_1\} \times \{q_0, q_1\}$

$= \{(P_0, q_0), (P_0, q_1)\ (P_1, q_0)\ (P_1, q_1)\}$

$q_0 = \{P_0, q_0)$

$F = \{P_1, q_1)$

$\delta'((P_0, q_0), a) = (P_1, q_0)$

$\delta'((P_0, q_0), b) = (\phi, q_1) = \phi$

$\delta'((P_0, q_1), a) = (P_1, q_0)$

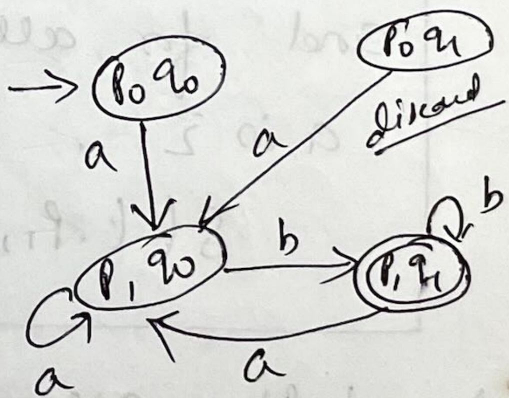
$\delta'((P_0, q_1), b) = (\phi, q_1) = \phi$

$\delta'((P_1, q_0), a) = (P_1, q_0)$

$\delta'((P_1, q_0), b) = (P_1, q_1)$

$\delta'((P_1, q_1), a) = (P_1, q_0)$

$\delta'((P_1, q_1), b) = (P_1, q_1)$

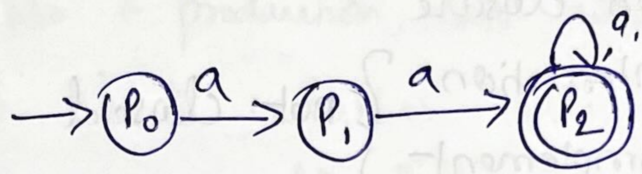


Q: Design DFA which accepts the set of Strings starts with 'aa' but not end with 'bb'.

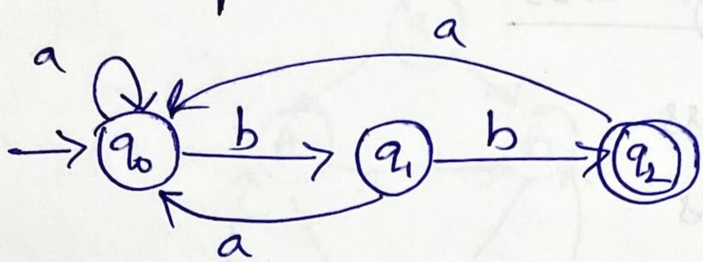
Sol^n:

M_1

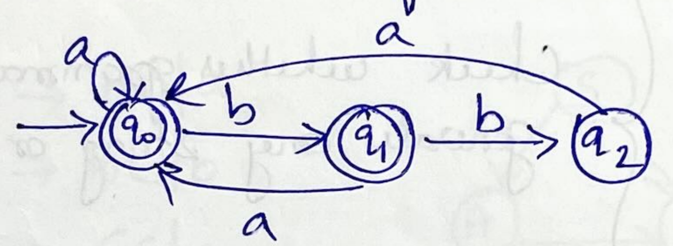
L_1 = Starts with 'aa'



DFA for end with bb



DFA which does not end with bb we will have to find its complement.



$Q = Q_1 \times Q_2$

$= \{P_0, P_1, P_2\} \times \{q_0, q_1, q_2\}$

$= (P_0, q_0)\ (P_0\, q_1)\ (P_0\, q_2)\ (P_1, q_0)\ (P_1, q_1)\ (P_1, q_2)$

$(P_2, q_0)\ (P_2, q_1)\ (P_2, q_2)$

Closure Properties of CFL :-

The Context free languages are closed under

- Union
- Concatenation
- Kleen closure
- Intersection } not closed
- Complement }

Decision Algorithm for CFL's

- Emptiness
- finiteness
- Membership

① $S \rightarrow PQ$
$P \rightarrow AP \mid AA$
$A \rightarrow a$
$Q \rightarrow BQ \mid BB$
$B \rightarrow b$

} Check whether grammar generates any string or not.

Soln:

$S \rightarrow PQ$
$P \rightarrow AP$
$P \rightarrow AA$
$A \rightarrow \dot{a}$
$Q \rightarrow BQ$
$Q \rightarrow BB$
$B \rightarrow \dot{b}$

LHS of dotted RHS

$S \rightarrow PQ$
$P \rightarrow \dot{A}P \mid \dot{A}\dot{A}$
$\dot{A} \rightarrow \dot{a}$
$Q \rightarrow \dot{B}Q \mid \dot{B}\dot{B}$
$\dot{B} \rightarrow \dot{b}$

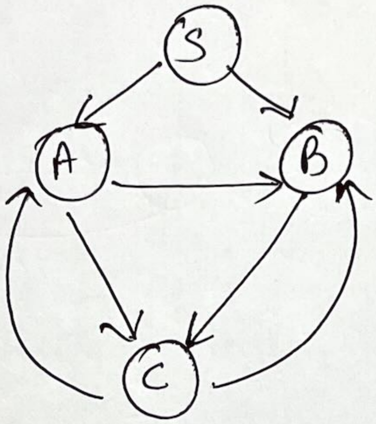
$S \rightarrow \dot{P}\dot{Q}$
$\dot{P} \rightarrow \dot{A}\dot{P} \mid \dot{A}\dot{A}$
$\dot{A} \rightarrow \dot{a}$
$\dot{Q} \rightarrow \dot{B}\dot{Q} \mid \dot{B}\dot{B}$
$\dot{B} \rightarrow \dot{b}$

② $S \to AB \mid a$
$A \to BC \mid B$
$B \to CC \mid c$
$C \to AB \mid d$

Check L(G) is infinite or not.

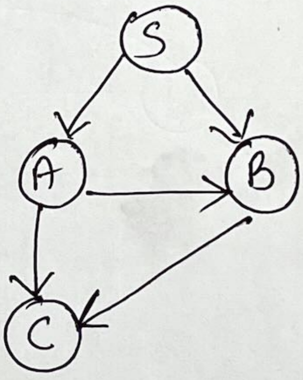
Step 1: No ∈ production, No useless production, No Unit

To check whether L(G) is infinite or not we draw a directed graph as follows:-



Since, there is cycle in the graph L(G) is infinite.

③ $S \to AB \mid a$
$A \to BC \mid b$
$B \to CC \mid c$
$C \to d$



Since, there is no cycle in L(G).
∴ L(G) is finite.

Q: $A = \{a^{n^2} \mid n \geq 1\}$ is not CFL.

Q: $L = \{0^i 1^j 2^i 3^j \mid i >= 1$ and $j >= 1\}$ is not CFL.