UNIT – II

—: Regular Languages: —

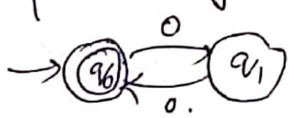
Regular expression: —

→ A symbolic representation of lang.
↓
set of strings.
↓
set of symbols.

→ The language for which we can define the regular expression is called a "Regular set"

(or)

→ Regular sets. are the sets which are accepted by Finite automata. Eg:- L={ε, 00, 0000, ---}



R.E Rules: —

1) ϕ is a regular expression which denotes the empty set. { }.
2) ε is a R.E and denotes the set {ε}. and it is a null string.
3) For each 'a' in Σ 'a' is a r.E and denotes the set {a}.
4) if r and s are r.Es denoting the languages L_1 and L_2 respectively then.
   a) r+s is equivalent to $L_1 \cup L_2$ i.e union.
   b) rs " " " L_1L_2 " Concatenation.

Scanned by CamScanner

c) r^* is equivalent to L_1^* i.e closure.

$r^* \rightarrow$ kleen closure.

$r^+ \rightarrow$ positive closure.

$$L^* = \bigcup_{i=0}^{\infty} L^i$$

$$L^+ = \bigcup_{i=1}^{\infty} L^i.$$

Eg:-

$\phi = \{ \ \}$

$\epsilon = \{\epsilon\}$

$R = a = \{a\} \rightarrow$ only 'a'

$R = a^* = \{\epsilon, a, aa, --\} \rightarrow$ Language accepting all combination of a's over the set $\Sigma = \{a\}$

$R = a^+ = \{a, aa, ---\} \rightarrow$ " except null string.

$R = (a+b)^* = \{\epsilon, a, b, aa, ab, ba, bb, ---\}$ all string having any no of a's and b's

1) $\Sigma = \{a, b\}$.

String length exactly 2	at least 2.	at most 2.
$L_1 = \{aa, ab, ba, bb\}$	$L_1 = \{aa, ab, ba, bb, aaa, -----\}$	$L_1 = \{\epsilon, a, b, aa, ab, ba, bb\}$
$aa + ab + ba + bb$	$(a+b)(a+b)(a+b)^*$	$\epsilon + a + b + aa + ab + ba + bb.$
$a(a+b) + b(a+b)$		
$(a+b)(a+b).$		$(a+b+\epsilon)(a+b+\epsilon)$

Scanned by CamScanner

(2) $\Sigma = \{a, b\}$

even length strings.

$L = \{\epsilon, aa, ab, ba, bb, \ldots\}$

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

odd length strings.

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

(3) length of string divisible by '3'. $\Sigma = \{a, b\}$

a) string of length '3': $\rightarrow ((a+b)(a+b)(a+b))^*$

b) $\cong 2 \bmod 3 . \rightarrow ((a+b)(a+b)(a+b))^*(a+b)(a+b)$

$\cong 1 \bmod 3 \rightarrow ((a+b)(a+b)(a+b))^*(a+b)$.

(4). $\Sigma \{a, b\}$.

→ no. of a's exactly '2'

$b^* a b^* a b^*$

→ a's atleast '2'.

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

→ a's at most '2'.

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

(4) → no of a's are even.

not possible w, this ~~is~~ ϵ. noo $\{b, bb, bbb, \epsilon\}$ added to b^* it is posible. ← $(b^* a b^* a b^*)^* + b^*$

(or)

$(b^* a b^* a)^* . b^*$

Scanned by CamScanner

→ set of all string start with 'a'.

$a(a+b)^*$

→ ends with 'a'

$(a+b)^* a$

→ Containing 'a'.

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

→ Starting and ending with different symbols.

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

→ Starting & ending with same symbol.

$L = \{\epsilon, a, b, aa, bb, aba, bab, \ldots\}$

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

5. $\Sigma = \{a, b\}$.

→. No two a's come together.

$L = \{\epsilon, b, bb, bbb, a, ab, aba, abab, ababa, ba,$
$bab, baba, babab, \ldots\}$

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

✓ $\boxed{(b+ab)^* (\epsilon + a)}$

(or)

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

✓ $\boxed{(a+\epsilon)(b+ba)^*}$

Scanned by CamScanner

(6) no of 2 a's & no of 2 b's should not come together.

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

	start with	end with
$\{a, aba, ababa, \ldots\} \longrightarrow$	a	$a \rightarrow (ab)^* a$ (or) $a(ba)^*$
$\{ab, abab, ababab, \ldots\} \longrightarrow$	a	$b \rightarrow (ab)^*$ (or) $a(ba)^* b$
$\{ba, baba, bababa, \ldots\} \longrightarrow$	b	$a \rightarrow (ba)^*$ (or) $b(ab)^* a$
$\{b, bab, babab, bababab, \ldots\} \longrightarrow$	b	$b \rightarrow (ba)^* b$ (or) $b(ab)^*$

$$\rightarrow (ab)^* a + (ab)^* + (ba)^* + (ba)^* b.$$

$$(ab)^* (a+\epsilon) + (ba)^* (\epsilon + b)$$

$$\boxed{(\epsilon+b)(ab)^*(a+\epsilon).}$$

(7). Strings of 0's & 1's with two consecutive 0's is represented as $\Sigma\{0,1\}$

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

(8) begins with 1 $\longrightarrow 1(0+1)^*$

(9). begins with 1. and not having two consecutive 0's.

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

(10). String ending in 011

$$(1+0)^* 011.$$

(11). any no of 0's followed by any no of 1's followed by any no of 2's.

$$0^* 1^* 2^*$$

Scanned by CamScanner

(12) . any no of 0's folloed by any no of 1's followed by any no of 2's folled by with at least one number each.

$00^* 11^* 22^*$

(13) ending in 00.

$(0+1)^* 00$. (or) $(1+0)^* 00$.

(14) . beginning with 'a' & ending with 'b'.

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

(15) . a's followed by one 'b' (or) b's folled by one 'a'.

$a^* b + b^* a$.

(16) write RE denote a language L which accepts all the strings which begin (or) end either 00 (or) 11.

$R = [(00+11)(0+1)^*] + [(0+1)^* (00+11)]$

(17) write RE to denote a language L over Σ^*, where $\Sigma = \{a, b\}$ such that the 3rd character from right end of the string is always a.

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

Scanned by CamScanner

Identity Rules:-

→ The two R.E P & Q are equivalent ($P=Q$) if and only if P represents the same set of strings as Q does.

→ Let P, Q, R are R.E.

1) $\epsilon R = R\epsilon = R$.
2) $\epsilon^* = \epsilon$ ϵ is null string.
3) $(\phi)^* = \epsilon$ ϕ is empty string.
4) $\phi R = R\phi = \phi$
5) $\phi + R = R$
6) $R + R = R$
7) $RR^* = R^* R = R^+$
8) $(R^*)^* = R^*$
9) $\epsilon + RR^* = R^*$
10) $(P+Q)R = PR + QR$.
11) $(P+Q)^* = (P^* Q^*)^* = (P^* + Q^*)^*$
12) $R^*(\epsilon + R) = (\epsilon + R)R^* = R^*$
13) $(R+\epsilon)^* = R^*$
14) $\epsilon + R^* = R^*$
15) $(PQ)^* P = P(QP)^*$
16) $R^* R + R = R^* R$.
17) $(P + PP)^* = P^*$

Scanned by CamScanner

① Prove $(1+00^*1)+(1+00^*1)(0+10^*1)^*(0+10^*1)$ ③

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

Soln

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

$(1+00^*1)(\varepsilon+\underbrace{(0+10^*1)^*}_{R^*}\underbrace{(0+10^*1)}_{R}) \quad (\varepsilon+R^*R=R^*)$

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

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

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

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

Hence two R.E are equivalent.

② Show that $(0^*1^*)^* = (0+1)^*$

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

② $(0+1+00)^* = (0+1)^*$

Soln RHS $= (0+1+00)^* = (\underbrace{0+00}_{P}+\underbrace{1}_{Q})^* \to (P+Q)^* = (P^*+Q^*)^*$

$= ((0+00)^*+1^*)^* \to (P+PP)^* = P^*$

$= (0^*+1^*)^* \Rightarrow (P+Q)^* = P^*Q^* = (P^*+Q^*)^*$

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

Scanned by CamScanner

(3) prove $\varepsilon + 1^*(011)^*(1^*(011)^*)^* = (1+011)^*$

sol. $L.H.S = \underset{\varepsilon +}{\varepsilon +} \underbrace{1^*(011)^*}_{R} \underbrace{(1^*(011)^*)^*}_{R^*}$ $\quad (\varepsilon + RR^* = R^*)$

$= (\underbrace{1^*}_{P} \underbrace{(011)^*}_{Q})^*$ $\quad (P+Q)^* = (P^*Q^*)^* = (P^*+Q^*)^*$

$= (1+011)^* = R.H.S$

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

sol. ~~L.H.S (a+b)*~~

$R.H.S = \underbrace{a^*}_{P} (\underbrace{ba^*}_{Q})^*$ $\quad (P^*Q^*)^* = (P^*+Q^*)^*$

$\quad (P^*+Q^*)^* = (P+Q)^*$

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

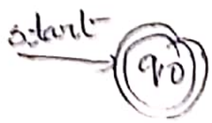
$= (a + ba^*)^*$

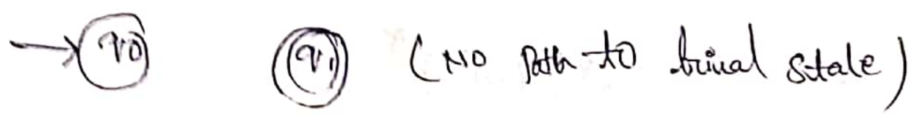
$= (a+b)^* = L.H.S$

Scanned by CamScanner

Constructing Finite Automata for given Regular expressions:→

* Let L be a R.E
→ r has zero operations, means r can either ε or φ (or) 'a' for some a in input set Σ.

$r = \epsilon$ →


$r = \phi$ →


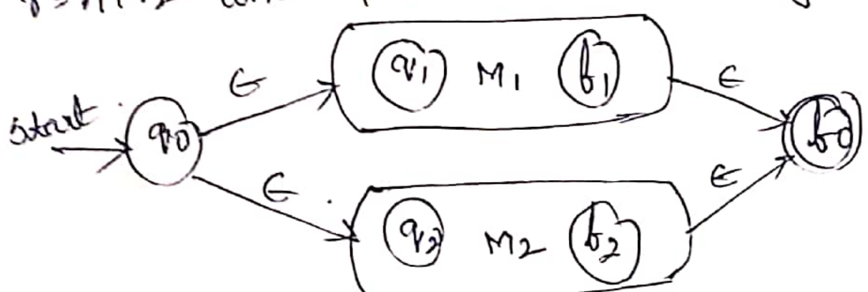
$r = a$ →


* → In any type of regular expressions there are only three cases possible.

1. Union 2. Concatenation 3. closure.

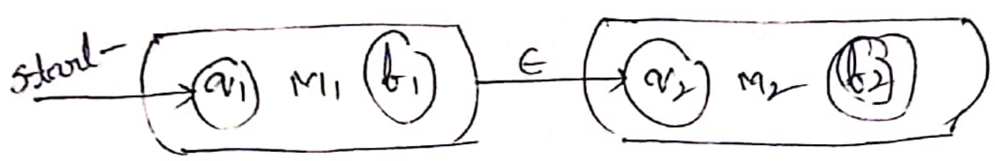
Case 1:— Union case.

Let $r = r_1 + r_2$ where r_1 and r_2 be the regular Expresions.



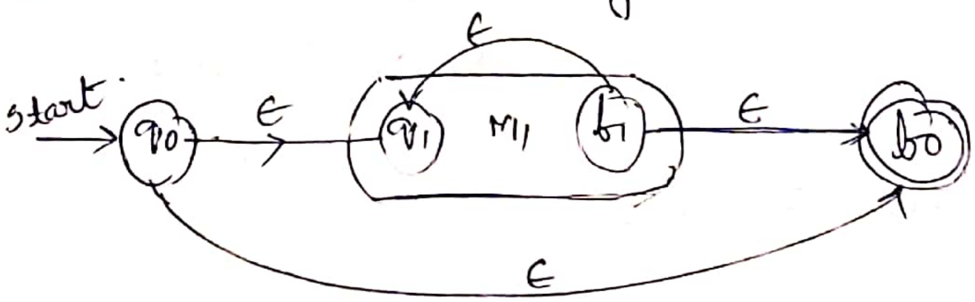
Case 2:— concatination case.

Let $r = r_1 r_2$ where r_1 & r_2 are two R.E's.



Case 3 :- closure case.

Let $r = r_1^*$ where r_1 be a regular expression.



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

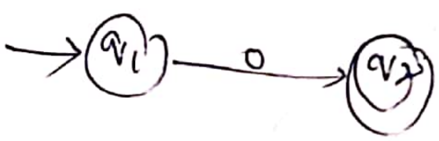
sol

$r_3 = r_1 + r_2$

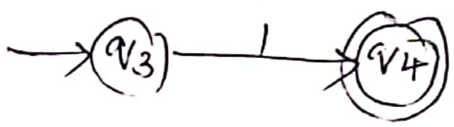
$r = r_3^*$

$r_1 = 0, \quad r_2 = 1$

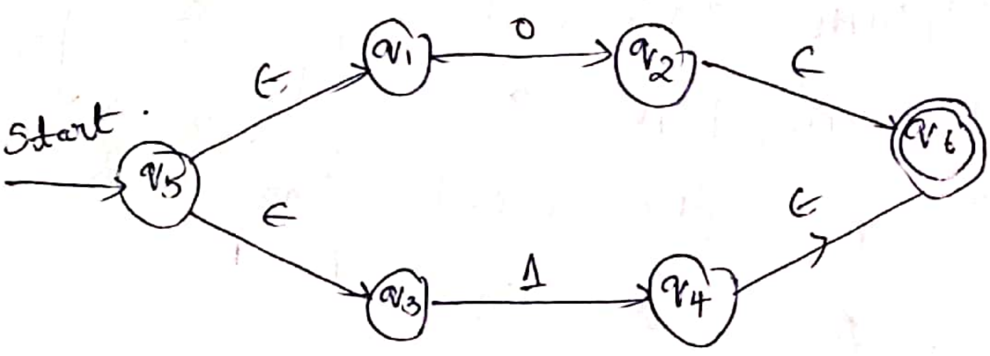
NFA for r_1 will be.



NFA for r_2 will be

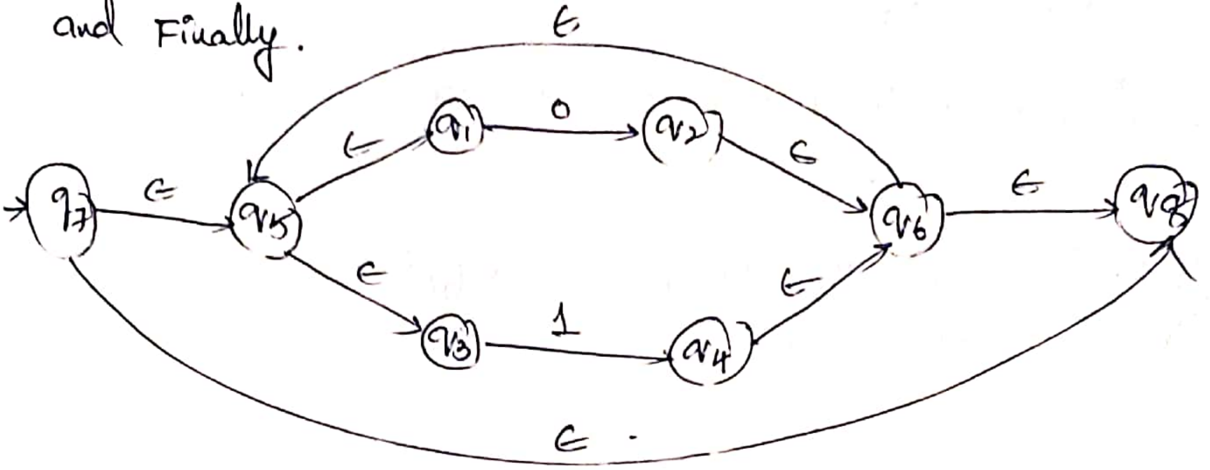


NFA for r3 will be.



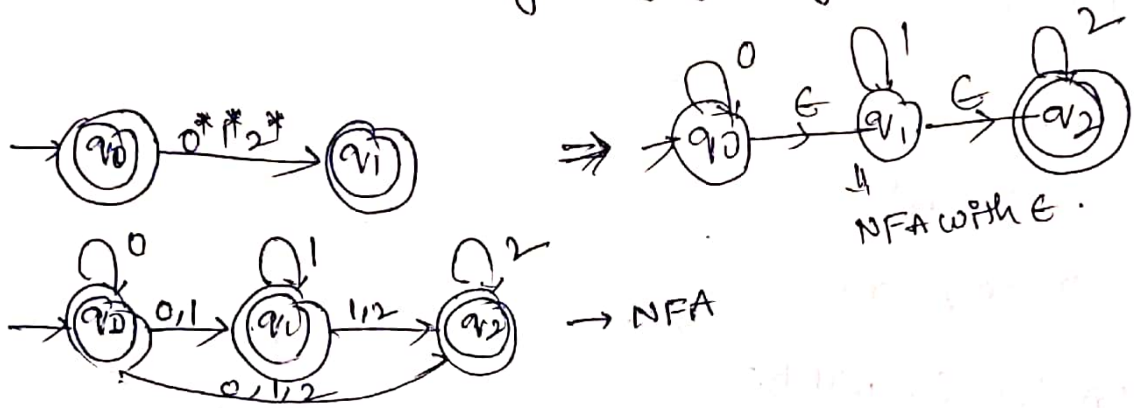
Scanned by CamScanner

and Finally.



② Construct a DFA accepting language by $0^* 1^* 2^*$

sol:

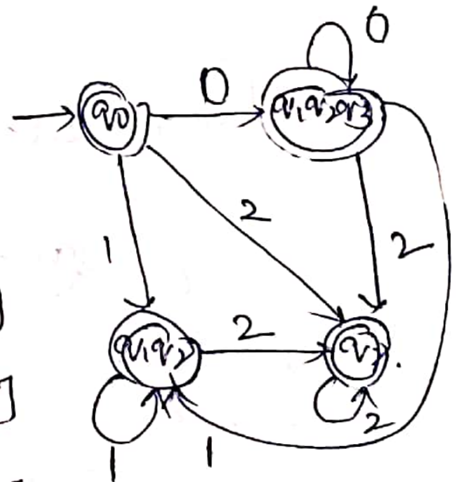


transition table.

	0	1	2
→ q_0	q_0,q_1,q_2	q_1,q_2	q_2
q_1	ϕ	q_1,q_2	q_2
q_2	ϕ	ϕ	q_2

convert NFA to DFA.

	0	1	2
→ $[q_0]$	$[q_0q_1q_2]$	$[q_1,q_2]$	$[q_2]$
* $[q_0q_1q_2]$	$[q_0q_1,q_2]$	$[q_1,q_2]$	$[q_2]$
* $[q_1q_2]$	ϕ	$[q_1,q_2]$	$[q_2]$
* $[q_2]$	ϕ	ϕ	$[q_2]$



Scanned by CamScanner

Q3 Construct NFA for the following Regular expression.

a) $0+10^*+01^*0$

b) $(0+1)^*(01+110)$

c) $((00^*(11))\cup 01)^*$

d) 01^*+1

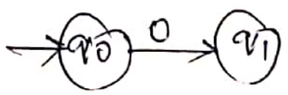
e) $01[((10)^*+(11)^*+0]^*\,1$

f) $[(0+1)(0+1)]^*+[(0+1)(0+1)(0+1)]^*$

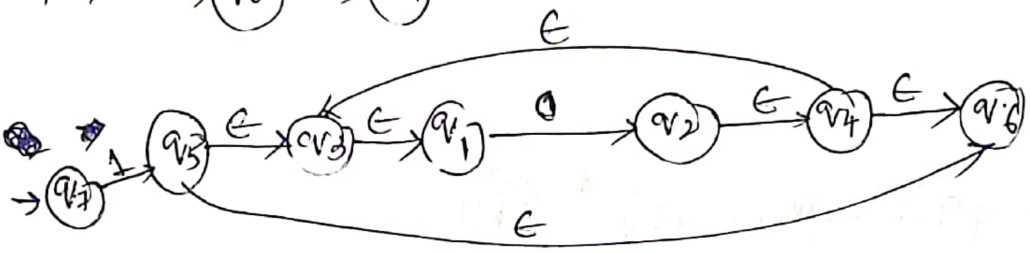
Sol:

$$r.e = \underset{r_1}{0} + \underset{r_2}{10^*} + \underset{r_3}{01^*0}.$$

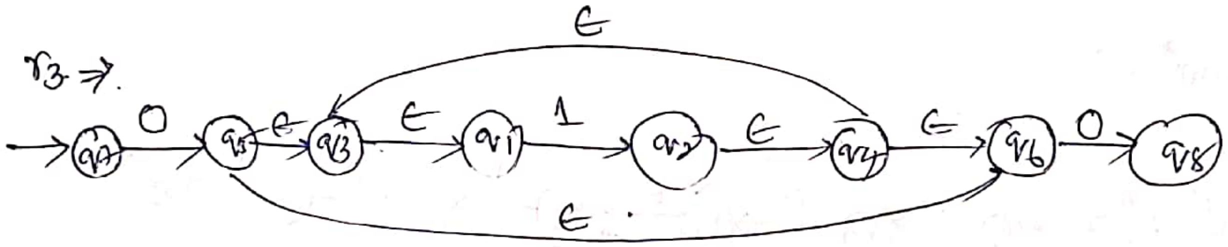
$r_1 \Rightarrow$



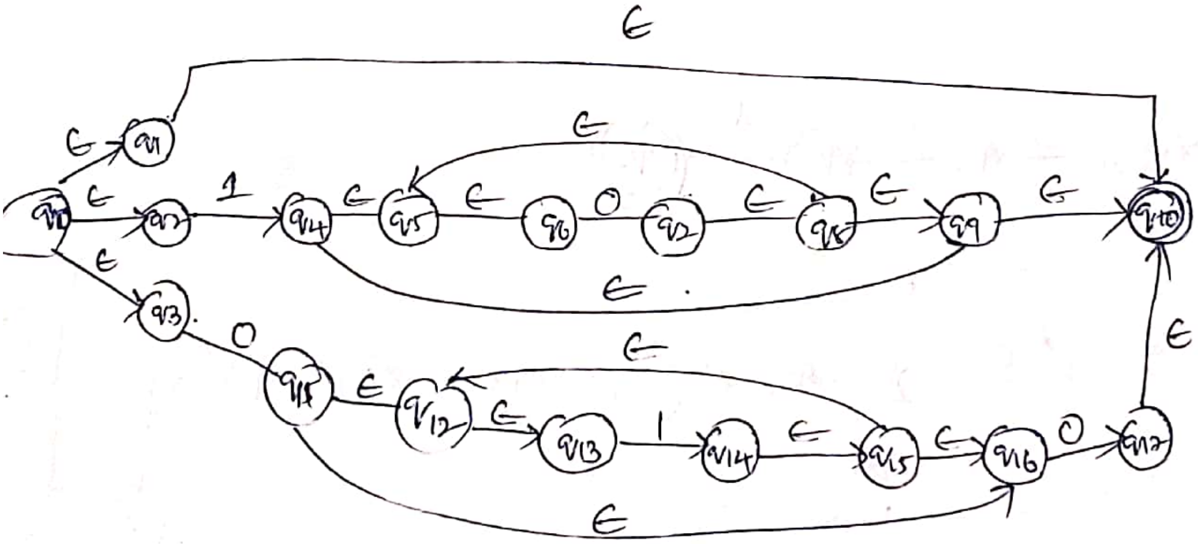
$r_2 \Rightarrow$



$r_3 \Rightarrow$.



$r.e = 0+10^*+01^*0.$

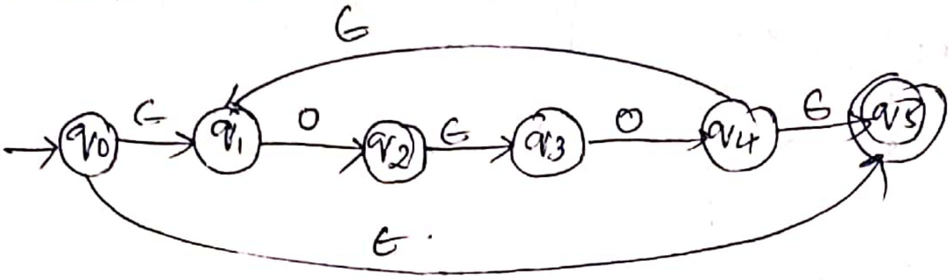


Scanned by CamScanner

(c)

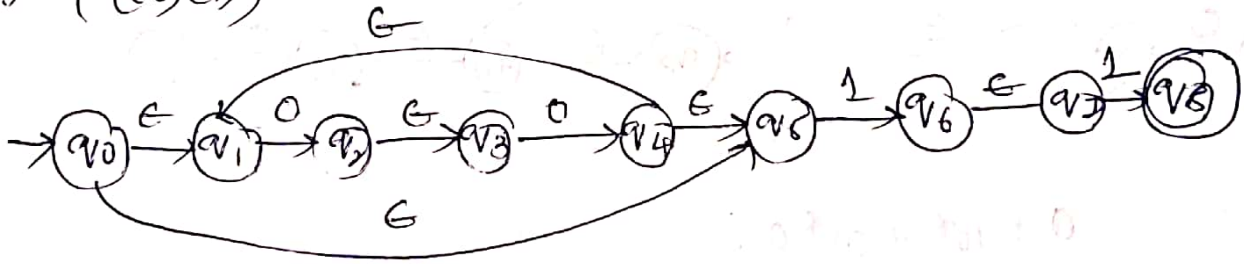
$$\Big(\underbrace{\big(\underbrace{(00)^*}_{r_1}\underbrace{(11)}_{r_2}\big)}_{r_3} \cup \underbrace{01}_{r_4} \Big)^*$$

$r_1 = (00)^*$

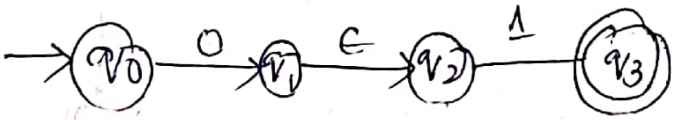


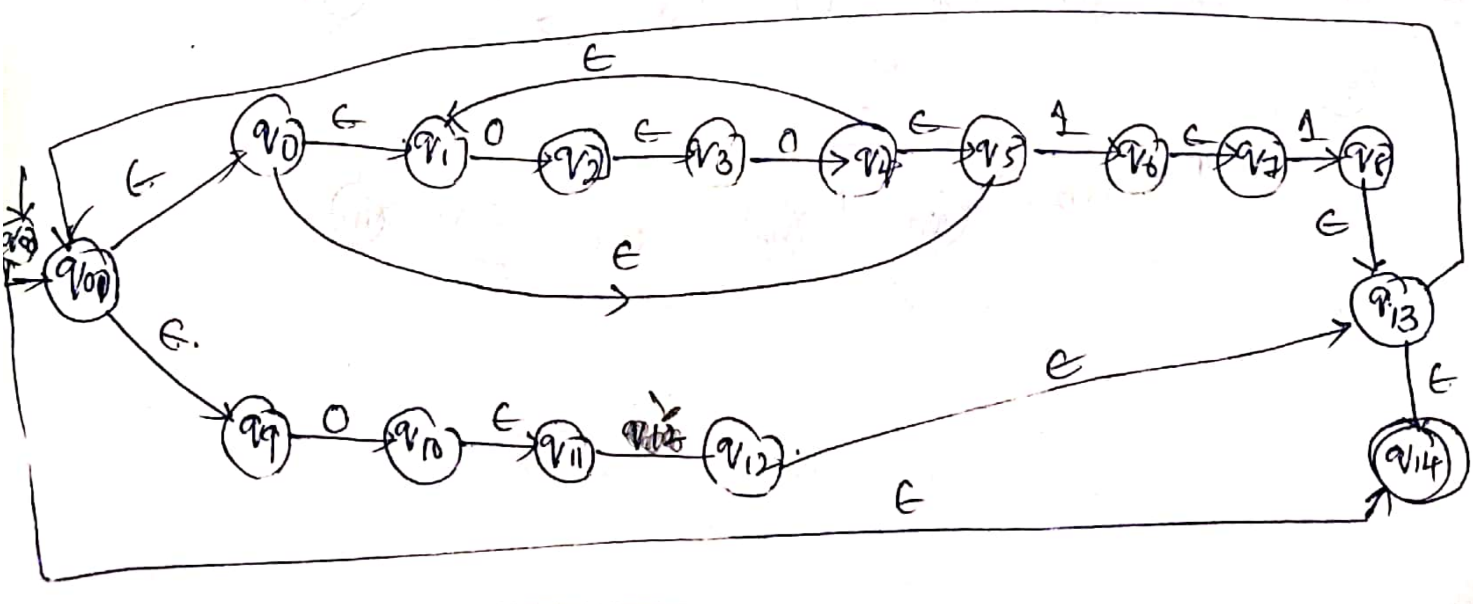
$r_2 = 11$

$r_3 = ((00)^*(11))$



$r_4 = 01$

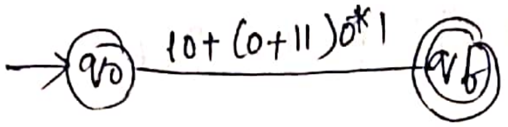


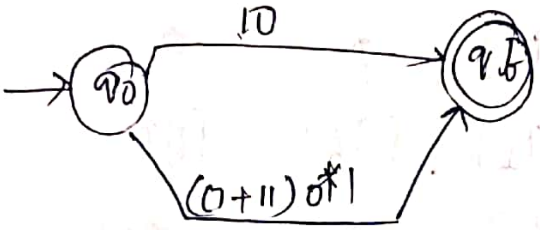


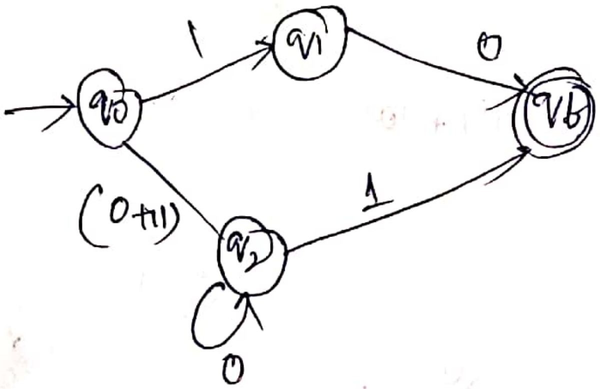
Scanned by CamScanner

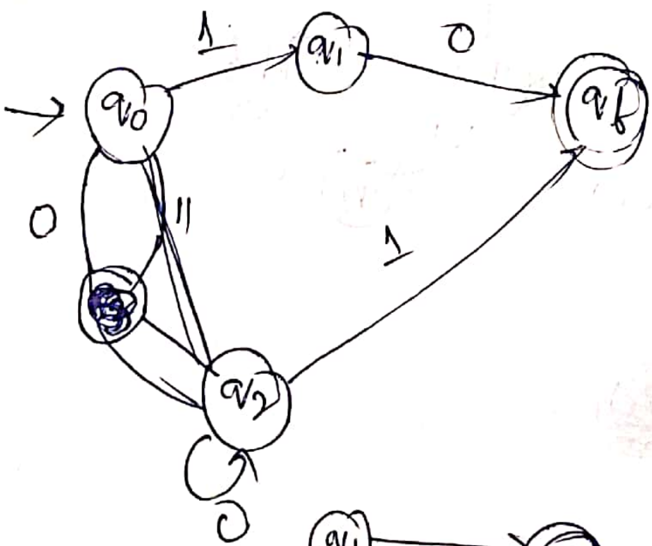
Direct Method for conversion of r.e to FA:->

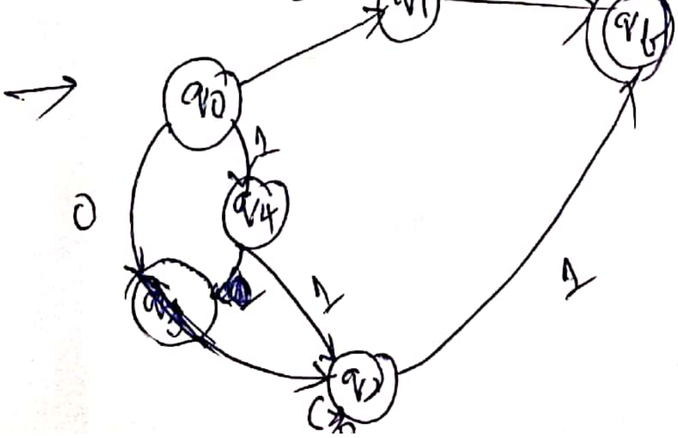
(1) Design a FA from given R.E $10+(0+11)0^*1$











Scanned by CamScanner

NFA

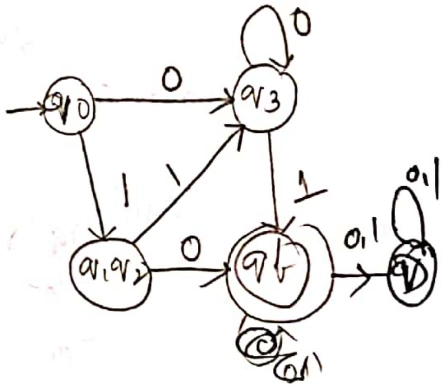
	0	1
→ q_0	q_3	$\{q_1, q_2\}$
q_1	q_f	ϕ
q_2	ϕ	q_3
q_3	q_3	q_f
q_f	ϕ	ϕ

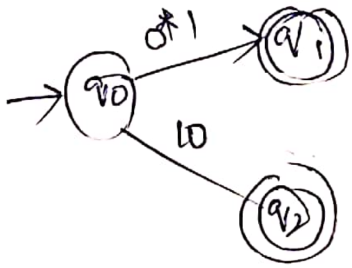
DFA

	0	1
→ $[q_0]$	$[q_3]$	$[q_1, q_2]$
$[q_3]$	$[q_3]$	$[q_f]$
$[q_1, q_2]$	$[q_f]$	$[q_3]$
$[q_f]$	ϕ	ϕ
~~$[q_f]$~~	~~$[q_f]$~~	~~$[\phi]$~~
~~$[q_2]$~~	~~ϕ~~	~~$[q_3]$~~

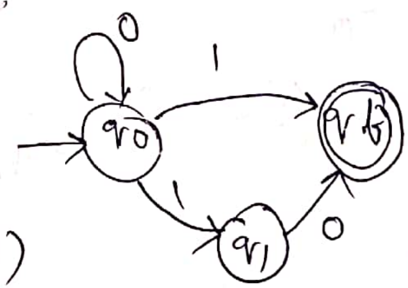
(3) const

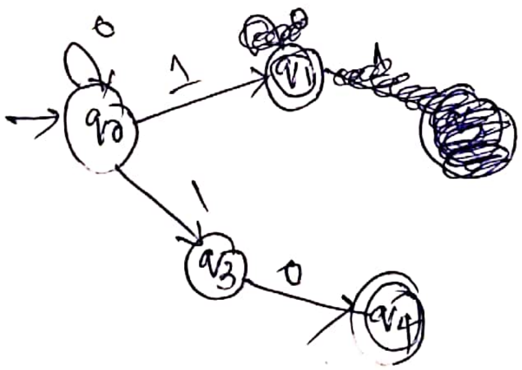
(2) Construct FA for R.G $0^*1 + 10$.





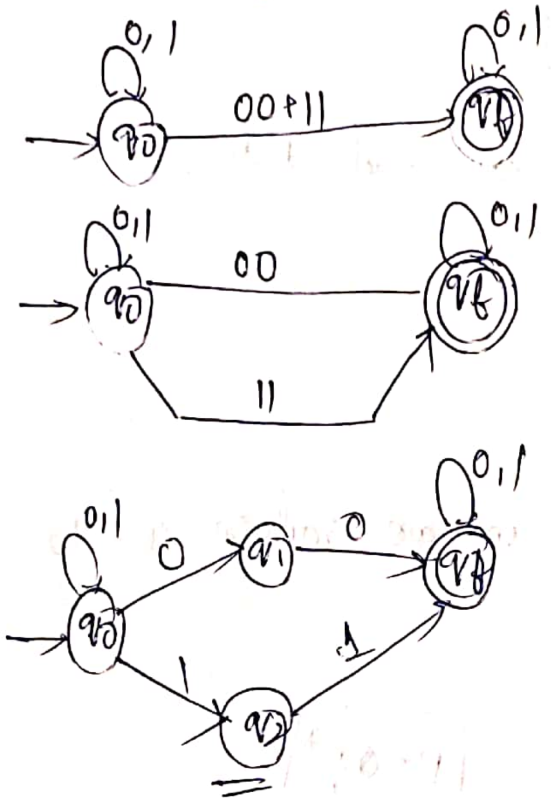
(or)





(3) Construct FA to accept the regular expression.

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



Arden's Theorem:—

→ Let P and Q be the two R.E over the input set Σ.

The regular expression R is given as

$$R = Q + RP$$

which has a unique solution as

$$\boxed{R = QP^*}$$

Scanned by CamScanner

1) construct r.e for the given DFA

soln - Let us build the R.E for each state.

$$q_1 = q_1 0 + \epsilon$$
$$q_2 = q_1 1 + q_2 1.$$
$$q_3 = q_2 0 + q_3 (0+1)$$

$\therefore$ final states are q_1, q_2. we are solving q_1 & q_2 only.

First. $q_1 = q_1 0 + \epsilon$

$$\underbrace{q_1}_{R} = \underbrace{\epsilon}_{Q} + \underbrace{q_1}_{R}\underbrace{0}_{P} \qquad \boxed{R = QP^*}$$

$$q_1 = \epsilon . 0^*$$
$$\boxed{q_1 = 0^*} \qquad \epsilon . R = R.$$

Second

$$q_2 = q_1 1 + q_2 1$$
$$\underbrace{q_2}_{R} = \underbrace{0^* 1}_{Q} + \underbrace{q_2}_{R}\underbrace{1}_{P}$$
$$\boxed{q_2 = 0^* 1 1^*}$$

r.e $r = q_1 + q_2$

$$= 0^* + 0^* 1 1^*$$
$$\boxed{r = 0^* + 0^* 1^+} \qquad \boxed{1 . 1^* = 1^+}$$

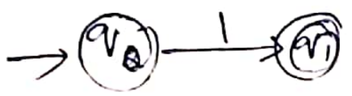
→ construct NFA for 01^*+1

Soln

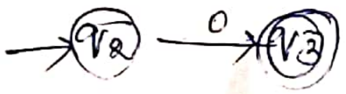
$r = 01^*+1$

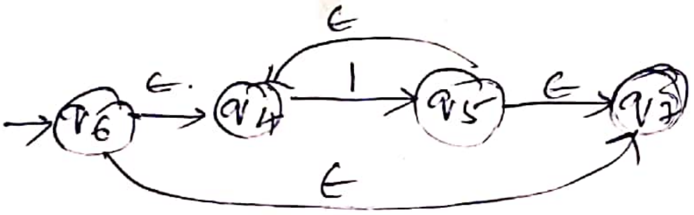
$r_1 = 01^*$ $r_2 = 1$

→ $r_2 = 1$



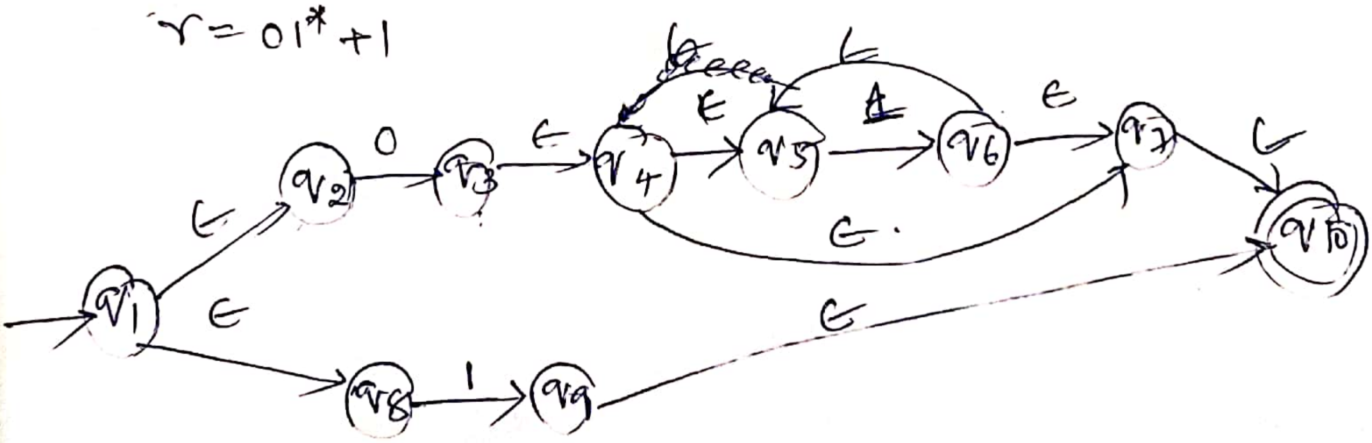
→ $r_1 = 01^*$ $r_3 = 0$, $r_4 = 1^*$





$r_1 = 01^*$

$r = 01^*+1$



Scanned by CamScanner

(2) Find Regular Expression for the following NFA's.

(a).

sol,

$q_0 = \epsilon$

$q_1 = q_0(0+1) + q_1 0$

$q_2 = q_1(0+1) + q_2 1.$

q_1 is only the Final State. sol q_1 only.

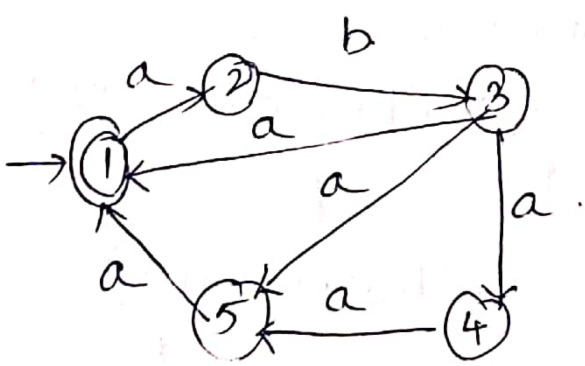
$q_1 = \underline{q_0}_{\epsilon}(0+1) + q_1 0$

$q_1 = \epsilon(0+1) + q_1 0.$

$\underbrace{q_1}_{R} = \underbrace{(0+1)}_{Q} + \underbrace{q_1}_{R} \underbrace{0}_{P}.$ $\boxed{R = QP^*}$

$\boxed{q_1 = (0+1)0^*}$ $\boxed{\therefore\ r_1 = (0+1)0^*}$

(3)



$1 = \epsilon + 3a + 5a$
$2 = 1a$
$3 = 2b$
$4 = 3a$
$5 = 3a + 4a.$

sol only 1 state.

$5 = 3a + 3aa$
$= 3a(\epsilon + a)$
$= 2ba(a + \epsilon)$
$= 1aba(a + \epsilon)$

$3 = 2b$
$= 1ab.$

$1 = \epsilon + 3a + 5a.$
$= \epsilon + 1aba + 1aba(a+\epsilon)a$
$= 1aba(\epsilon + aa + a) + \epsilon.$

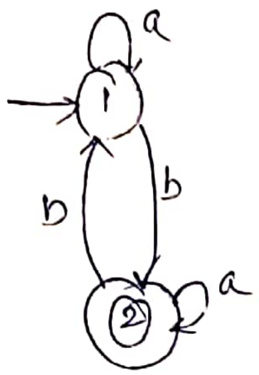
$\frac{1}{R} = \frac{1}{R}\underbrace{aba(\epsilon + aa + a)}_{P} + \frac{\epsilon}{Q}$ $\Rightarrow 1 = \epsilon[aba(\epsilon + aa + a)]^*$

$\boxed{r.e = [aba(\epsilon + aa + a)]^*}$

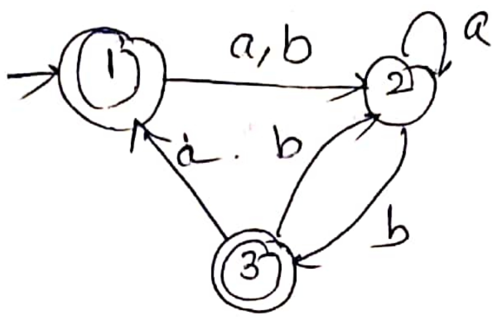
Scanned by CamScanner

4) convert the following F.A to R.E.

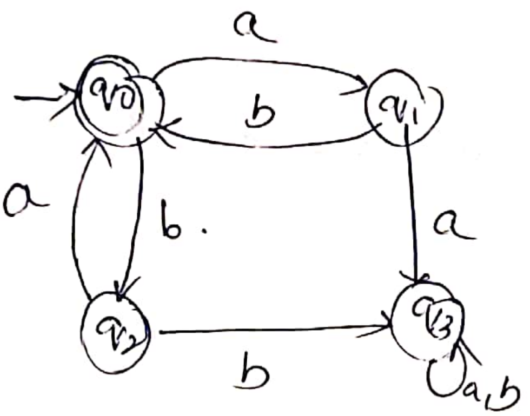
(a)



(b)



(c)



(a).

$1 = 1a + 2b$

$2 = 2a + 1b.$

$\underset{R}{1} = \underset{Q}{2b} + \underset{R}{1}\underset{P}{a}$

$\boxed{1 = 2ba^*}$

$2 = 2a + 2ba^*b.$

$2 = 2(a + ba^*b) + \epsilon$

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

$\boxed{r = (a + ba^*b)^*}$

(b).

$1 = 3a$

$2 = 1(a+b) + 3b + 2a \Rightarrow 2 = (1(a+b) + 3b)a^*$

$2 = (3a(a+b) + 3b)a^*$

$2 = 3(aaa^* + aba^* + ba^*)$

$3 = 2b.$

$\cancel{3 = (1(a+b) + 3b + 2a)b}$

$\cancel{3a(a+b) + 3b +}$

$\underset{R}{3} = \underset{R}{3}(\underbrace{aaa^*b + aba^*b + ba^*b}_{P}) + \underset{+Q}{\epsilon}$

$3 = \epsilon(a^*b(aa + ab + b))^*$

$\boxed{r = (a^*b(aa + ab + b))^*}$

(c).

$q_0 = q_2a + q_1b.$

$q_1 = q_0a$

$q_2 = q_0b$

$q_3 = q_1a + q_3(a+b) + q_2b.$

$q_0 = q_0ba + q_0a.$

$\underset{R}{q_0} = \underset{R}{q_0}(\underset{P}{ba+a}) + \underset{+Q}{\epsilon}.$

$q_0 = \epsilon(ba+a)^* \Rightarrow \boxed{r = (ab+a)^*}$

closure properties: —

(1) The union of two regular languages is regular.
if L_1 and L_2 are two languages then $L_1 \cup L_2$ is regular.

(2). The complement of regular language is regular.
if L_1 be R.L then $\overline{L_1}$ is also regular.

(3) The intersection of two languages is regular.
if L_1 and L_2 are two regular languages then $L_1 \cap L_2$ is regular.

(4) The difference of two regular languages is regular.
if L_1 and L_2 are two R.L then $L_1 - L_2$ is regular.

(5) The reversal of a RL is regular.

(6) The closure operation on a regular languages is regular.
if L_1 is regular then $L_1 = L(R_1^*)$

(7) The concatenation of R.L is regular.
if L_1 & L_2 are two languages then $L_1 \cdot L_2$ is regular.

Scanned by CamScanner

(8) A homomorphism of regular language is regular.

→ The term homomorphism means substitution of string by some other symbols.

Eg. String aabb → written 0011 under homomorphism.

a is replaced by 0

b " " " 1.

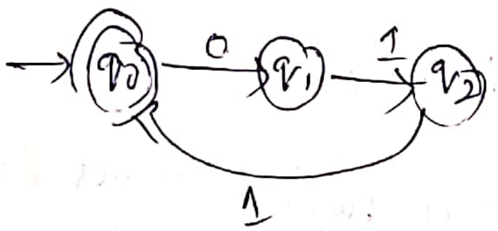
(9) The inverse homomorphism of Regular language is regular.

Eg. $L = (010)^*$ be a R.L. and $L = \{010, 010010, ---\}$

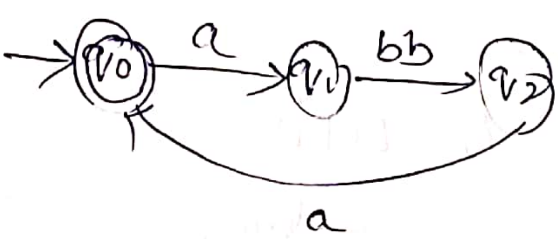
Let $h(0) = a$, $h(1) = bb$. be homomorphism fun

$h(L) = (abba)^*$

$L \Rightarrow$



$h(L) \Rightarrow$



$h^{-1}(L) =$

Grammar Formalism

Regular Grammar :→

A regular grammar is defined as

$G = (V, T, P, S)$ where.

$V \rightarrow$ set of symbols called non-terminals which are used to define the rules.

$T \rightarrow$ set of symbols called terminals.

$P \rightarrow$ set of production rules.

$S \rightarrow$ is a start symbol. which $\in V$.

The production rules P are of the form.

$$\left.\begin{array}{l} A \rightarrow aB \\ A \rightarrow a \end{array}\right\} \quad \begin{array}{l} A, B \rightarrow \text{non terminals} \\ a \rightarrow \text{terminal.} \end{array}$$

→ consider $G = \{V, T, P, S\}$

$V = \{S, A\}$

$T = \{0, 1\}$

$S \rightarrow$ start symbol. Then P.

$S \rightarrow aS$

$S \rightarrow 1B$

$B \rightarrow \epsilon$.

This is called Regular language and it can be represented by some DFA.

Scanned by CamScanner

Construction of Regular Grammar from Regular expression:—

· Convert R.E into it's equivalent R.G by using following method.

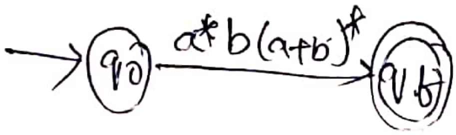
(1) construct a NFA with ϵ from it it equivalent DFA.

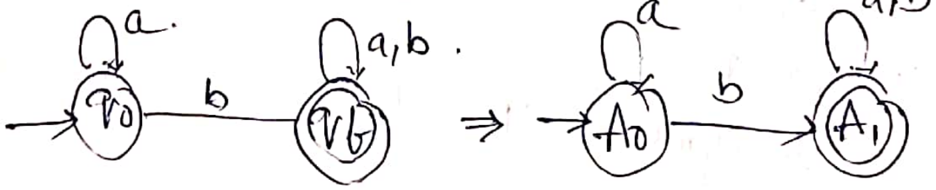
(2) Eliminate ϵ transitions and convert it to equivalent DFA.

(3). From constructed DFA, the corresponding states become nonterminal symbols and transitions made are equivalent to production rules.

Eg construct a R.g for the regular expression $a^*b(a+b)^*$.

sol. construct DFA for the given R.E.





$G = \{V, T, P, S\}$

$V = \{A_0, A_1\}$

$T = \{a, b\}$

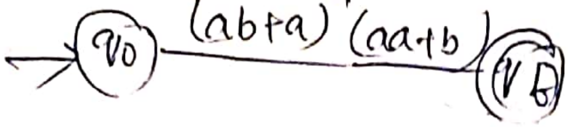
$A_0 \rightarrow S$.

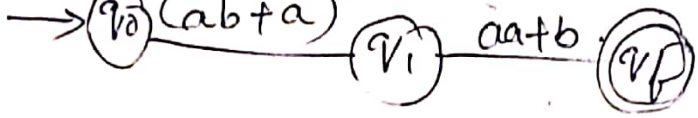
$A_0 \rightarrow aA_0$	$A_1 \rightarrow aA_1$
$A_0 \rightarrow bA_1$	$A_1 \rightarrow bA_1$
$A_0 \rightarrow b$.	$A_1 \rightarrow a$
	$A_1 \rightarrow b$.

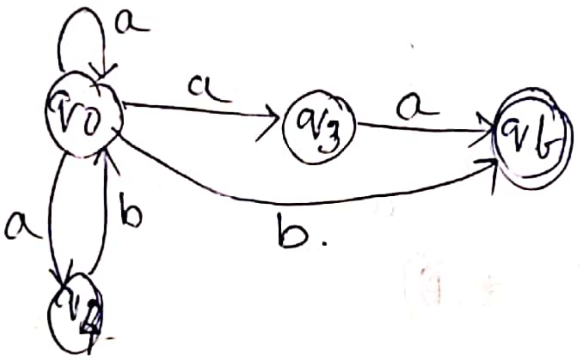
Scanned by CamScanner

② construct a R.G. for $(ab+a)^*(aa+b)$

Sol.

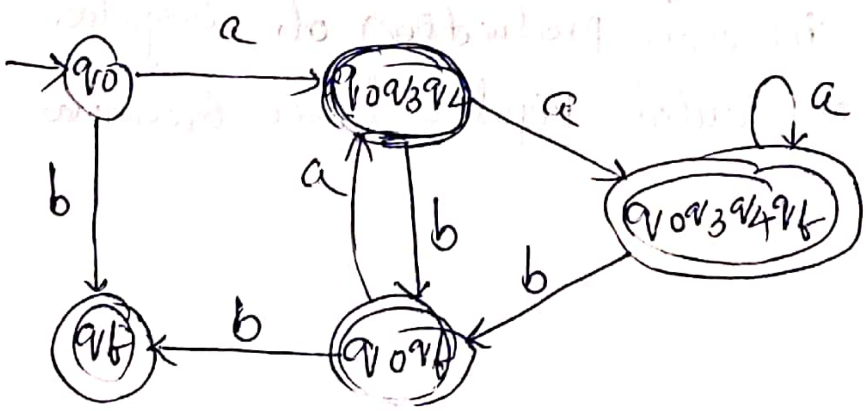






	a	b
→ q_0	$[q_0, q_3, q_4]$	q_f
q_3	q_f	ϕ
q_4	ϕ	q_0
q_f	ϕ	ϕ

	a	b
→ q_0	$[q_0q_3q_4]$	$[q_f]$
$[q_0q_3q_4]$	$[q_0q_3q_4q_f]$	$[q_0q_f]$
$[q_f]$	ϕ	ϕ
$[q_0q_3q_4q_f]$	$[q_0q_3q_4q_f]$	$[q_0q_f]$
$[q_0q_f]$	$[q_0q_3q_4]$	$[q_f]$



Now rename the states

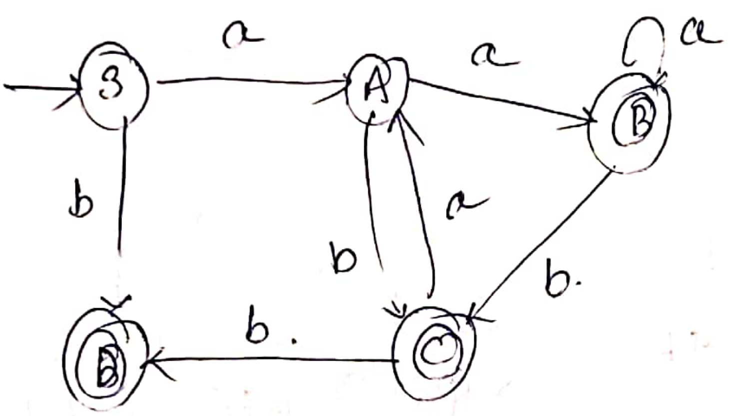
$[q_0] = S$

$[q_0 q_3 q_4] = A$

$[q_0 q_3 q_4 q_6] = B$

$[q_0 q_6] = C$

$[q_6] = D$.



$S \to aA$	$A \to aB$	$B \to aB$	$C \to aA$
$S \to bD$	$A \to bC$	$B \to bC$	$C \to bD$
$S \to b$.	$A \to a$	$B \to a$	$C \to b$.
	$A \to b$.	$B \to b$.	$D \to \epsilon$.

Right-linear and left-linear Grammar :—

→ if the non-terminal symbols appears as a rightmost symbol in each production of regular grammar then it is called Right-linear Grammar.

$A \to aB$

$A \to a$

$A \to \epsilon$.

Scanned by CamScanner

→ if the non-terminal symbol appears as a left most symbol in each production of a regular grammar then it is called left linear grammar.

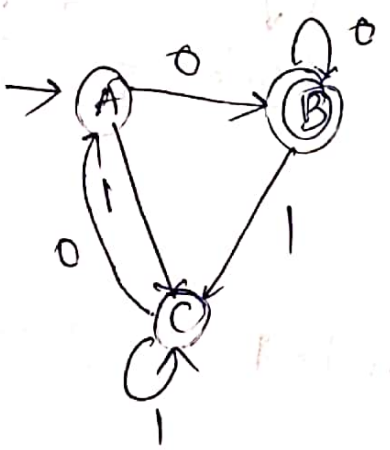
$A \rightarrow Ba$
$A \rightarrow a$
$A \rightarrow \epsilon$

* → The language is called regular if it is accepted by either left linear or right linear grammar.

Eg Construct right linear grammar for the following DFA.

	0	1
→ A	B	C
(B)	B	C
C	A	C

$A \rightarrow 0B \mid 1C \mid 0$
$B \rightarrow 0B \mid 1C \mid 0$
$C \rightarrow 0A \mid 1C$

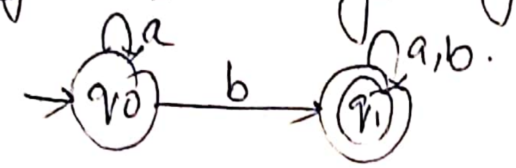


Scanned by CamScanner

Construction of FA to Regular Grammar :-

③

Eg construct regular grammar for given DFA



Sol:> $G = \{V, T, P, S\}$

$V = \{A_0, A_1\}$

$T = \{a, b\}$

$S = q_0$ ~~a_0~~.

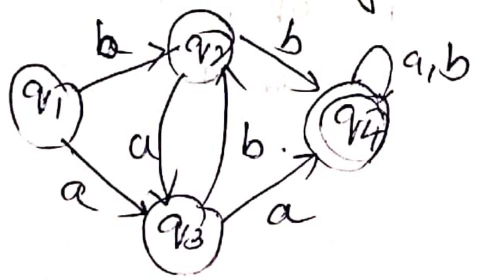
$A_0 \rightarrow aA_0$ $\quad$ $A_1 \rightarrow aA_1 \mid a$.

$A_0 \rightarrow bA_1$ $\quad$ $A_1 \rightarrow bA_1 \mid b$.

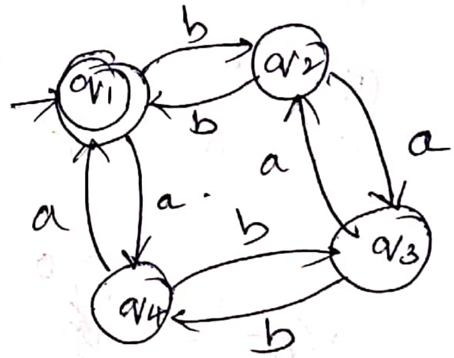
$A_0 \rightarrow b$

② construct a regular grammar for given FA.

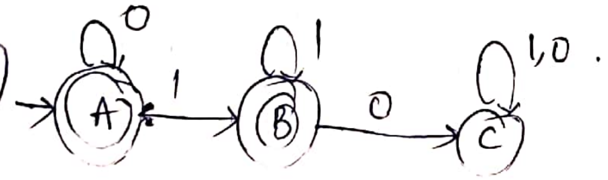
(a)



(c)



(b)



$A \rightarrow 0A \mid 0$

~~A~~ $\rightarrow 1B$

$B \rightarrow 0C \mid 1$

$B \rightarrow 1B$

$C \rightarrow 1C$

$C \rightarrow 0C$.

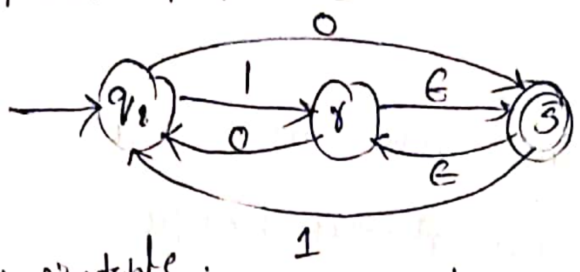
$S \rightarrow A \mid \epsilon$.

Here initial state is also final state.

we add new state i.e.

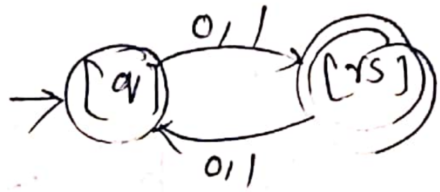
$\boxed{S \rightarrow A \mid \epsilon}$.

③ Find R.G for the shown F.A.



→ Final transition table.

	0	1
→ [q]	[rs]	[rs]
[rs]	[q]	[q]



$\epsilon\text{-closure}(q) = \{q\}$

$\quad " \quad (r) = [r,s] \rightarrow [rs]$ state.

$\quad " \quad (s) = [r,s] = [rs]$

$\delta'(q,0) = \epsilon\text{-closure}(\delta(q,0))$

$\quad = \quad " \quad (s)$

$\quad = [rs]$

$\delta'(q,1) = [rs]$

$\delta'([rs],0) = [q]$

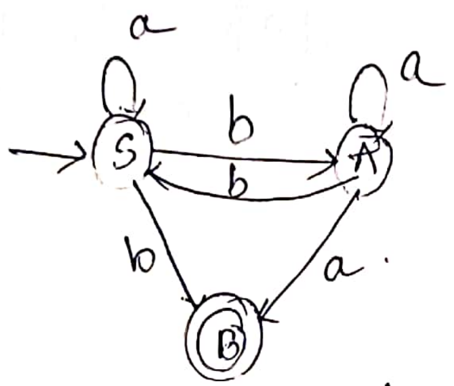
$\delta'([rs],1) = [q]$

$\cancel{\delta'(s,0) =}$

$\cancel{\delta'(s,1)}$

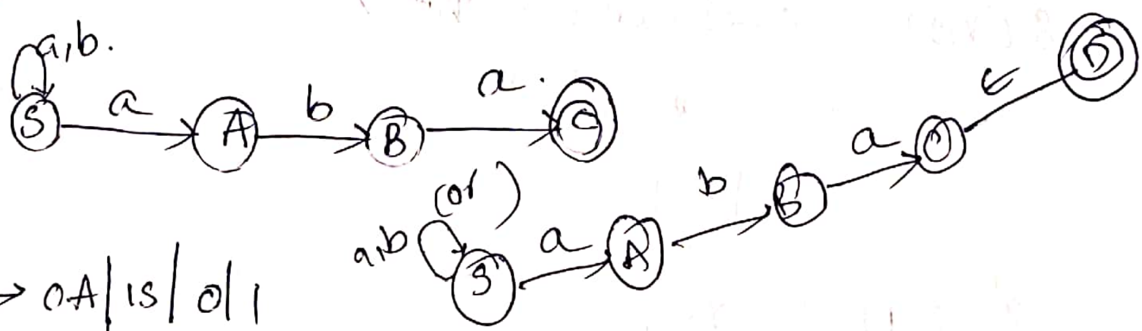
Construction of R.G to FA:—

(1) Construct FA recognizing, where G is a grammar $S \to aS|bA|b$ and $A \to aA|bS|a$.

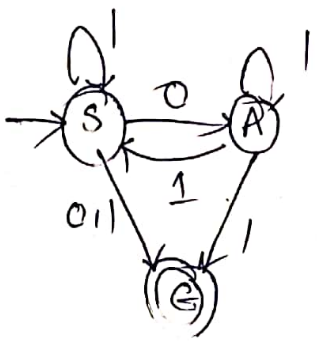


(2) Design DFA equivalent to the grammar

$S \to aS|bS|aA$
$A \to bB$
$B \to aC$
$C \to \epsilon$.



(3) $S \to 0A|1S|0|1$
$A \to 1A|1S|1$



Scanned by CamScanner

construction of R.G to F.A :—

Rule 1:→ Include the edges $q_i \to q_j$ with an edge lable 'a'. if the production $A_i \to aA_j$ is present in grammar

i.e, we include $\delta(q_i, a) = q_j$ if $A_i \to aA_j$ (exists)

Rule 2: Include the edge $q_k \xrightarrow{a} q_f$. if $\delta(q_k, a) = q_f$

if $A_k \to a$ (exists).

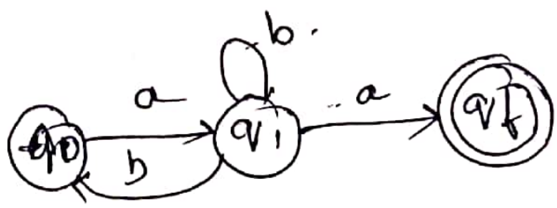
<u>Note</u>:- q_f is a final state.

Examples:—

① $A_0 \to aA_1$
$A_1 \to bA_1$
$A_1 \to a$
$A_1 \to bA_0$

<u>Sol</u>:

$A_0 \to aA_1$	$\delta(A_0, a) = A_1$	(Rule 1)	$\to q_1$
$A_1 \to bA_1$	$\delta(A_1, b) = A_1$	(Rule 1)	$\to q_1$
$A_1 \to a$	$\delta(A_1, a) = A_f$	(Rule 2)	$\to q_f$
$A_1 \to bA_0$	$\delta(A_1, b) = A_0$	(Rule 1)	$\to q_0$



$M = \{(q_0, q_1, q_f), \{a, b\}, \delta, q_0, q_f\}$

$A_0 \to aA_1$
$\Rightarrow aa$

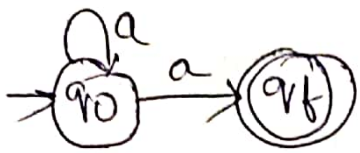
$A_0 \Rightarrow aA_1$
$\Rightarrow abA_1$
$\Rightarrow aba$

Scanned by CamScanner

6

2) $S \rightarrow aS \mid a$.

$q_0 \quad q_0$

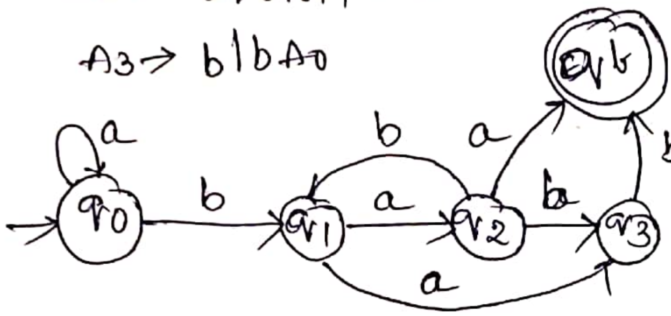


3) $A_0 \rightarrow aA_0 \mid bA_1$

$A_1 \rightarrow aA_2 \mid aA_3$

$A_2 \rightarrow a \mid bA_1 \mid bA_3$

$A_3 \rightarrow b \mid bA_0$



$A_0 \rightarrow aA_0 \Rightarrow \delta(q_0, a) = q_0$

$A_0 \rightarrow bA_1 \Rightarrow \delta(q_0, b) = q_1$

$A_1 \rightarrow aA_2 \Rightarrow \delta(q_1, a) = q_2$

$\vdots$

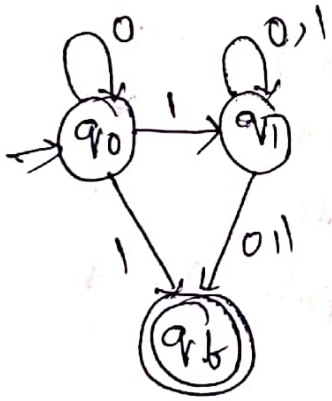
$A_2 \rightarrow a \Rightarrow \delta(q_2, a) = q_f$.

$\vdots$

$A_3 \rightarrow a \Rightarrow \delta(q_3, a) = q_f$

4) $q_0 \leftarrow S \rightarrow 0S \mid 1A \mid 1$

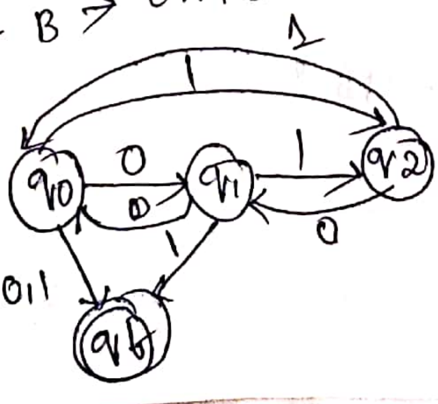
$q_1 \leftarrow A \rightarrow 0A \mid 1A \mid 0 \mid 1$



5) $q_0 \leftarrow S \rightarrow 0A \mid 1B \mid 0 \mid 1$

$q_1 \leftarrow A \rightarrow 0S \mid 1B \mid 1$

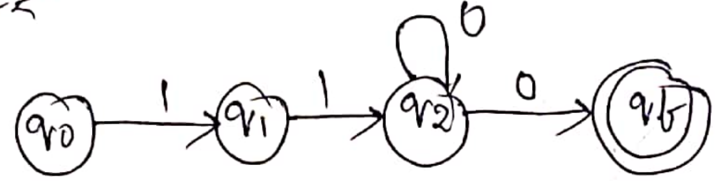
$q_2 \leftarrow B \rightarrow 0A \mid 1S$



Scanned by CamScanner

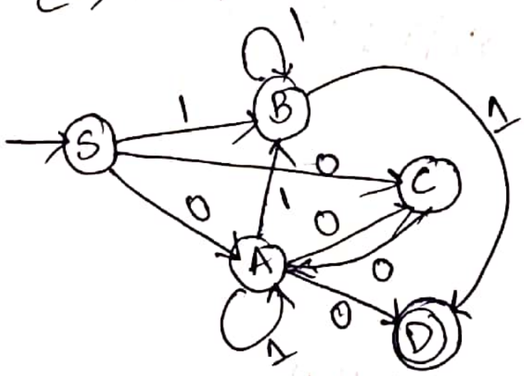
(6) construct the given left linear grammar to the Finite automata.

$q_0 \leftarrow S \to A1$
$q_1 \leftarrow A \to B1$
$q_2 \leftarrow B \to B0|0$

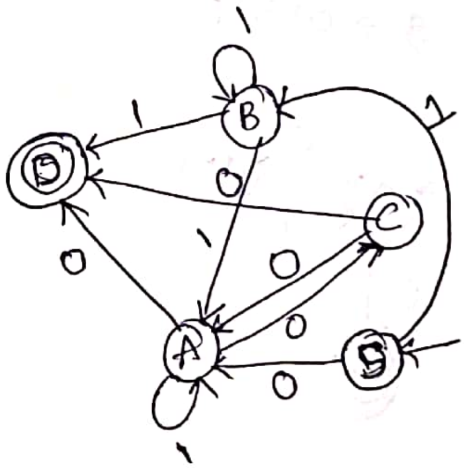


(7) convert R.L.G from the given L.L.G.

$S \to B1|A0|C0$
$A \to C0|A1|B1|0$
$B \to B1|1$
$C \to A0$



$\Leftrightarrow$



$S \to 0A|1B$
$A \to 1A|0D|0C|0$
$B \to 1B|1D|1A|1$
$C \to 0A|0D|0$

Scanned by CamScanner

(8) Construct F.A given grammar

$S \to aA \mid bB$

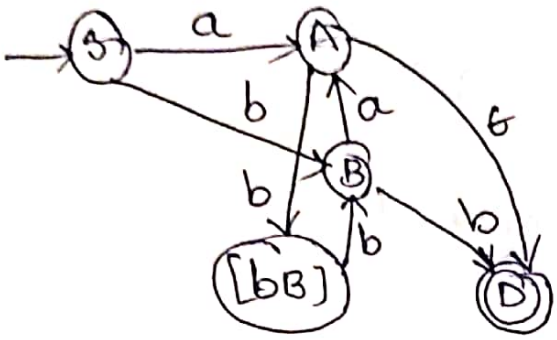
$B \to aA \mid b$

$A \to bbB \mid \epsilon$

$A \to bbB$

$\delta(A, b) = [bB]$

$\delta([bB], b) = B$

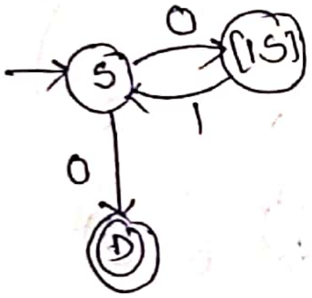


(9) $S \to 01S \mid 0$

$S \to 01S$

$\delta(S, 0) = [1S]$

$\delta([1S], 1) = S$



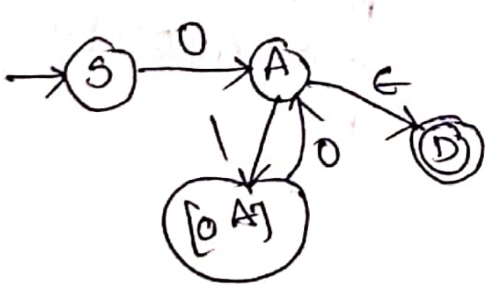
(10) $S \to 0A$

$A \to 10A \mid \epsilon$

$A \to 10A$

$\delta(A, 1) = [0A]$

$\delta([0A], 0) = A$



Scanned by CamScanner

Conversion of Right-linear Grammar to left-linear Grammar:→

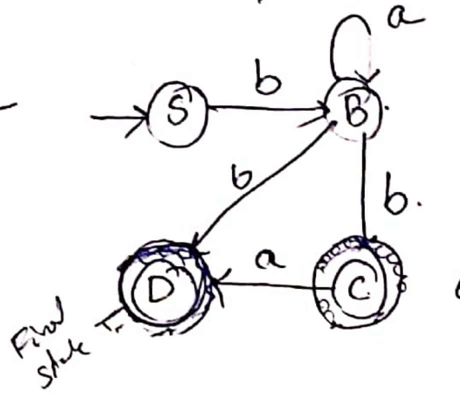
Method -

(1) convert the given right L.G to equivalent FA.
(2) From the FA interchange the initial state and final state.
(3) Reverse the directions of arrows on all edges.
(4) construct the regular grammar from this F.A

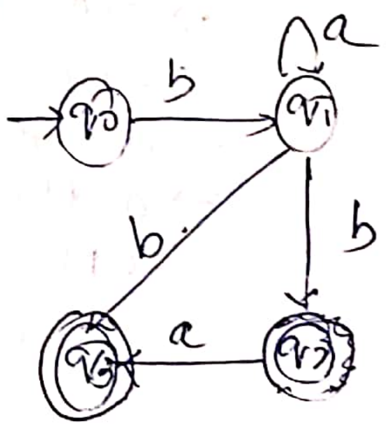
(1) convert the given R.L.G into equivalent L.L.G

$S \to bB$
$B \to bC$
$B \to aB$
$C \to a$
$B \to b$.

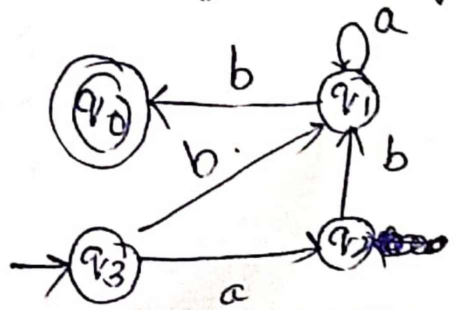
Sol:



$S \to q_0$
$B \to q_1$
$C \to q_2$
$D \to q_3$.



⇓ interchange.



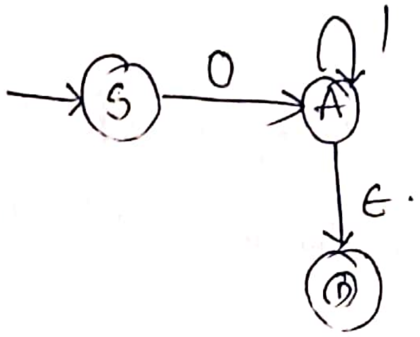
Equivalent Grammar.

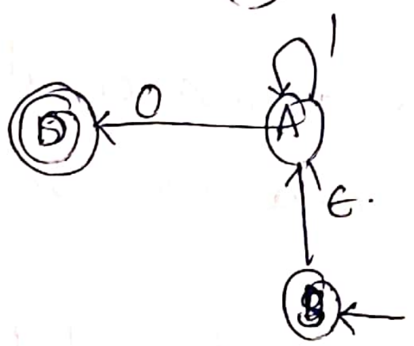
$q_2 \to q_1 b$ ~~$q_0 \to$~~
$q_1 \to q_1 a$ $q_3 \to q_2 a$.
$q_1 \to q_0 b$ $q_3 \to q_1 b$.

Scanned by CamScanner

② convert the following R.L.G to L.L.G

$S \to 0A$
$A \to 1A$
$A \to \epsilon$.





$B \to A\epsilon$ ie $B \to A$
$A \to A1$
$A \to 0$.

consider a String 01111, we will derive this String using R.L.G

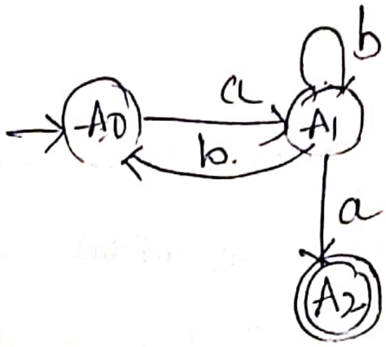
$S \to 0A$
$\to 01A$ $(A \to 1A)$
$\to 011A$ "
$\to 0111A$
$\to 01111A$
$\to 01111\epsilon$ $(A \to \epsilon)$
$\to 01111$

L.L.G $\Rightarrow$ $S \to A$.
$S \to A1$ $(A \to A1)$
$S \to A11$ "
$S \to A111$
$S \to A1111$ $(A \to 0)$
$S \to 01111$ "

Scanned by CamScanner

(2) Give an equivalent left linear grammar for the following right linear grammar.

$A_0 \to aA_1$
$A_1 \to bA_1$
$A_1 \to a$
$A_1 \to bA_0$.

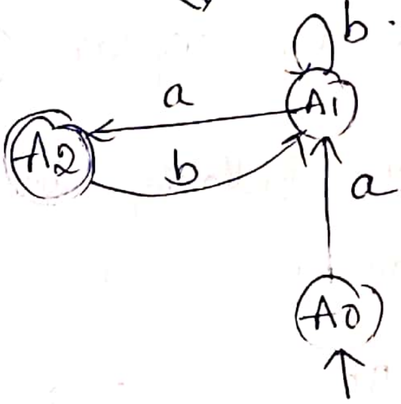


$\Downarrow$ conversion.

grammar.

$A_0 \to A_1a$.
$A_1 \to A_1b$
$A_1 \to A_2a$
$A_1 \to a$.
$A_2 \to A_1b$

$\Leftarrow$



consider the String abbaa.

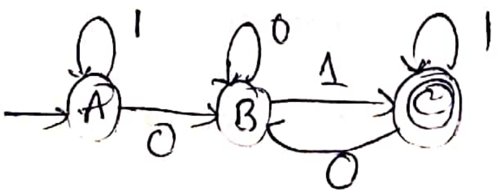
R.L.G

$A_0 \to aA_1$ $(A_0 \to aA_1)$
$\to abA_1$ $(A_1 \to bA_1)$
$\to abbA_0$ $(A_1 \to bA_0)$
$\to abbaA_1$ $(A_0 \to aA_1)$
$\to abbaa$ $(A_1 \to a)$

L.L.G:-

$A_0 \to A_1a$ $(A_0 \to A_1a)$
$\to A_2aa$ $(A_1 \to A_2a)$
$\to A_1baa$ $(A_2 \to A_1b)$
$\to A_1bbaa$ $(A_1 \to A_1b)$
$\to abbaa$ $(A_1 \to a)$

(3) obtain a L.L.G for the DFA as shown.



Scanned by CamScanner

Conversion of left-linear Grammar to Right-linear Grammar :-

Method :-

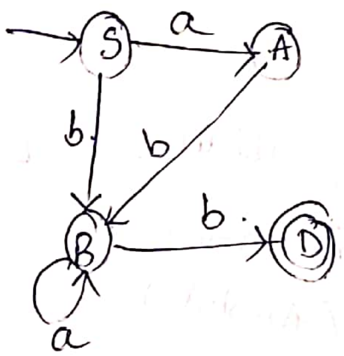
1) Convert the given left linear Grammar to FA.
2) Interchange the initial and final states of FA
3) Reverse the directions of all the arrows on all edges.
4) construct the regular grammar from this FA.

① Convert the following left-linear grammar to Right-linear.

$S \rightarrow Aa \mid Bb$

$A \rightarrow Bb$

$B \rightarrow Ba \mid b.$



⇓

Grammar

$S \rightarrow bB$

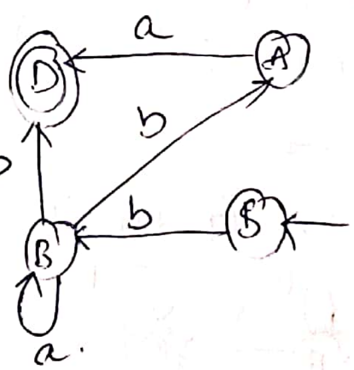
$A \rightarrow a$

$B \rightarrow aB$

$B \rightarrow bA$

$B \rightarrow b$. //.

⇐



Eg derivation of string bba.

$S \rightarrow Aa$
$\rightarrow Bba$
$\rightarrow bba$ //

$S \rightarrow bB$
$\rightarrow bbA$
$\rightarrow bba$ //.

Pumping Lemma for regular sets:—

→ Pumping Lemma provides a condn to prove that a language is not regular.

→ By considering a string z from the lang and pumping (generating) many i/p string from it.

→ if these i/p strings belong to the given lang then we say that lang is regular. if not it's not a regular language.

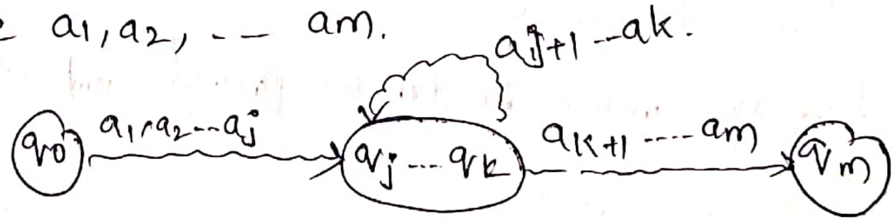
Theorem:—

Let L be a lang accepted by DFA denoted as $M = \{Q, \Sigma, \delta, q_0, F)$ where a_i is a string of L.

let n - be the no of states of DFA.

$i = 1, 2, \text{---} m$ where $m \geq n$.

i.e $a_1, a_2, \text{--} a_m$.



Case 1:— if there is a transition from q_0 to q_m without going thru the loop, then it accepts the string given as $a_1, a_2 \text{---} a_j \text{--} a_{k+1} \text{---} a_m$

$$\delta(q_0, a_1, a_2, a_j\, a_{k+1} \text{--} a_m) = \delta(\delta(q_0, q_1, \text{--} q_j), q_{k+1} \text{--} a_m)$$

$$= \delta(q_j, q_{k+1} \text{--} q_m)$$

$$= \delta(q_k, q_{k+1}, \text{--} q_m)$$

$$= q_m \,//$$

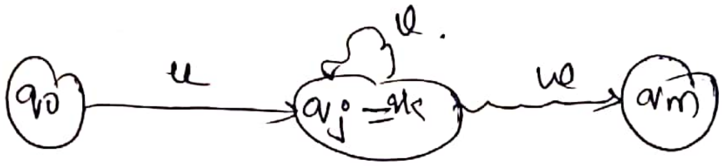
Scanned by CamScanner

Hence the String is pumped into three parts.

i.e $a_1, a_2 -- a_m$ is given as

$$\underbrace{a_1, a_2, --a_j}_{u} \quad \underbrace{a_{j+1} -- a_k}_{v} \quad \underbrace{a_{k+1} -- a_m}_{w}.$$

Lemma:— if L is lang given by a DFA which has a String z. the String z is devided int uv^iw such that $|uv| \leq n$ and $|v| \geq 1$



$z \to uv^iw$
$i = 1, 2, -- m$.

The String z is devided in u, v, w. if three Strings also along to the same lang then we call the lang as regular otherwise not regular. where $|uv| \leq n$ and $1 \leq |v| \leq n$.

Applications of pumming lemma:—

1) select the language which is to be proved not regular.

let $L = \{0^{i^2} \mid i = 1, 2, --- n\}$

↳ containing zeros of perfect square.

Assume that L is regular.

2) take an adversary variable 'n' where n denotes the no of states.

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

3) Consider a String z such that

$z = 0^{n^2} \Rightarrow |z| = |0^{n^2}| = n^2$.

Scanned by CamScanner

The String z is devided into u, v and w i.e.

$z = uv^iw$

if $i=1$

$|z| = |uvw| = |u| + |v| + |w| = n^2$

if $i=2$

$uv^2w = |u| + 2|v| + |w|$

$= |u| + |v| + |v| + |w|$

$n^2 + |v| \Rightarrow i \leq |v| \leq n$

$n^2 + n < (n+1)^2$

i.e $n^2 < n^2 + n < (n+1)^2$

as the value is not a perfect square.

5) As the resultant value are not belonging to the same lang. Hence we say that the lang is not regular.

Examples:-

① S.T $L = \{ 0^{i^2} \mid i = 1, 2, \ldots n \}$

$L = \{0, 0000, 000000000, \ldots\}$

$z = \underset{u}{0}\,\underset{v}{0}\,\underset{w}{00}$ $\quad i=2$

$uv^2w \Rightarrow 00000 \Rightarrow 5$ 0's not in the language.

Hence not a regular language

Scanned by CamScanner

(2) S.T $L = \{0^i 1^i \mid i \geq 1\}$ not regular.

$L = \{01, 0011, 000111, ---\}$

$Z = \underset{u}{0}\underset{v}{0}\underset{w}{11}$ $i=2$

$Z = uv^2w \Rightarrow 0\,01011 \notin L$

Here not a regular lang.

(3) $L = \{a^p \mid p \text{ is prime}\}$.

$Z = a^p$ 1,2,3,5

$L = \{a, aa, aaa, aaaaa, --\}$

$Z = \underset{u}{a}\underset{v}{a}\underset{w}{a}$.

$uv^2w \Rightarrow aaaa \notin L$. Here L is not regular.

(4) $L = \{a^n b a^n \mid n \geq 0\}$

$L = \{b, aba, aabaa, ---\}$

$Z = \underset{u}{a}\underset{v}{b}\underset{w}{a} \Rightarrow abba \notin L$. not regular.

(5). $L = \{a^n b^{2n} \mid n > 0\}$.

$L = \{abb, aabbbb, aaabbbbbb, ---\}$

$Z = \underset{u}{a}\underset{v}{b}\underset{w}{b} \Rightarrow abbb \notin L$. not regular.

(6) $L = \{ww \mid w \text{ in } (a+b)^*\}$.

$Z = ww$ (w in $(a+b)^*$).

$L = \{abab, baba, abaaba, ---\}$

$Z = \underset{u}{ab}\underset{v}{a}\underset{w}{b} \Rightarrow ababab \notin L$ not regular.

Scanned by CamScanner

grammars

① Consider the grammar $G = \{V, T, P, S\}$ where $V = \{S\}$

$T = \{a, b\}$ $P = \{S \to aSb, S \to ab\}$.

Sol:- $S \Rightarrow aSb$.

$\Rightarrow aaSbb$

$\Rightarrow aaaSbbb$.

$aaaabbbb \Rightarrow a^n b^n$.

$L(G) = \{a^n b^n \mid n \geq 1\}$.

② . $S \to 1S1$

$S \to 11$. $G = \{V, T, P, S\}$ where $V = \{S\}, T = \{1\}$,

$P \to \{S \to 1S1, S \to 11\}$.

sol:- $S \Rightarrow 1S1$

$11S11$

$111S111$ ... $\Rightarrow 1^n 1^n \Rightarrow 1^{2n}$.

$L(G) = \{1^{2n} \mid n \geq 1\}$

ii) $S \to 1S1$

$S \to \epsilon$.

Sol. $1 \Rightarrow 1S1$

$11S11$

$111S111$ ... $\Rightarrow 1^n 1^n \Rightarrow 1^{2n}$.

$L(G) = \{1^{2n} \mid n \geq 0\}$

Scanned by CamScanner

(2) Find the grammar that Generates the language,

$L = \{ 1^n 0^m \mid n, m \geq 0 \}$.

$S \to AB \qquad A \to 1A \qquad B \to 0B$

$\qquad\qquad A \to \epsilon \qquad B \to \epsilon$.

Checking :-

$S \to AB$

$1AB$

$11AB$

$11A0B$

$11\epsilon 0B$

110ϵ.

$110 \Rightarrow 1^n 0^m$

Scanned by CamScanner

Regular grammars:-

A Grammar $G=\{V,T,P,S\}$ is a regular grammar if it is either right linear grammar (or) left-linear grammar.

→ if the productions are of the form $A \to wB$ (or) $A \to w$ where A & B are variables and w is a String of terminals then such grammar is called a "right linear grammar".

→ if the productions are of the form $A \to Bw$ (or) $A \to w$ then it is called "left linear grammar"

Eg: consider grammar $G=(V,T,P,S)$ $V=\{S,X,Y\}$, $T=\{0,1\}$
$P \to \{S \to 0X, X \to 1Y, X \to 0, Y \to X0\}$

The grammar is not regular, bz to be a regular grammar. either it should be R.L (or) L.L. But the given grammar is of both the forms. Such a grammar is called "linear grammar"

→ The Regular grammar is always linear but a linear grammar need not be a regular grammar.

→ The language generated by the regular grammar is a regular lang.

Scanned by CamScanner

① Consider a right linear grammar G with prod's. of follows.

$S \to 01X$
$X \to 10Y$
$Y \to 0X$
$Y \to 11$

deriving string $S \Rightarrow 01X$
0110Y
01100X
0110010Y
01100100X
0110 010 010 Y.
0 11~~0~~ 010 010 0X
0 110 010 010 010Y.
0 110010 010 01011

Here the lang generated by grammar G is

$L(G) = 0110(010)^*11.$

Grammar: A grammar G is a 4-tuple G = (V, T, P, S)

where.

V ⇒ finite set of variables (or) non-terminals

T ⇒ " " constants (or) terminals.

P ⇒ finite set of productions (or) rules where.

each production of the form $A \to \alpha$.

A ⇒ variable & α is a string of symbols from $(V \cup T)^*$

S ⇒ special variable called the start symbol.

Scanned by CamScanner

Consider the lang $L(G) = 0(10)^*$

(i) Right Linear Grammar
$S \to 0A$
$A \to 10A \mid \epsilon$

(ii) $S \to S10 \mid 0 \to$ left linear grammar.

(or)

(iii)
$S \to AB$
$A \to 0$
$B \to 10B \mid \epsilon$.

Checking :—

$S \to 0A$
$\to 0\,10A$
$\to 01010A$
$\to 0101010A$
$\to 0\,10\,10\,10$
$\to 0(10)^*$

$S \Rightarrow S10$
$\Rightarrow S1010$
$\Rightarrow S101010$
$\to 0\,10\,10\,10$
$\to 0(10)^*$.

(3). Find the lang generated by the grammar

$G = \{\{S, X\}, \{0,1\}, P, S\}$

P includes: (i) $S \to 0X \mid 1S$
$X \to 1X \mid 1$.

Checking

$S \Rightarrow 0X$
$S \to 01$

$S \Rightarrow 1S$
$\Rightarrow 10X$
$\Rightarrow 101$

$S \Rightarrow 0X$
$01X$
$011X$
$0111X$
$01111X$
011111
01^m $(m \geq 1)$

$S \Rightarrow 1S$
$11S$
$111S$
$11101X$
111011
$1^n 0 1^p$
$n \geq 0$
$p \geq 0$.

$L(G) = \{1^n 0 1^p \mid n \geq 0, p \geq 1\}$

$\Downarrow$

Set of strings over $\{0,1\}$
Containing exactly one '0' and ending in '1'

Scanned by CamScanner

(4) ii) $S \to 1S1$
$S \to \epsilon$

$S \Rightarrow 1S1$
$\Rightarrow 11S11$
$111S111$
$1^n 1^n \Rightarrow 1^{2n}$ $L(G) = \{1^{2n} \mid n \geq 0\}$

(4). Describe the lang set $L(G)$ for each of the foll

a) $S \to aX$ $X \to bX$
$S \to bX$ $X \to b.$

$S \Rightarrow aX \Rightarrow abX \Rightarrow abbX \Rightarrow abbbX \Rightarrow abbbb \Rightarrow ab^n \; n \geq 1$
$S \Rightarrow bX \Rightarrow bbX \Rightarrow bbbX \Rightarrow b^m \quad m \geq 1$

$L(G) = \{ab^n \mid n \geq 2\}$

b) $S \to a|aX$, $X \to bY$, $Y = aX|a$.

$S \Rightarrow a$, $S \Rightarrow aX$
abY
$abaX$
$ababY$
$ababaX$
$ababab Y$
$abababa \Rightarrow a(ba)^n$.

$L(G) = \{a(ba)^n \mid n \geq 0\}$

Scanned by CamScanner

$L = \{a^n \mid n \geq 0\}$.

) $A \to Aa \mid \epsilon$

$A \to aA \mid \epsilon$.

ii) $(a+b)^*$

$S \to aS \mid bS \mid \epsilon$.

(iii) $L = \{aa, ab, ba, bb\}$

$S \to aa \mid ab \mid ba \mid bb$.

$(a+b)(a+b)$

$S \to AA \quad A \to a \mid b$.

(iv) Atleast length '2'.

$\underbrace{(a+b)}_{A} \underbrace{(a+b)}_{A} \underbrace{(a+b)^*}_{B}$

$S \to AAB$

$A \to a \mid b$.

$B \to aB \mid bB \mid \epsilon$.

(v) Atmost two.

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

$S \to AA$

$A \to a \mid b \mid \epsilon$.

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

$S \to aAb$.

$A \to aA \mid bA \mid \epsilon$.

Scanned by CamScanner

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

$S \to aAb \mid bAa$
$A \to aA \mid bA \mid \epsilon$.

Scanned by CamScanner

3.5.2 Direct Method for Conversion of r.e. to FA

This method is a direct method for obtaining FA from given regular expression. This is called a **subset** method. The method is given as below -

Step 1 : Design a transition diagram for given regular expression, using NFA with ε moves.

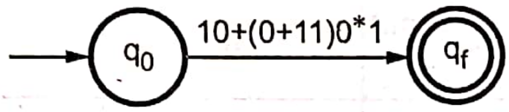
Step 2 : Convert this NFA with ε to NFA without ε.

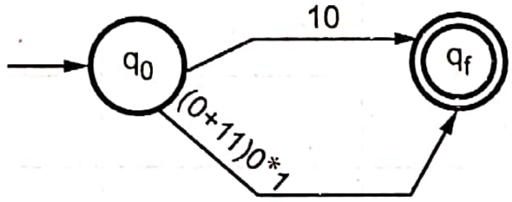
Step 3 : Convert the obtained NFA to equivalent DFA.

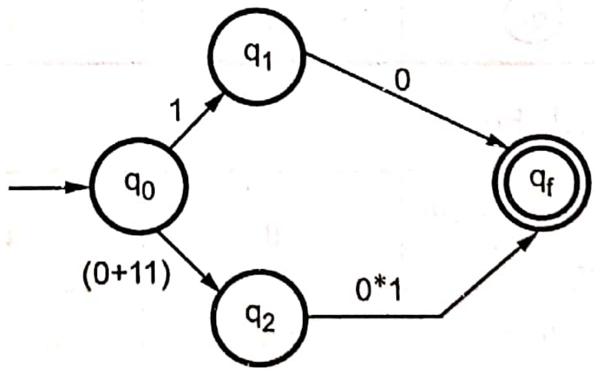
Let us understand this method with the help of some example.

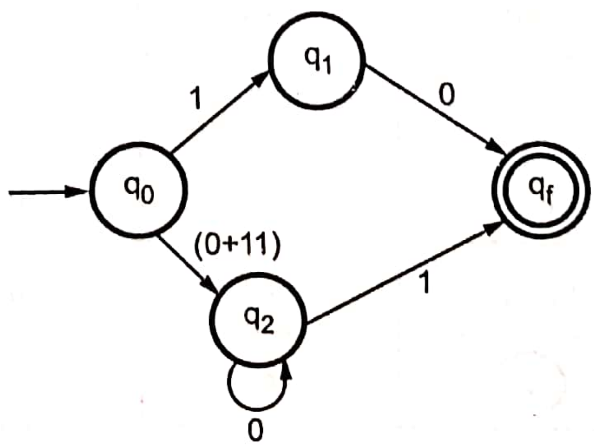
➡ **Example 3.34 :** *Design a FA from given regular expression* $10 + (0 + 11)\,0^{*}\,1$.

Solution : First we will construct the transition diagram for given regular expression.

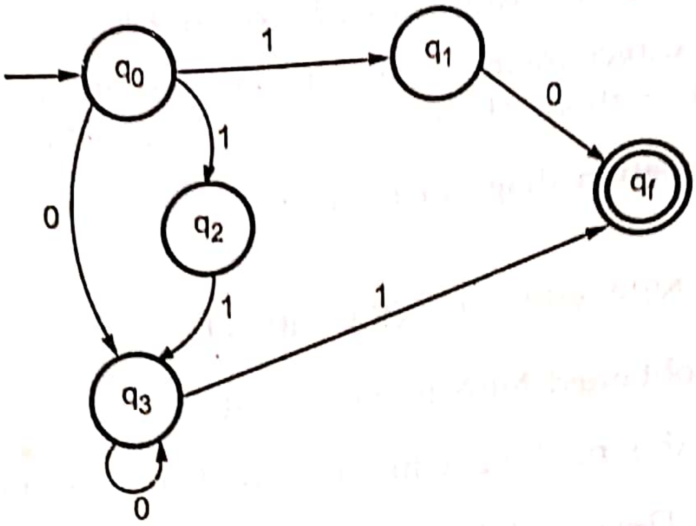








Scanned by CamScanner



Now we have got NFA without ε. Now we will convert it to required DFA for that, we will first write a transition table for this NFA.

State \ Input	0	1
q_0	q_3	$\{q_1, q_2\}$
q_1	q_f	ϕ
q_2	ϕ	q_3
q_3	q_3	q_f
q_f	ϕ	ϕ

The equivalent DFA will be

State \ Input	0	1
$[q_0]$	$[q_3]$	$[q_1, q_2]$
$[q_1]$	$[q_f]$	ϕ
$[q_2]$	ϕ	$[q_3]$
$[q_3]$	$[q_3]$	$[q_f]$
$[q_1, q_2]$	$[q_f]$	$[q_3]$
$[q_f]$	ϕ	ϕ

Scanned by CamScanner