Regular languages & Finite Automata

DOMS | Page No. | Date / /

* Before understanding a regular language we have to understand Kleene closure.

- **Kleene closure**

- given an alphabet Σ, we wish to define a language in which any string of letters from Σ is a word, even the null string.

- This language we shall call the Closure of the alphabet. It is represent by Σ^*

- This notation is sometimes known as the Kleene star.

Example

(1) if $\Sigma = \{x\}$, then

$\Sigma^* = \{\wedge, x, xx, xxx \dots\}$

(2) if $\Sigma = \{0, 1\}$, then

$\Sigma^* = \{\wedge, 0, 1, 00, 01, 10, 11, 000 \dots\}$

(3) if $\Sigma = \{a, b, c\}$ then

$\Sigma^* = \{\wedge, a, b, c, aa, ab, ac, \dots\}$

DOMS | Page No.
Date / /

lexicographic order → listed all the words according alphabatically & length wise.
{ ∧, aa, ab, ba, bb, aaa, aab, ... }

Definition of S^*

→ if S is a set of words, they by S^* we mean the set of all finite strings formed by concatenating words from S, where any words may be used as often as we like, & where the null string is also included.

* if S = {aa, b} then

S^* = { ∧ plus any word composed of factor of aa & b }.

= { ∧ plus all string of a's & b's in which the a's occure in even clumps }

= { ∧, b, aa, bb, aab, bbb, aaaa, aabb, baab, bbaa, bbbb, aaaab, aabbaa, aabbb, baaaa, baabb, bbbaa, bbbbb... }

* if S = {a, ab} then

S^* = { ∧ plus any word composed of factor of a & ab }

S^* = { ∧ plus all string of a's & b's except those that start with b & those that contain bb as substring }

= { ∧, a, aa, ab, aab, aba, aaaa, aaab, aaba, aabaa, abaa, abab, aaaaaa, aaaaab, aaaaba, aaabaa, ---- } //.

⇒ if for some reason we wish to modify the concept of closure to refer to only the concatenation of some (not zero) string from a set S, we use the notation + instead of *. for example,

if $\Sigma = \{x\}$

$\Sigma^+ = \{ x, xx, xxx, --- \}$

$\Sigma^* = \{ \wedge, x, xx, xxx, --- \}$

→ if S is a language that do not contain ∧, then $S^+ = S^*$.

→ This plus operation is sometimes called positive closure.

* Means arbitary number of concatenation

RE is used to generate pattern & FA is " " recognize that pattern

DOMS Page No. Date / /

* Regular language :- A regular language over an alphabet Σ is one that can be obtained from these basic language using the operation of union, concatenation, & Kleene *.

- Basic language is $\{\wedge, \phi, a\}$ generalize (formate) $a \in \Sigma$ //o

- It FA is used to recognize pattern of strings, regular Expression are used to generate pattern of strings.

- RE is an algebric formula whose value is a pattern consisting of a set of strings. Called the language of the Expression

A Regular languages & Regular Expressions over Σ

The set R of regular languages over Σ, & the corresponding regular expressions, are defined as follows

1. ϕ is an element of R, & the corresponding regular expression is ϕ.

2. $\{\wedge\}$ is an element of R, & the corresponding regular Expression is $\wedge$.

FA → acceptors
Grammer → generators of strings of language
RE → Representors of language
RL → accepted by FA

DOMS | Page No.
Date / /

3. for each $a \in \Sigma$, $\{a\}$ is an element of R, & the corresponding regular Expression is a.

4. if L_1 & L_2 are any elements of R, & r_1 & r_2 are the corresponding regular Expressions,

(a) $L_1 \cup L_2$ is an element of R, & the corresponding regular Expression is $(r_1 + r_2)$;

(b) $L_1 L_2$ is an element of R, & the corresponding regular Expression is $(r_1 r_2)$;

(c) L_1^* is an element of R, & the corresponding regular expression is (r_1^*).

only those languages that can be obtained by using Statement 1-4 are regular language over Σ.

RE is used to represent a RL

→ you can only Represent RE if language is RL

- Example

→ union $(+)$

→ concatenation $(.)$

→ kleen closure $(*)$

Example

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

$L = \{$ set of all string of length exactly 2 $\}$

$= \{aa, ab, ba, bb\}$ ← (union of)

R.G $= aa + ab + ba + bb$

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

$= (a+b).(a+b)$

↑ Concatenation

[Here $a.b \neq b.a$] Not Commutative
[$a+b = b+a$] Commutative.

(2)

$\Sigma = \{a\}$

$\Sigma^0 = \{\epsilon\}$

$\Sigma^1 = \{a\}$

$\Sigma^2 = \{aa\}$

$\Sigma^3 = \{aaa\}$

$\Sigma^4 = \{aaaa\}$

⋮

Σ^* = Infite union of strings in given all possibles alphabet

$\Sigma^* = \Sigma^0 \cup \Sigma^1 \cup \Sigma^2 \cup \Sigma^3 \cup \Sigma^4 \cup \ldots$

$= \{\epsilon, a, aa, aaa, aaaa \ldots\}$

$\Sigma^0 = \{\epsilon\}$

$\Sigma^1 = \{a, b\}$

$\Sigma^2 = \{aa, ab, ba, bb\}$

$\Sigma^3 = \{aaa, aab, aba, abb \ldots\}$

$\Sigma^* = \{\in, a, b, aa, ab, ba, bb, aaa \ldots\}$

$= \Sigma^0 \cup \Sigma^1 \cup \Sigma^2 \cup \Sigma^3 \cup \ldots$

$\Rightarrow a^* = \{\in + a + aa + aaa + aaaa\}$.

$(a+b)^* = \{\in + a + b + aa + ab + ba + bb + aaa \ldots\}$

$\Rightarrow a^+ = \{a + aa + aaa + aaaa\}$

$a^+ = a \cdot a^* = a^* \cdot a$

Example

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

$L = \{$set of all strings of length $r = 2\}$

$= \{w \mid |w| <= 2\}$

$= \{\in, a, b, aa, ab, ba, bb\}$ ← strings

R.E = (1) $\in + a + b + aa + ab + ba + bb$ } length is limited

$= \in + a + b + a(a+b) + b(a+b)$

$= \in + (a+b) + (a+b) \cdot (a+b)$

(2) $= \in + (a+b) \cdot (\in + (a+b))$ } $\in \cdot a = a \cdot \in = a$

③ R.E = ε + a + aa + ab + b + ba + bb
= ε + a (ε + a + b) + b (ε + a + b)
= ε + (ε + a + b) . (a + b)

④ R.E = $\overline{(\varepsilon + a + b) . (\varepsilon + a + b)}$

② Σ = {a, b}
L = {set of strings of length exactly three}
= {aaa, aab, aba, abb, baa, bab, bba, bbb}

R.E = aaa + aab + aba + abb + baa + bab + bba + bbb

= a (aa + ab + ba + bb) + b (aa + ab + ba + bb)
= (aa + ab + ba + bb) . (a + b)
= [a (a + b) + b (a + b)] . (a + b)
= (a + b) . (a + b) . (a + b)

① L = {set of all strings of even length strings}
= {ε, aa, ab, ba, bb, aaaa, ...}
= ε + aa + ab + ba + bb + aaaa + ...

= ((a + b) . (a + b))*

* can be replaced by power $\frac{1}{2}$

If it is power 0 = ε
1 = 2 length string
2 = 4 length string
3 = 6 " "
$\frac{3}{4}$

DOMS | Page No.
Date / /

(3) $L = \{ w \mid |w| \bmod 2 \cong 1 \bmod 2 \}$

odd length string

$= \{ a + b + aaa + aab + aba + \cdots \}$

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

1
3
5
7
!

$2n$	$n \geq 0$	even no
$2n+1$	$n \geq 0$	odd no

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

$L = \{$ set of all strings over starting with 'a' $\}$

$= \{ a, aa, ab, aaa, aba, abb, aab, aaaa \}$

$R.E = a + aa + ab + aaa + aba + abb + aab + aaaa$

$= a [\varepsilon + a + b + aa + ab + ba + bb + ab + \cdots]$

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

(5) $L = \{$ set of all string ending with a $\}$

$R.E = \boxed{(a+b)^*.a}$

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

$L = \{$ set of all string

$= \{ w \mid |w| \bmod 3 \cong 0 \bmod 3 \}$

length 0, 3, 6, 9, 12...

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

[3n =]

7. $L = \{ w \mid |w| \bmod 3 \cong 2 \bmod 3 \}$

$=$

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

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

Language string length $= 2, 5, 8, 11$

$n^* = 0$

→ String of length $3n = 3*0 = 0$

$\rightarrow n^* = 1 \quad (2 + 3n) = 5$

$\rightarrow n^* = 2 \quad (2 + 6) = 8$

8. $L = \{$ starting & ending with same symbols $\}$

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

Starting with a, a, b, b

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

→ this does not represent single a & b which also a start with a & b. So we can write.

Ans: $a(a+b)^*a + b(a+b)^*b + a + b$ ✓

DOMS | Page No.
Date / /

(9) L = {Starting & Ending with diff. symbol}

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

(10) L = {String of length atleast 2}

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

(11) $L = \{a^n b^m \mid n, m \geq 1\}$ {ab, abb, abbb+... aab, aabb}

= $a^+ b^+$

= $a.a^*.b.b^*$

(12) $L = \{a^n b^m \mid n, m \geq 0\}$ {ε, aa, aaa, aaaa... a+b, bb, bbb...}

= $a^* b^*$

(13) $L = \{a^n b^m c^k \mid n, k \geq 1, m \geq 0\}$

= $a^+.b^*.c^+$

(14) $L = \{a^n b^m c^k \mid n, m, k \geq 1\}$

= $a^+ b^+ c^+$

= $a.a^* b.b^* c.c^*$

Date / /

(15) String with an odd numbers of 1's

$\Sigma = \{0, 1\}$

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

$= 0^*10^*(10^*10^*)^*$ or

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

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

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

(16) String of length ≤ 6 or less.

Ans :- $(0+1+\wedge)^6$

(17) String Ending in 1 & Not containing 00.

Ans :- ~~$(0+(0)^*1$~~

$(1+01)^*(1+01)$ or

$(1+01)^+$ or

01^*1

DOMS | Page No.
Date / /

(18) String ending with 0

$= (0+1)^* 0$

(19) String with next-to-last Symbol is 0

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

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

(20) String ending with 11.

$= (0+1)^* 11$

(21) String with an Even number of 0's & an Odd Number of 1's.

$\Rightarrow 01(1^*10^*)^*0$ (Even 1's, odd 0's)

or

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

(22) The language of C identifiers

temporarily use l = letters (a--z, A--Z)
d = digits (0---9)

Identifier ⇒ any string of length 1 or more, that contains only letters, digits, _ (underscores), begins with a letter or an underscore.

R.E :- $(l+\_)(l+d+\_)^*$

Date / /

* Explain $(a+b)^* \neq a+b^*$?

→ Kleene * has highest precedence & + has the lowest.

→ $(a+b)^*$ can generate ∧, a, b, abba ——
where
$a+b^*$ " " ∧, a, b, bb, bbb,

* Difference bet^n $(a+b)^*$, $(a+b)^+$ And $(a+b)$.

→ $(a+b)^*$ → It contains all substring of a's & b's including NULL.

→ $(a+b)^+$ → It contains all substring of a's & b's but it does not include NULL.

→ $(a+b)$ → Means that it compulsory take any one of either a or b. Hence there will be only 2 substring a & b.

DOMS | Page No.
Date / /

☆ Consider the Find R.E corresponding to each of the following subset of $\{0,1\}^*$

(1) The language of all strings containing exactly two 0's.

⇒ $1^* 0 (1)^* 0 1^*$

(2) The language of all strings containing at least two 0's.

→ $(0+1)^* 0 (0+1)^* 0 (0+1)^*$ or

$0^+ 1^* 0^+$ or

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

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

(3) The language of all strings that do not end with 01.

→ $(0+1)^* 10^+$

$\wedge + 1 + (0+1)^* 0 + (0+1)^* 11$

(4) The language of all string that begin or end with 00 or 11.

→ $(00+11)(0+1)^*(00+11)$ or

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

5. The language of all strings not containing the substring 00.

→ ~~(10+1+01)*(0+1)*~~

$\equiv$

$(0+\wedge)(10+1)^*$ $\equiv$ $0(10+1)^*$

$(01+1)^*(\wedge+0)$ ~~$(01+1)0^*$~~

6. The language of all string in which the number of 0's is even.

→ $1^*(01^*01^*)^*1^*$

7. The language of all strings containing no more than one occurrence of the string 00.

→ $(1+01)^*(00)(10+1)^*$

$\equiv$

$(1+01)^*(00+0+\wedge)(10+1)^*$

8. The language of all string in which every 0 is followed immediately by 11.

→ $1^*(011)^*1^*$

9. The language of all string in which every 0 is followed by containing both 11 & 010 as substring

→ $(0+1)^*(11)(0+1)^*010$

DOMS | Page No.
Date / /

$$\underline{\underline{=}}$$

$$(0+1)^*(11(0+1)^*010 + 010(0+1)^*11)(0+1)^*$$

Find R.E corresponding to each of the language defined recursively below.

$\Rightarrow$

(1) $\wedge \in L$; if $x \in L$, then $00|x$ & $x11$ are elements of L; nothing is in L unless it can be obtained from these 2 statements.

$\rightarrow$ $(001)^*(11)^*$

(2) $0 \in L$; if $x \in L$, then $00|x$, $x00|$, & $x11$ are elements of L; nothing is in L unless it can be obtained from these 2 statements.

$\rightarrow$ $(001)^*0(001+11)^*$

(2) $\wedge \in L$; $0 \in L$; if $x \in L$, then $00|x$ & $11|x$ are in L; nothing is in L unless it can be obtained from these 2 statements.

$\rightarrow$ $(0+\wedge)(00+11)^*$

DOMS | Page No.
Date / /

☆ ① Describe as simply as possible the language corresponding to each of the following regular Expressions

① $(0+10^*)$: $L = \{0, 1, 10, 100, 1000, 10000, 10000 \cdots\}$

② $(0^* \bullet 10^*)$: $L = \{1, 01, 10, 010, 0010, 0100 \cdots\}$

③ $(0+\varepsilon)(1+\varepsilon)$: $L = \{\varepsilon, 0, 1, 01\}$

④ $(a+b)^* abb$: L = Set of strings of a's & b's ending with the string abb. So $L = \{abb, aabb, babb, aaabb, ababb \cdots\}$

⑤ $(aa)^*(bb)^* b$: L = Set of string consisting of even number of a's followed by odd number's of b's so $L = \{b, aab, bbb, aabbb, aabbbbb \cdots\}$

⑥ $(aa+ab+ba+bb)^*$: String of a's & b's of even length can be obtained by concatenating any combination of the strings aa, ab, ba and bb including null,
So, $L = \{aa, ab, ba, bb, aaab, aaba, aabb, abba, abbb, baabb \cdots\}$

⑦ $0^*1(0^*10^*1)^*0^*$: All string containing odd number of 1
: $L = \{1, 010, 10, 01010, 111 \cdots\}$

⑧ $((0+1)^3)^*(\wedge+0+1)$

→ $\{\wedge, 000, 0, 1, 111, 1110, 1111, 0000, 0001,$

DOMS | Page No.
Date / /

$= \{\wedge, 0, 1, 000, 111, 010, 011, 110, 100, 001, 101, 0000, 0001, 1110, 1111, 0100 \ldots\}$

(9) $(1+01)^*(0+01)^*$

All string which does not contain substring like 00×11

$= \{\wedge, 01, 111, 011, 101, 110, 010, 101, \ldots\}$

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

$= \{010, 101, 0100, 0101, 1010, 1011, 00100, 00101, 00110, 01101, 11010, 10101, 11011, \ldots\}$

All String containing 10 & 01 both as substring.

Consider the two regular expressions

$r = 0^* + 1^*$ $\quad S = 01^* + 10^* + 1^*0 + (0^*1)^*$

(1) find a string corresponding to r but not to S.

Ans: $00, 11, 000, 111, \ldots$

(2) find a string corresponding to S but not to r.

Ans: $01, 10, 001, 110 \ldots$

(3) find a string corresponding to both r & S.

Ans: $010, 0101 \ldots$

(4) find a string corresponding to neither r nor S.

Ans $\wedge, 0, 1$.

☆ Prove the formula

$$(111^*)^* = (11 + 111)^*$$

L.H.S = $(111^*)^*$

= $\wedge, 111, 11, 1111, 11111, 111111 \ldots$

R.H.S = $(11 + 111)^*$

= $\wedge, 11, 111, 1111, 11111, 111111 \ldots$

L.H.S = R.H.S

☆ Prove the formula

$$(aa^*bb^*)^* = \wedge + a(a+b)^*b$$

L.H.S = $(aa^*bb^*)^*$

= $\wedge, ab, aab, abb, aaab, aabb,$
$aaabbb \ldots$

R.H.S = $\wedge + a(a+b)^*b$

= $\wedge, ab, aab, abb, aaab,$
$abbb, aaabbb \ldots$

L.H.S = R.H.S

Finite Automata

DOMS | Page No.
Date / /

→ Finite Automata is a simple idealized machine used to recognize patterns within input taken from some character set.

It is a recognizer for "Regular Languages".

Finite Aurома can be classified into 2 types
(1) Deterministic finite Automata (DFA)
(2) Non-deterministic finite Automata (NFA/NDFA)

DFA :- The machine can exist in only one state at any given time.

NFA :- The machine can exist in multiple states at the same time.

② Formal Definition of a DFA

→ A FA can be represented by 5-tuple $(Q, \Sigma, q_0, A, \delta)$, where

Q is a finite set (whose elements we will think of as a states); (non empty set) (It is a finite set of sets)

Σ is a finite alphabet of input Symbols; (non-empty)

$q_0 \in Q$ (the initial state)

$A \subseteq Q$ (the set of accepting states);

δ is a fun^n from $Q \times \Sigma$ to Q $(Q \times \Sigma \to Q)$ (the transition fun^n).

for any element q of Q & any symbol $a \in \Sigma$,

DOMS | Page No.
Date / /

We interpret $\delta(q, a)$ as the state to which the FA moves, if it is in state q & receives the i/p a.

② Draw a finite Automata for given language

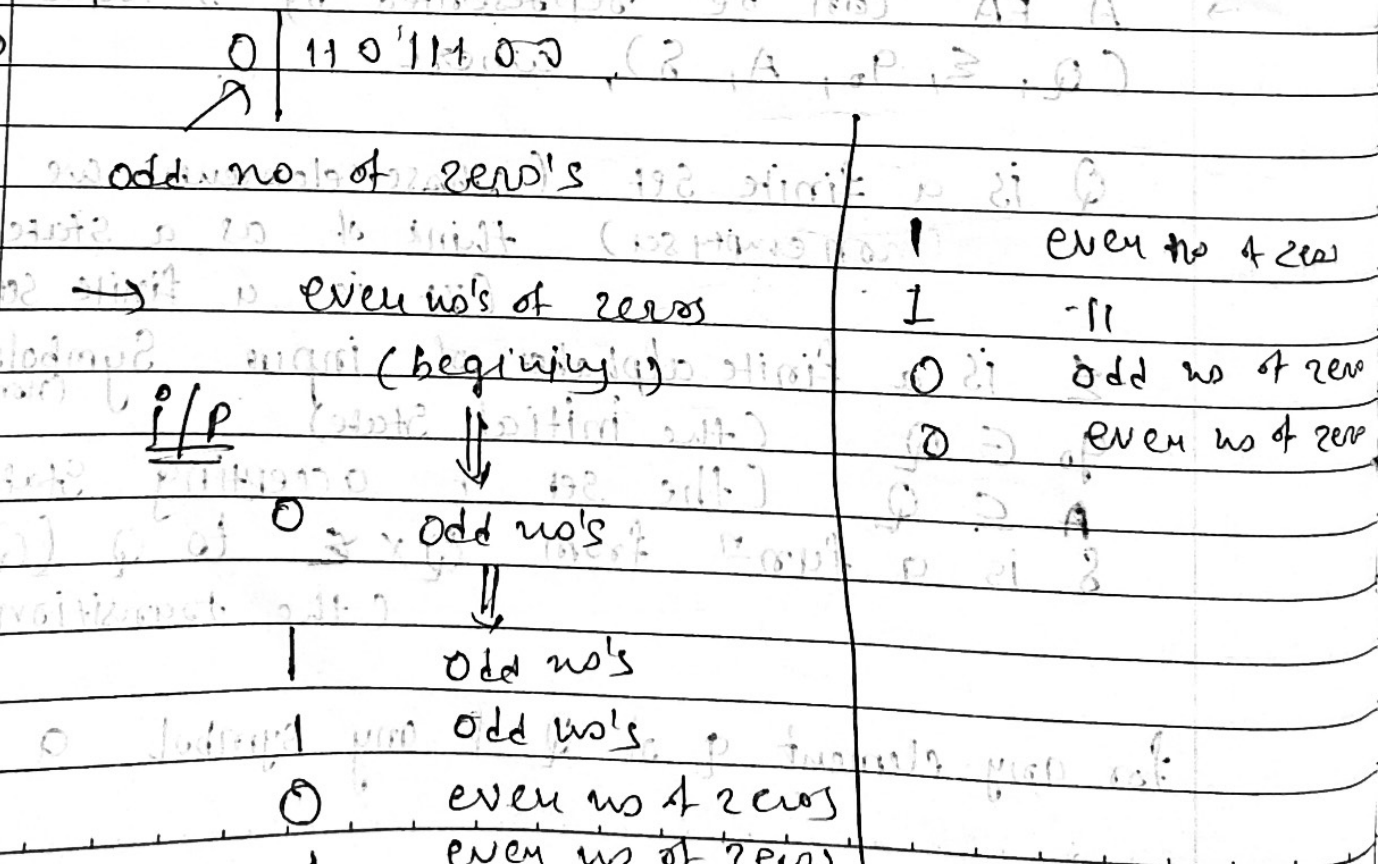
$\Sigma = \{0, 1\}$

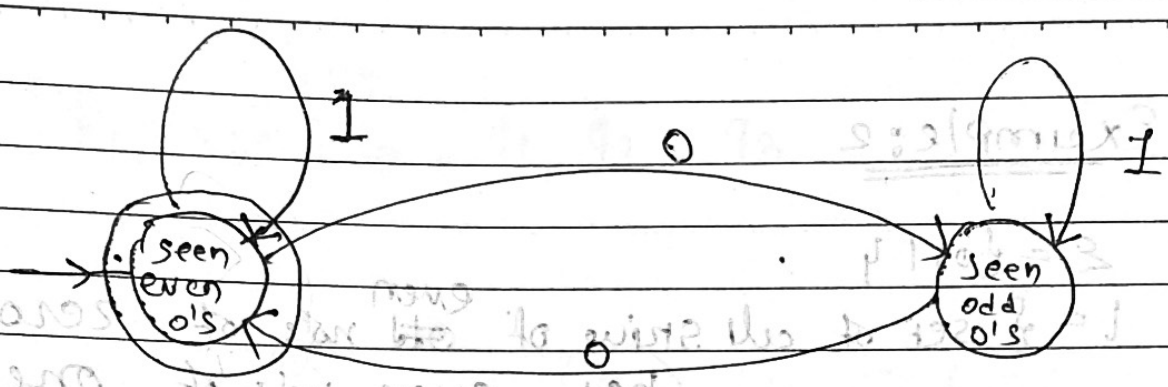
$L = \{$set of all string which has even no. of zero's$\}$

$L = \{\Lambda, 1, 11, 111, \ldots, 00, 100, 001, 11100, \ldots\}$

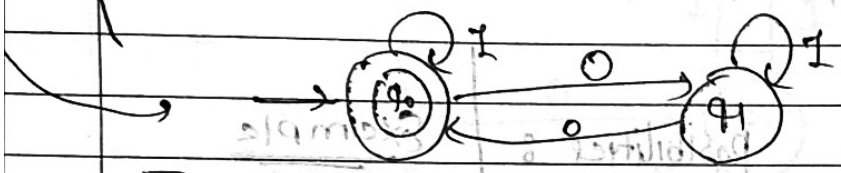
Ex :- 0110111 00

↑ empty string → even no. of zeros (zero zero's)





(Possibilities of a State → even no of zero's, odd no of zero's)



Transition Table

	Present State	Symbol Seen	Next State
δ	q_0	0	q_1
	q_0	1	q_0
	q_1	0	q_0
	q_1	1	q_1

Note: There will be relationship b/w

Q & Σ

Q → finit set, Σ → finit set of symbols.

$\delta : Q \times \Sigma \rightarrow Q$

Ex $Q = 100$ States

$\Sigma = 10$ symbols

Transition tables has 1000 entry. Here in above example 2 State & 2 i/p symbols so $2 \times 2 = 4$ entry.

DOMS | Page No.
Date / /

④ Example: 2

$\Sigma = \{0, 1\}$,

$L = \{$ set of all string of even no's of zero & has even no's of one$\}$

I have to give machine

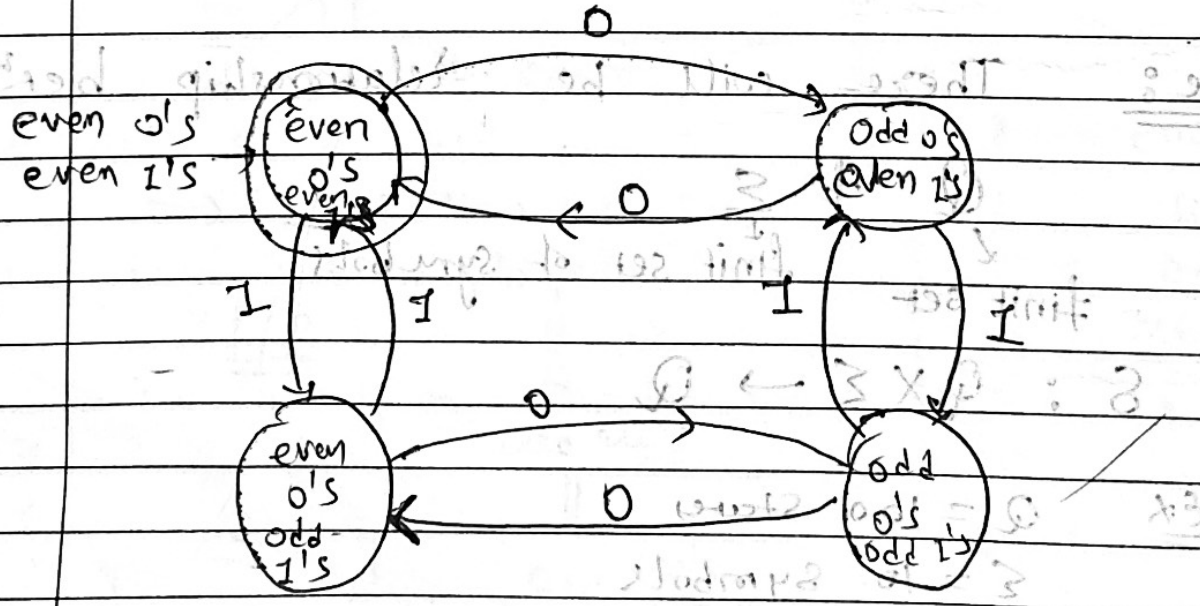
$M_2 = (Q, \Sigma, \delta, q_0, A)$

★ There are 4 posibilities :

	even no's of	even 1's
①	0's	
②	even 0's	odd 1's
③	Odd 0's	Odd 1's
④	odd 0's	even 1's

Example

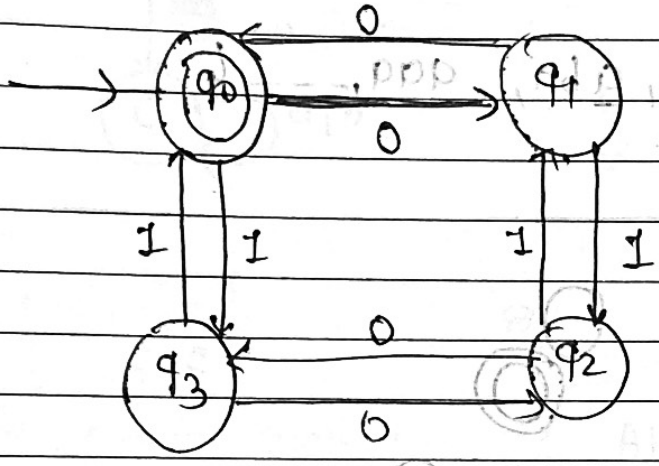
0 1 1 0 1 1 0 0 1 0



DOMS | Page No.
Date / /

with

Replace q_0, q_1, q_2, q_3



M for L

$Q = \{ q_0, q_1, q_2, q_3 \}$

$q_0 = \{$ initial state $\}$

$A = \{ q_0 \}$

Present State	i/p Symbol	next State
q_0	0	q_1
q_0	1	q_3
q_1	0	q_0
q_1	1	q_2
q_2	0	q_3
q_2	1	q_1
q_3	0	q_2
q_3	1	q_0

Example : 3

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

$L = \{$ set of all string starting with 'a' $\}$

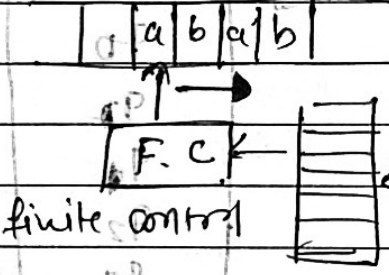
$= \{a, aa, ab, aba, aaa, \ldots\}$

Note:-

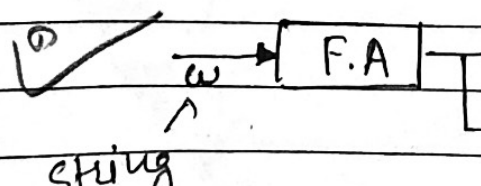
- State :
- final state :

Initial State : →

transition : $\xrightarrow{a}$



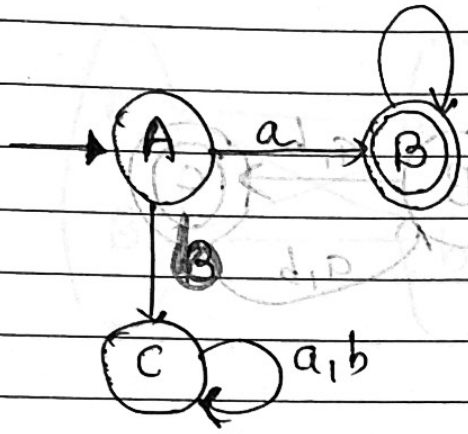
F.A basically represent your computer machine. (It is a abstract diagram of computer). Every language represent a program which is going to load into machine.

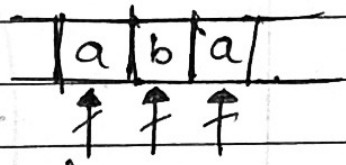


→ If string is accepted than that string is in language

DOMS | Page No.
Date / /

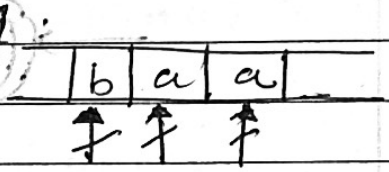
Check this string





→ this, language string is in the final state.

b'cz we are at the final state



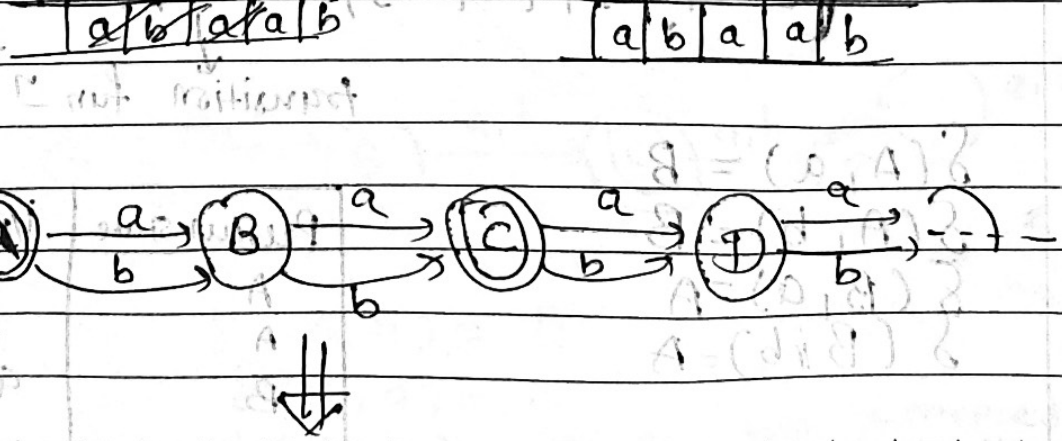
→ At the end we reached at state C which is not a accepting state, so this string is rejected. it means this is not present in the language.

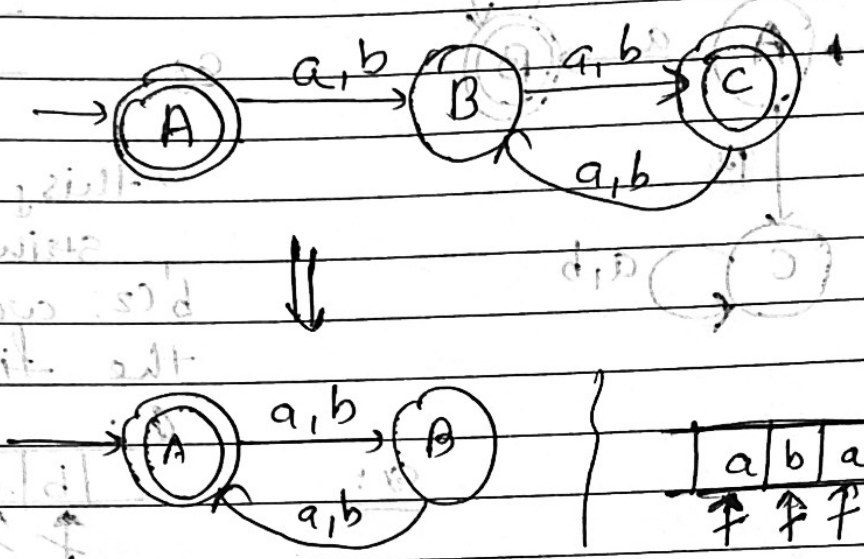
Example : 4

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

$L = \{$ set of all even length strings $\}$

$= \{ \wedge, aa, ab, ba, bb, aaaa, aaab, aaba, aabb, \_\_ \}$





→ Here string will be at state B, but it is not a accepting state so it is string which is not in language.

→ $\delta(A, a) \Rightarrow B$
$\delta(B, b) \Rightarrow A$
$\delta(A, a) \Rightarrow B$
$\delta(B, a) \Rightarrow A$
$\delta(A, b) \Rightarrow B$

* give langu formal definition of above

Example

F. A = { Q, Σ, q_0, δ, A }

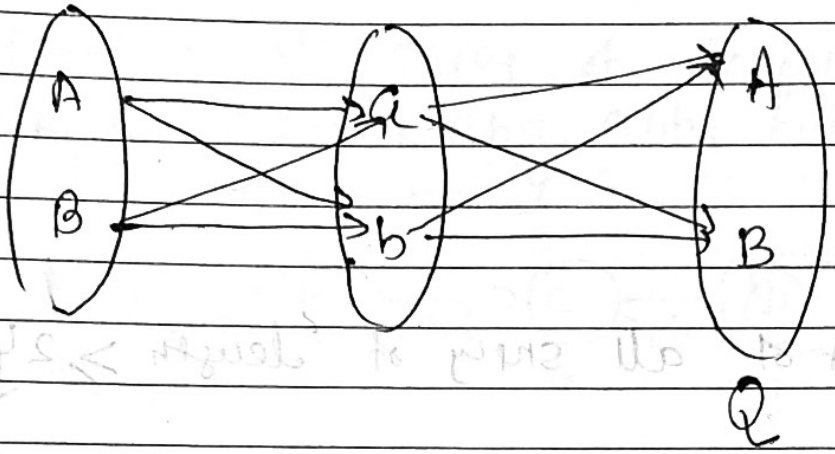
F. A = ({ A, B }, {a, b}, A, δ, {A})

transition fun

$\delta(A, a) = B$
$\delta(A, b) = B$
$\delta(B, a) = A$
$\delta(B, b) = A$

Present state	i/p symbole	next state
A	a	B
A	b	B
B	a	A
B	b	A

$Q \times \Sigma \rightarrow Q$



Example : 5

$\Sigma = \{a, b\}$ length

$L = \{$set of all strings $\leq 2s\}$

$L = \{\underset{0}{\wedge}, \underset{1}{a,b}, \underset{2}{aa, ab, ba, bb}\}$ ← length

a	a	a
↑		

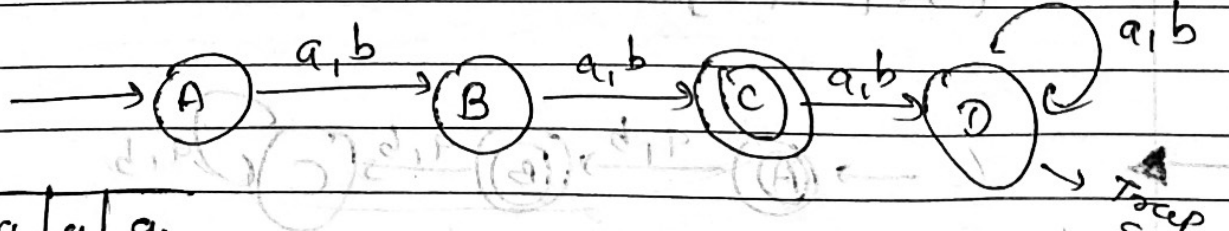
$\delta(A, a) = B$

$\delta(B, a) = C$

$\delta(C, a) = D$ ← not a final state

$L = \{$set of all string of length $= 2\}$

$= \{aa, ab, ba, bb\}$

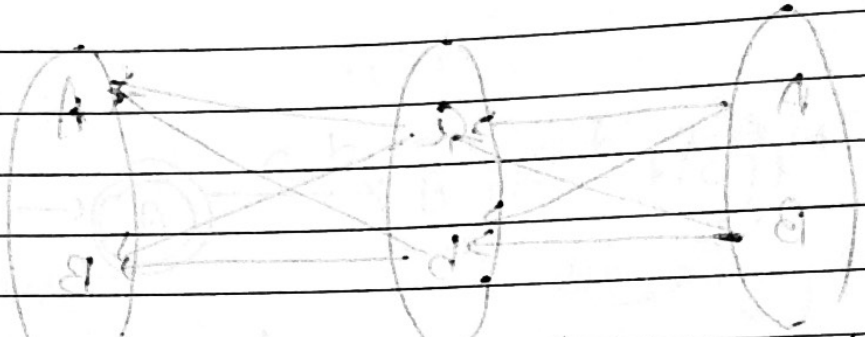


a	a	a

$\delta(A, a) = B$

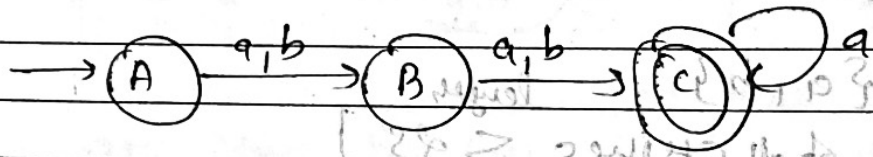
$\delta(B, a) = C$

$\delta(C, a) = D$ ← not accepted not a final state



L = { set of all string of length ≥ 2 }

= { aa, ab, ba, bb, aaa, aab, aba -- }

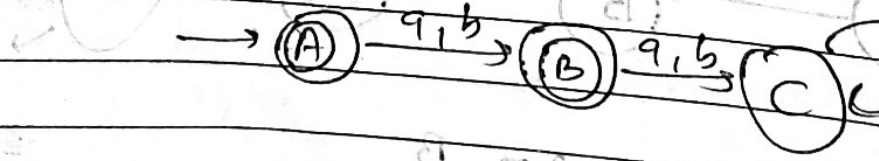


Note: Generalize Observation

		State required
$\|\omega\| \leq n$	⇒	$n+2$
$\|\omega\| < n$	⇒	$n+1$
$\|\omega\| = n$	⇒	$n+2$
$\|\omega\| \geq n$	⇒	$n+1$
$\|\omega\| > n$	⇒	$n+2$

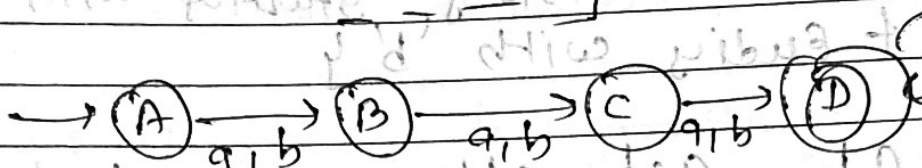
L = { set of all string length < 2 }

= { $\wedge$, a, b }



L = { set of all string of length > 2 }

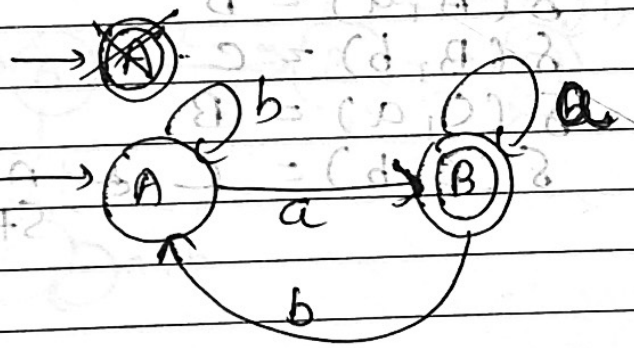
= { aaa, aab, aba, abb, baa, bab, bba, bbb ---- }

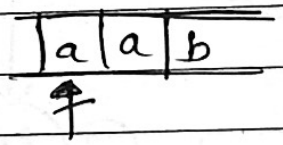


Example :- 6:

L = { set of all string ~~starting~~ ending with a }

= { a, aa, ba, aba, baa, aaa, aba a ---- }





$\delta(A, a) = B$,

$\delta(B, a) = B$

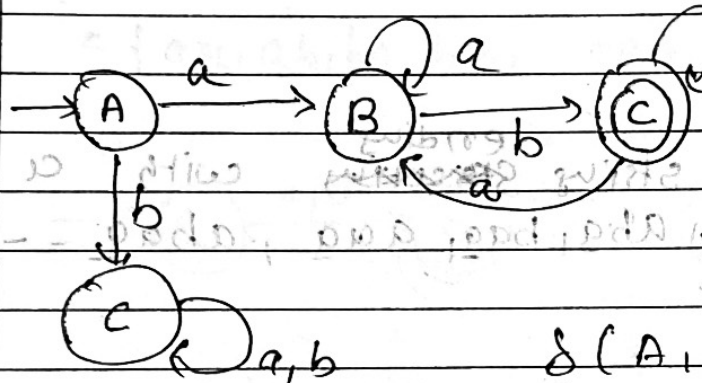
$\delta(B, b) = A$ ← not a final state not accepted.

DOMS | Page No.
Date / /

★ **Example : 7**

L = { set of all string starting with 'a' & ending with 'b' }

= { ab, aab, abb, aaab, abab ---- }



a | b | a | b

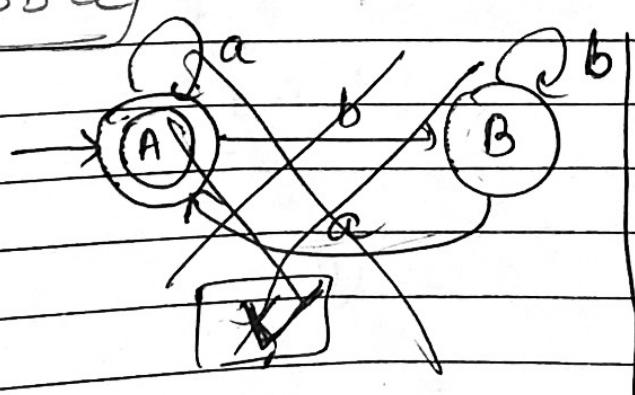
$\delta(A, a) = B$
$\delta(B, b) = C$
$\delta(C, a) = B$
$\delta(B, b) = C$ ← Accepting State

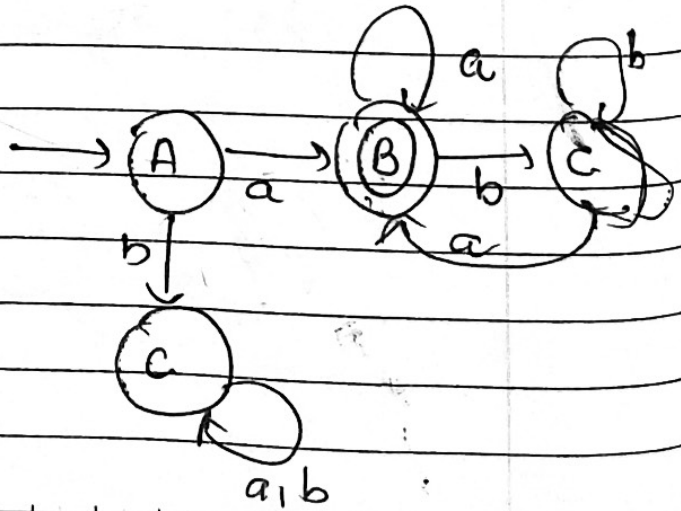
★ **Example : 8**

L = { set of all string starting & ending with 'a' }

= { a, aa, aba, abba ---- }

abbba





It is not necessary that you have only one final state

DOMS | Page No. | Date / /

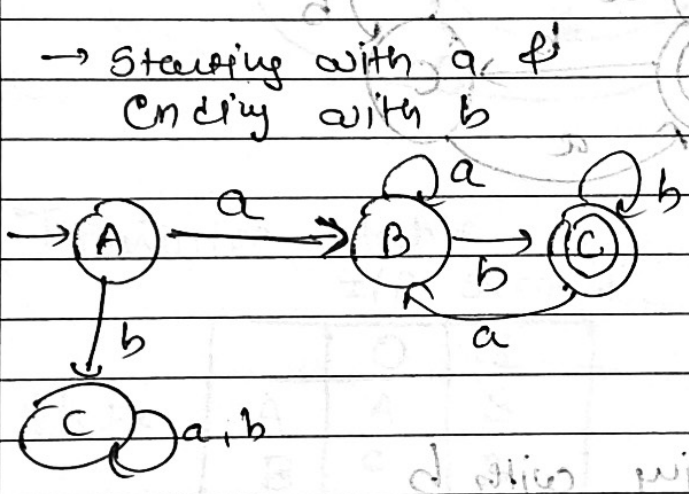
Example : 9

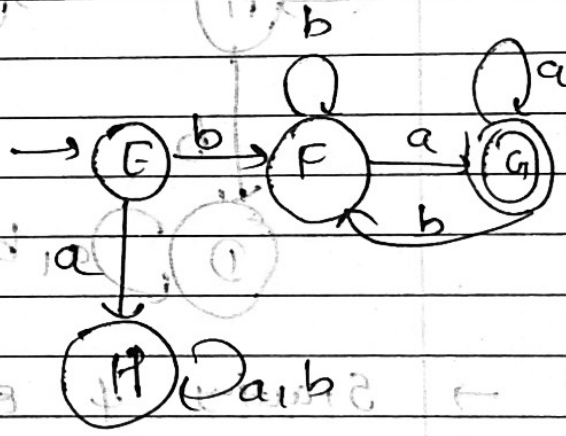
L = { set of all string starting & ending with different alphabets }

= { a ___ b, b ___ a, -- }
anything, anything

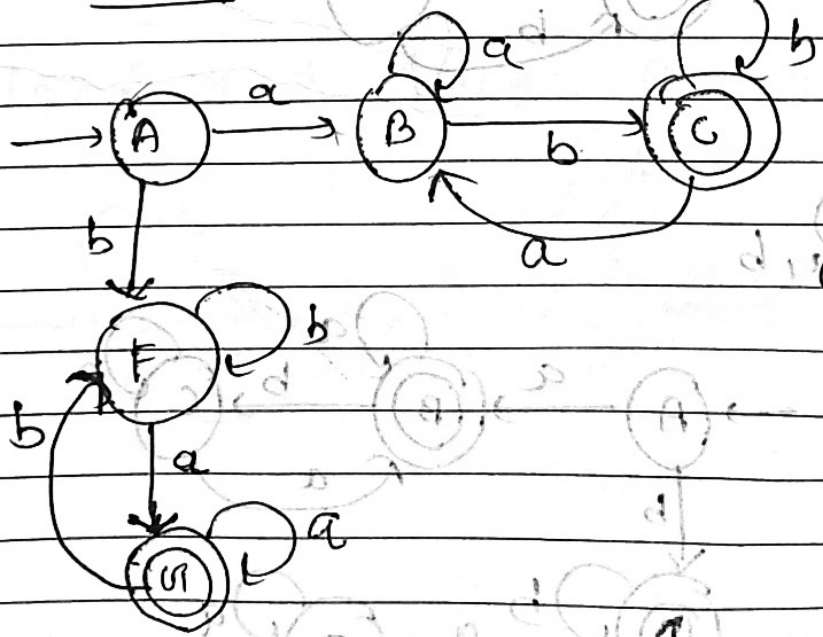
= { ab, aab, abb -- }
{ ba, baa, bba -- }

→ Starting with a & Ending with b





Combine



Combine initial state

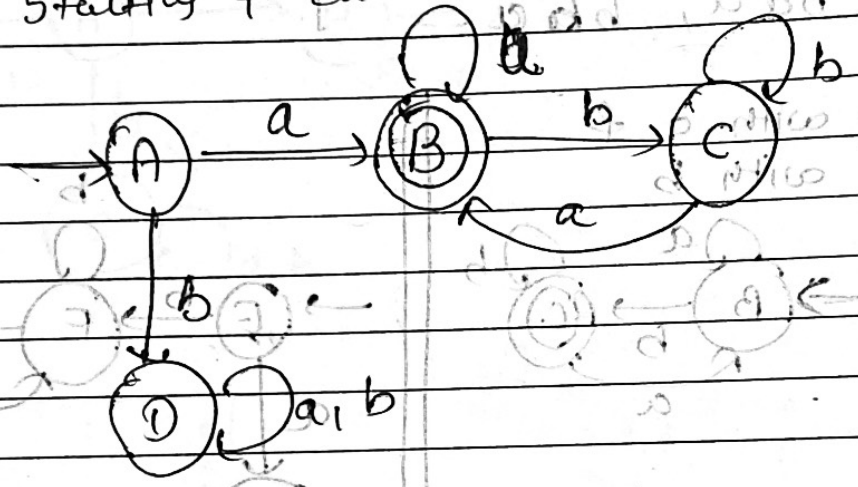
DOMS | Page No.
Date / /

Example : 10

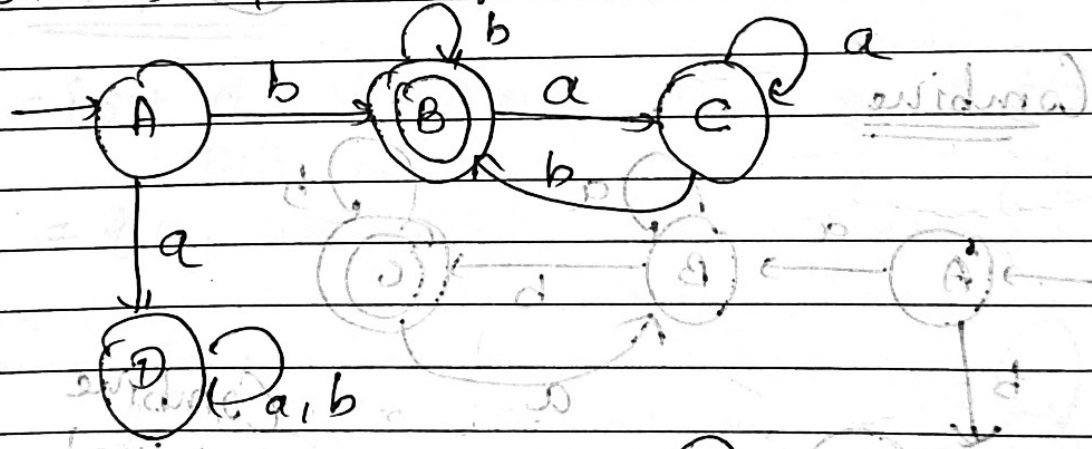
L7 = { Set of all string starting & Ending with same alphabets }

= { a, aa, aba, abba, abbba ... }
b, bb, bab, baab ... }

→ Starting & Ending with a

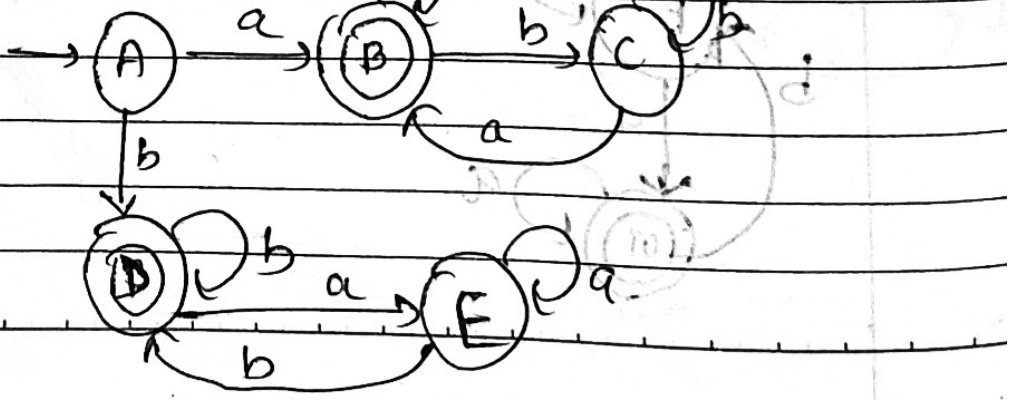


→ Starting & ending with b



→ Combine

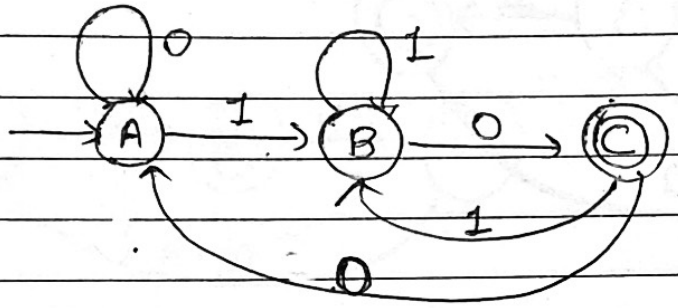
Combine Initial state



Example: II

L_8 = { Set of string ending with 10 }

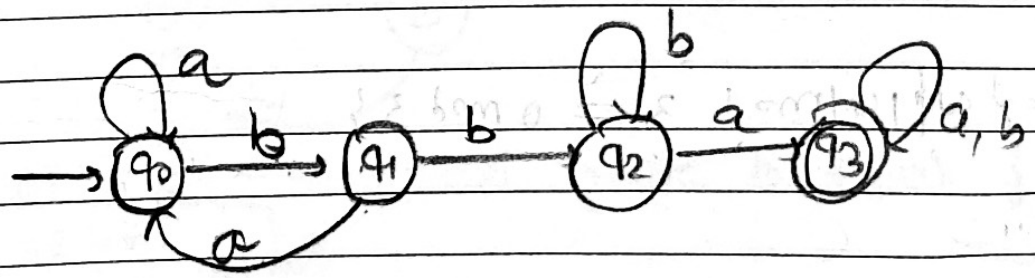
= { 10, 010, 110, 0010, 0110 ... }



Transition table

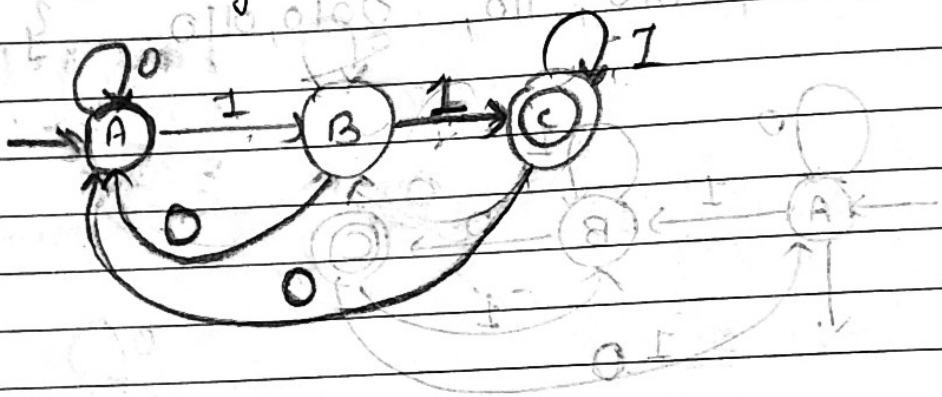
State	I/P symbol 0	1
A	A	B
B	C	B
C	A	B

L_9 = { set of string containing a substring bba }



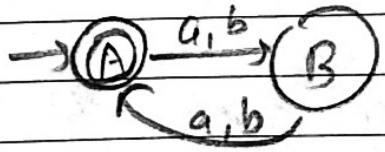
DOMS | Page No.
Date / /

* L_{10} = { string ending with 11 }



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

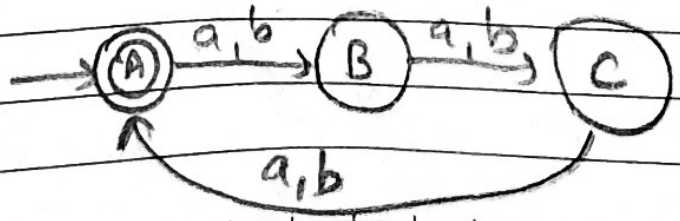
* $L_{11} = \{ w \mid |w| \bmod 2 \cong 0 \bmod 2 \}$
= { even length string }



* $L_{12} = \{ w \mid |w| \bmod 3 \cong 0 \bmod 3 \}$
= { length of string is multiple 3 } - 0, 3, 6, 9, 12

aaabbb

not accepted



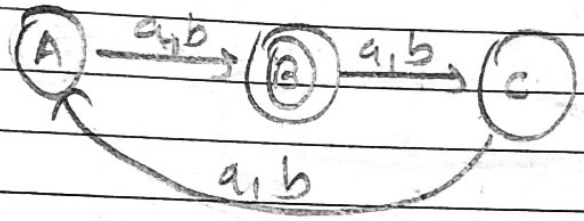
6 mod 3 = 0
7 mod 3 = 1
8 mod 3 = 2

9 mod 3 = 0
10 mod 3 = 1
11 mod 3 = 2

DOMS | Page No.
Date / /

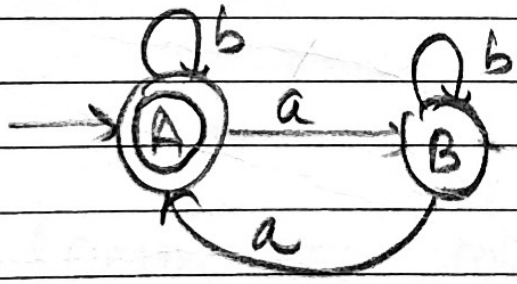
$L_{13} = \{ w \mid |w| \bmod 3 \cong 1 \bmod 3 \}$

$1 \bmod 3 = 1$
$4 \bmod 3 = 1$
$7 \bmod 3 = 1$



$L_{14} = \{ w \mid |N_a(w)| \bmod 2 \cong 0 \bmod 2 \}$
$= \{$ no of a are even $\}$
$= \{ \lambda, b, bb, bbbb \ldots, aa, aab, aaa$
$aba \ldots \}$

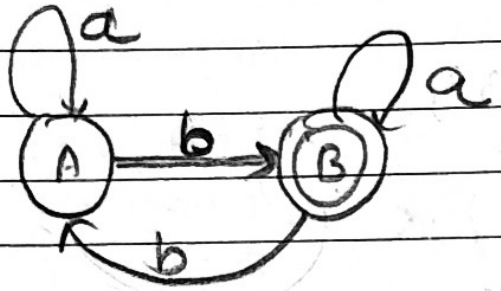
0 a's
2 a's
4, 6, 8 ...



DOMS | Page No.
Date / /

★ $L_5 = \{ w \mid |N_b(w)| \bmod 2 \cong 1 \bmod 2 \}$
$= \{$ odd no of b's $\}$

1 mod 2 = 1
3 mod 2 = 1
5 mod 2 = 1
7 mod 2 = 1



★ $L_{16} = \{ w \mid |N_a(w)| \bmod 2 = 0 \bmod 2 \ \& \ |N_b(w)| \bmod 2 = 0 \bmod 2 \}$

$= \{$ no. of a & b are even $\}$

DOMS | Page No.
Date / /

Ⓟ $L_{17} = \{ w \mid |N_a(w)| \bmod 2 \cong 1 \bmod 2 \quad \& \quad |N_b(w)| \bmod 2 \cong 0 \bmod 2 \}$

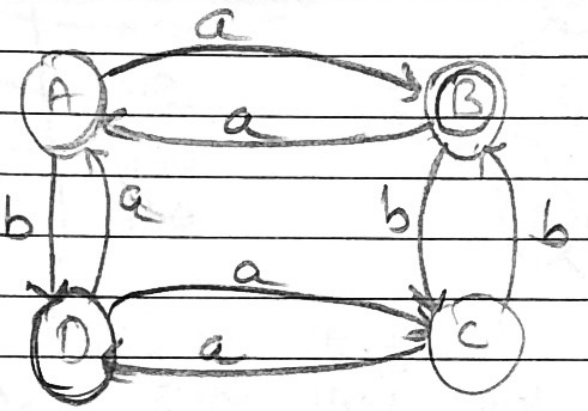
= { odd no of a's & even no of b's }

A = even a's even b's

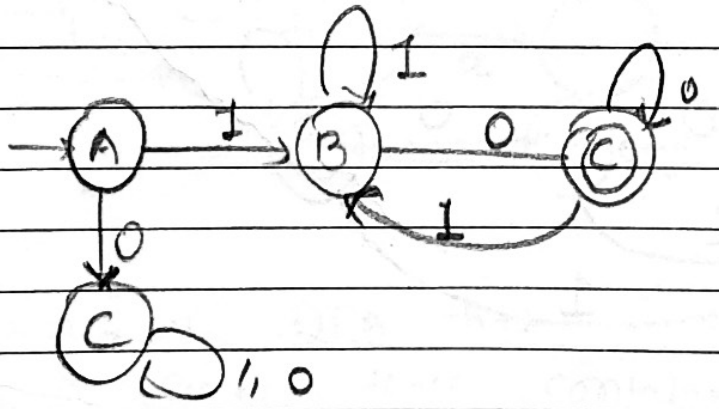
B = odd a's even b's

C = Odd a's odd b's

D = even a's odd b's

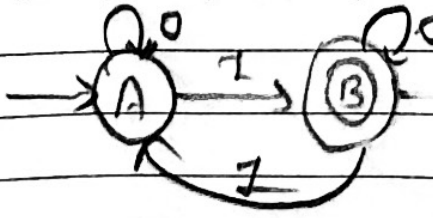


☆ Design DFA which accepts only those string which starts with 1 & ends with zero.



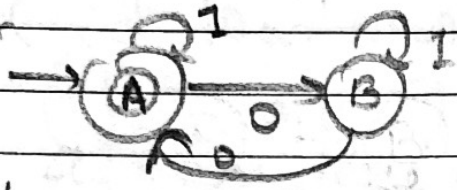
☆ Design DFA which accepts odd number 1's & any number of zero's

= { 1, 10, 010, 0010 ... }

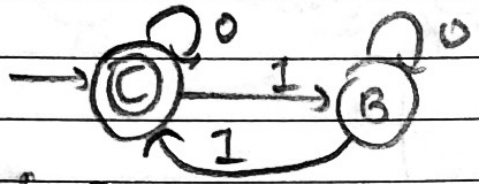


☆ Even no. of one's & even no of zero's
$= \{\wedge, 0011, 1100, 1010, 1001 \ldots\}$

Even no 0's



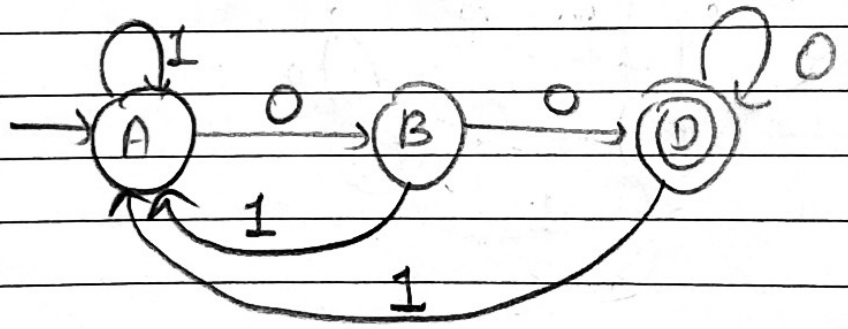
Even not 1's



# of 0's	# of 1's
even	even
even	odd
odd	even
odd	odd

$\{A, B\} \times \{C, D\}$
$= \{AC, AD, BC, BD\}$

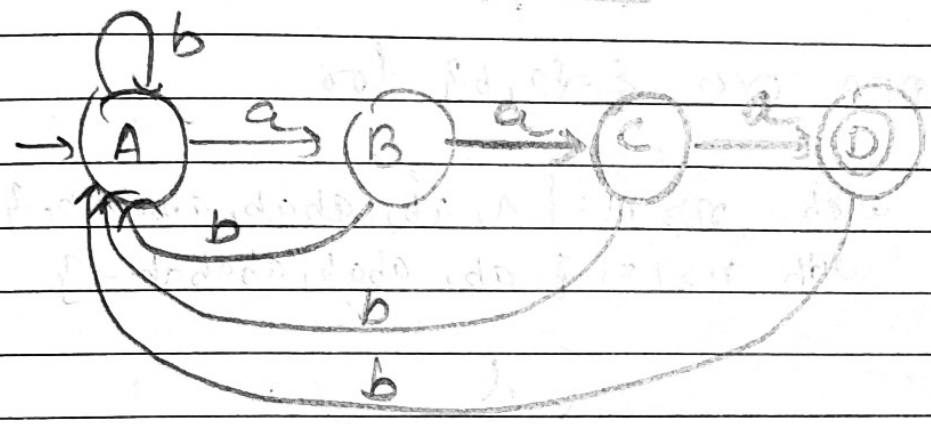
☆ Design DFA to accept the string that alway ends with 00.



☆ Design DFA to accept L, where L = {string in which 'a' alway appear in tripled} over the set $\Sigma = \{a, b\}$

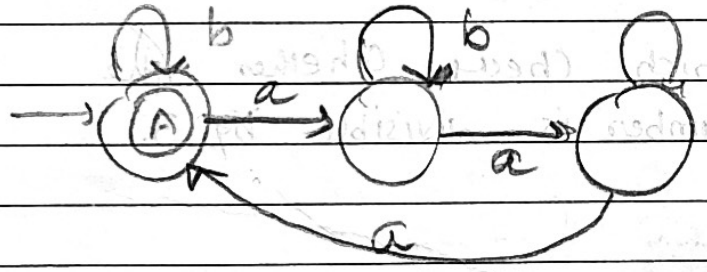
DOMS | Page No.
Date / /

Collective a

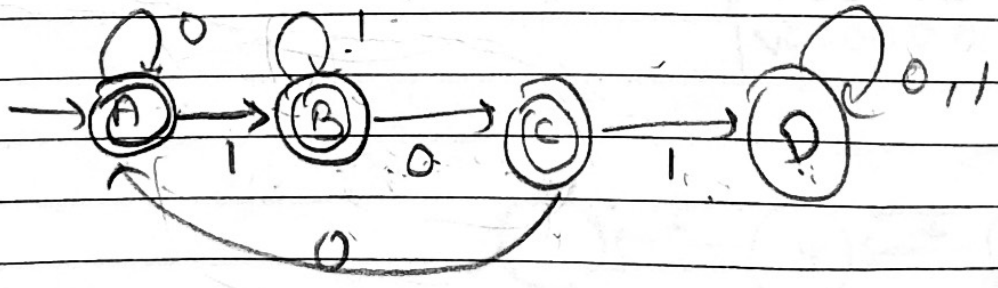


Q3 → Design DFA to accept L where all the string in L are such that total numbers of a's in them are divisible by 3.

⇒



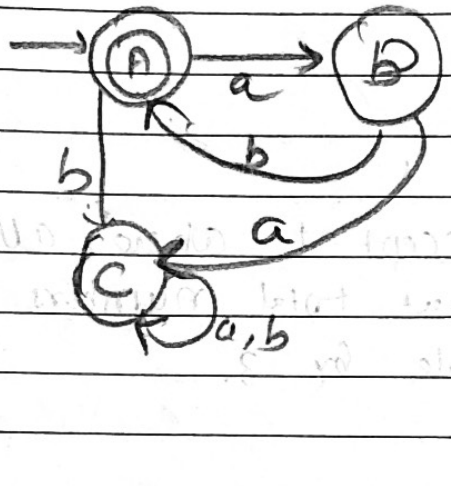
Construct a DFA that accepts all string on {0,1} except those containing the substring 101.

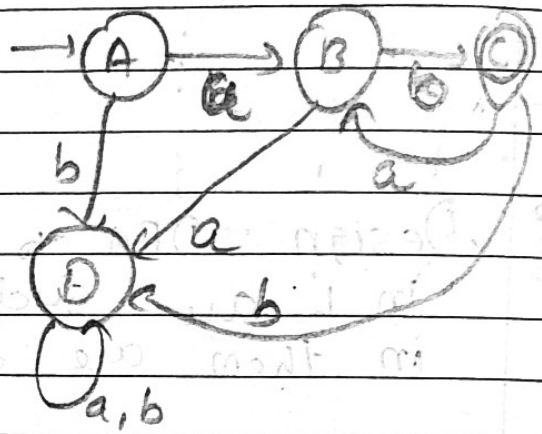


DOMS | Page No.
Date / /

★ Desig DFA over $\Sigma = \{a, b\}$ for

$\{ab\}^n$ with $n \geq 0 = \{\wedge, ab, abab, ababab\text{--}\}$
$\{ab\}^n$ with $n \geq 1 = \{ab, abab, ababab\text{--}\}$



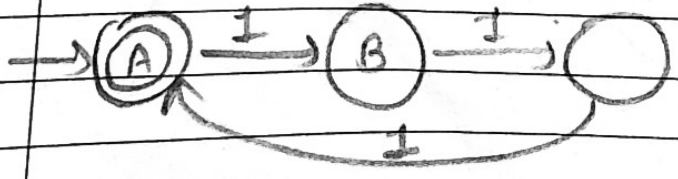


★ Design DFA which checks whether the given unary number is divisible by 3.

(unary) → 0 → zero
00 → two
000 → three $\Sigma = \{1\}$

& = 0 X
00X
000✓
0000X
00000X
000000✓

Replace 0 by 1



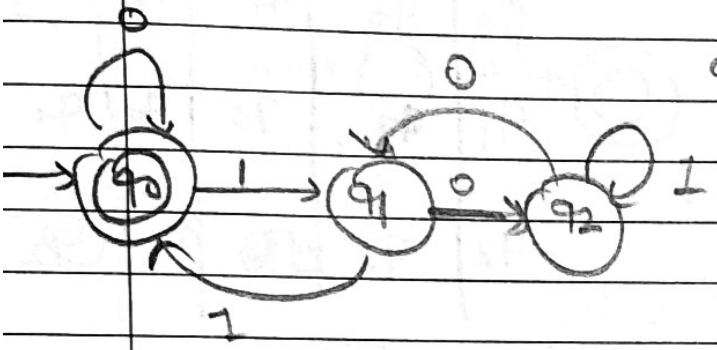
DOMS | Page No.
Date / /

* Design DFA which checks whether the given binary number is divisible by 3.

binary = 0, 1

{ 0, 00, 000, --- 000 ---
01, 1x, 11, 011, 00011
10x, 010x, 000010x, 100x,
---- }

divisible by 3 i get only 3 remainder 0, 1, 2 (3 state)



	0	1	
q0	q0	q1	write sequence
q1	q2	q0	
q2	q1	q2	

* Design DFA which checks whether the given Ternary number is divisible by 3.

Ternary = 0, 1, 2

	0	1	2
*q0	q0	q1	q2
q1	q0	q1	q2
q2	q0	q1	q2

DOMS | Page No.
Date / /

* Design DFA which checks whether the given quinary ~~binary~~ no. is divisible by 3.

NS → 0, 1, 2, 3

	0	1	2	3
q_0	q_0	q_1	q_2	q_0
q_1	q_1	q_2	q_0	q_1
q_2	q_2	q_0	q_1	q_2

*

Whether the given octal number is divisible by 6.

octal → 0, - - 7

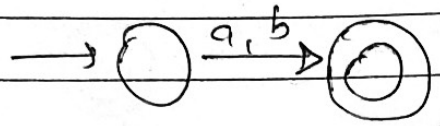
Decimal - 0 - - - 9

	0	1	2	3	4	6	7
q_0	q_0	q_1	q_2	q_3	q_4	q_5	q_0
q_1							
q_2							
q_3							
q_4							
q_5							

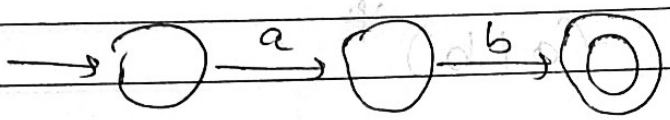
DOMS | Page No.
Date / /

Regular Expression to DFA

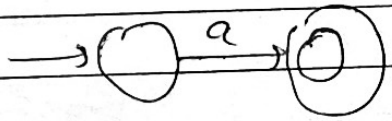
1. a+b



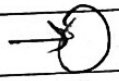
2. ab



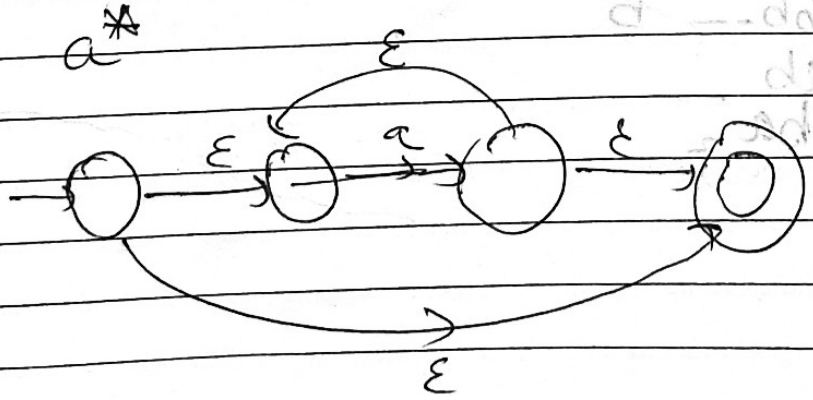
3. a



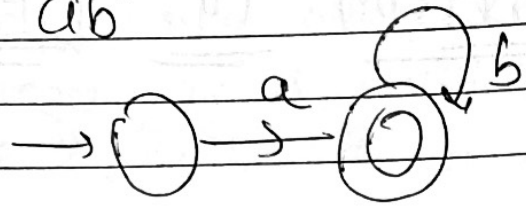
4. ϕ



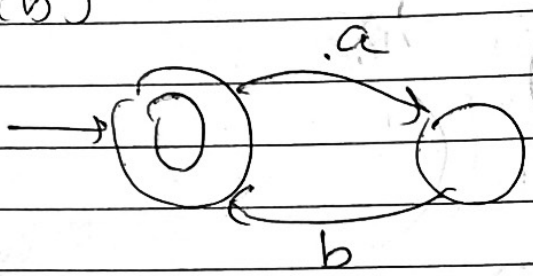
5. a^*



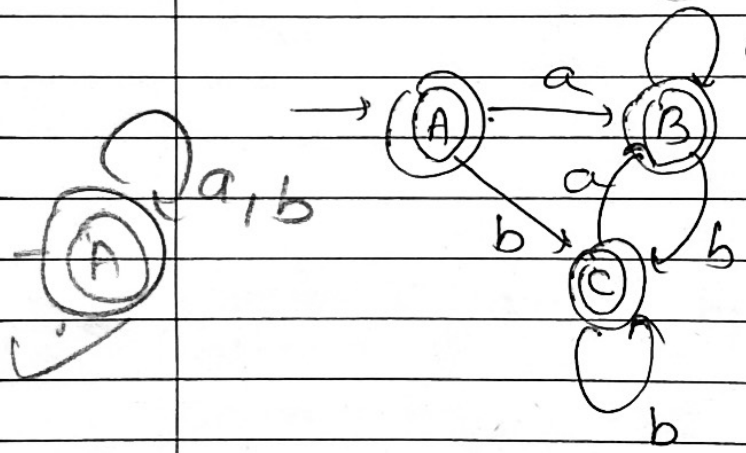
① ab^*

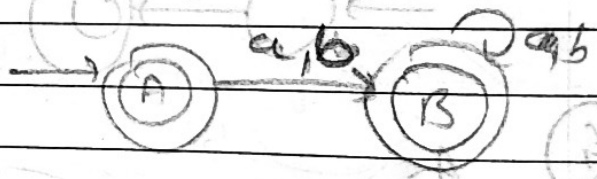


② $(ab)^*$



③ ~~$(ab+ba)^*$~~ $(a+b)^*$





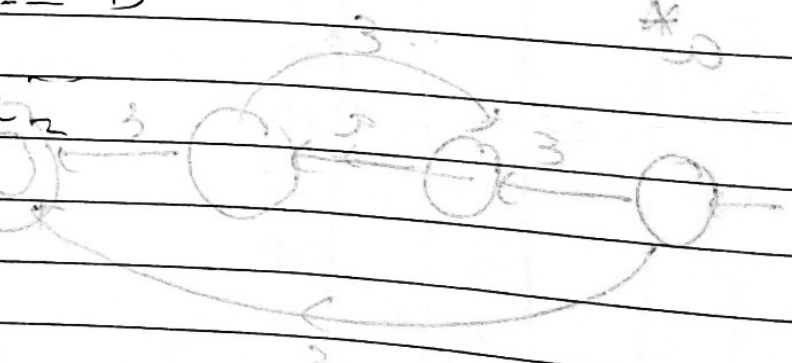
Ex

aaa -- a

bbb -- b

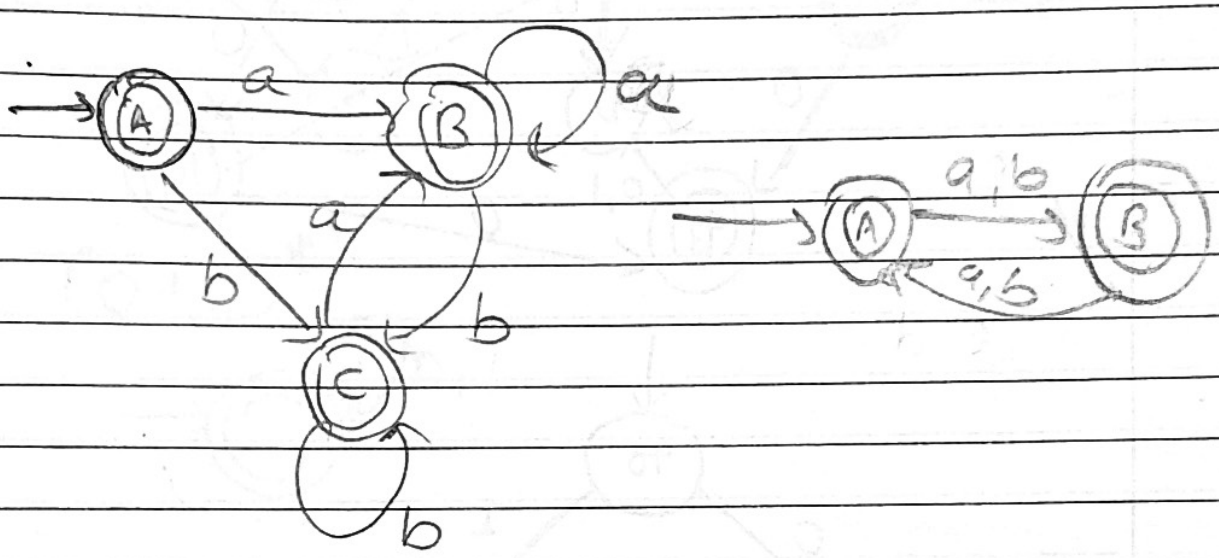
ab

ba

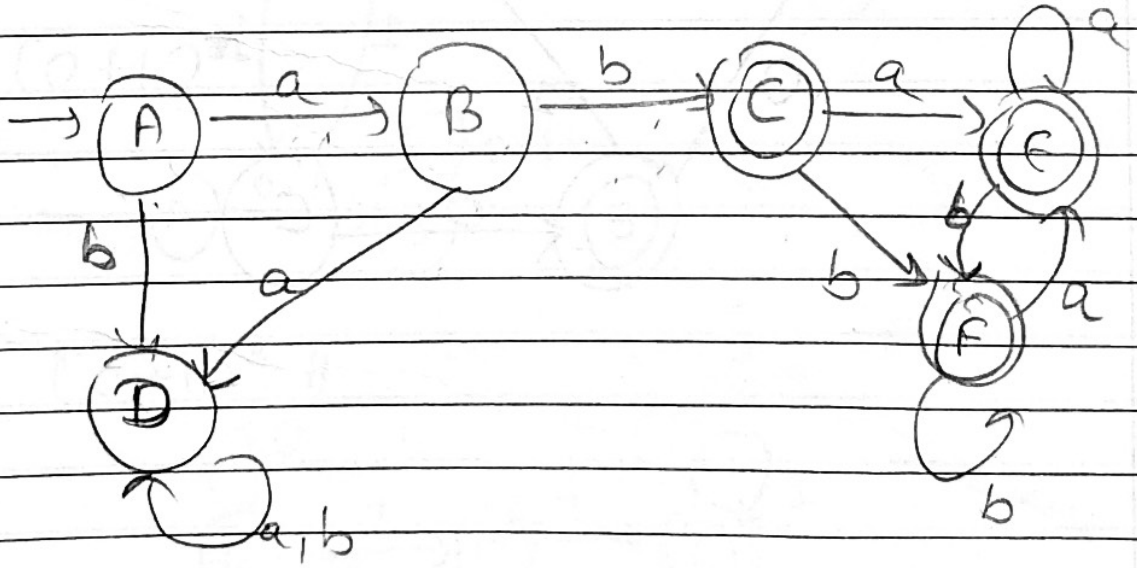


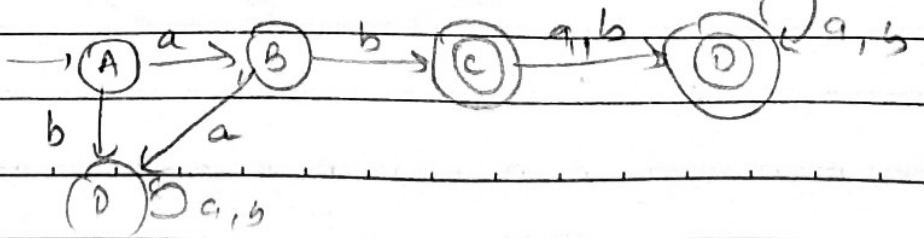
(4) $(a^* + b^*)^*$

Ex :- {∧, aaa, bbb, aabb, ...}

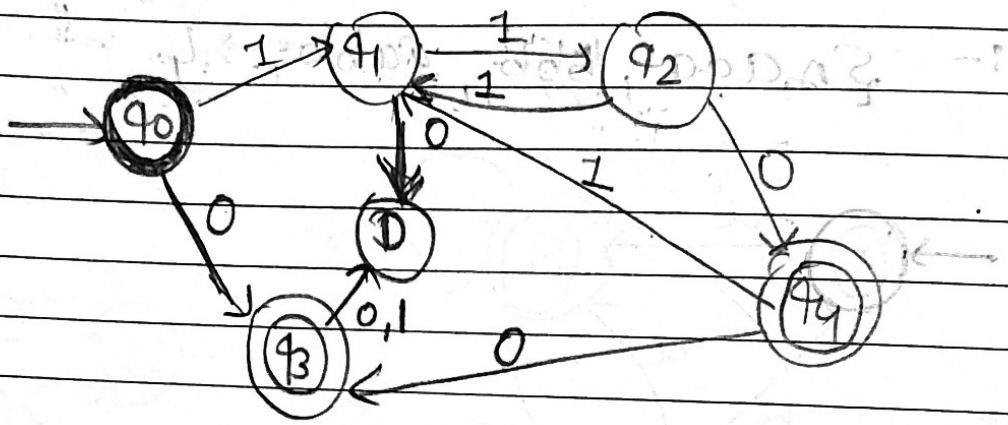


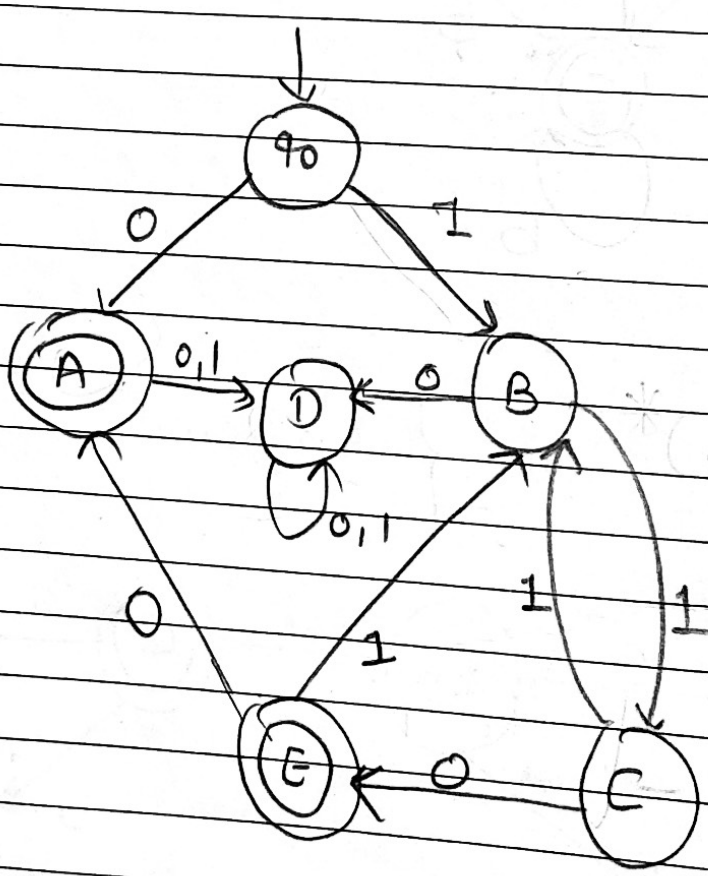
(5) $ab(a|b)^*$ ξ = { abaaa



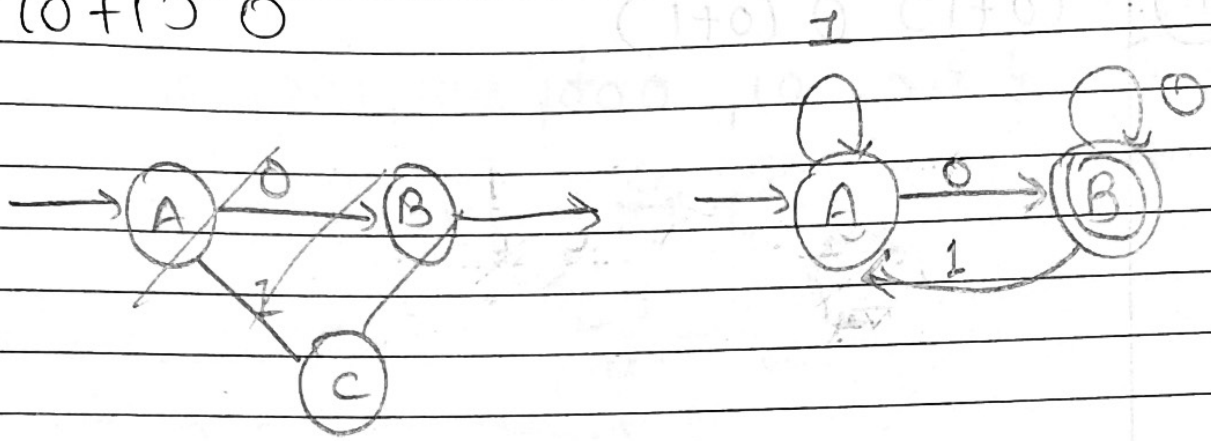


6. $R = (11 + 110)^* 0$

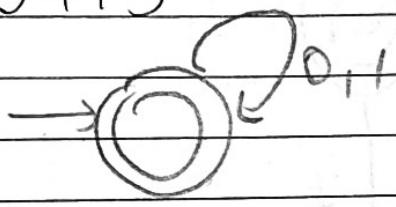




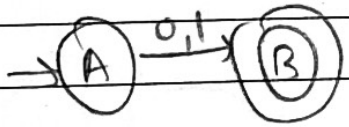
7. $(0+1)^*0$



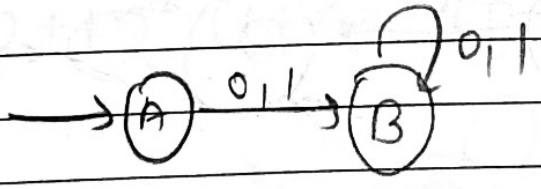
8. $(0+1)^*$



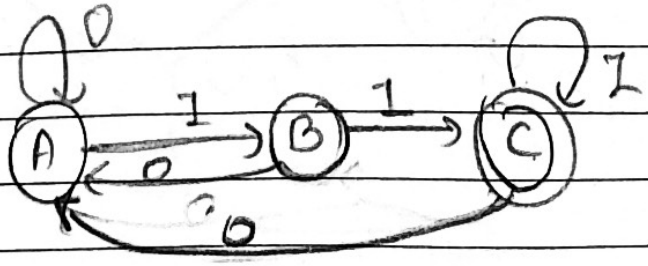
9. $(0+1)$



10. $(0+1)^*$

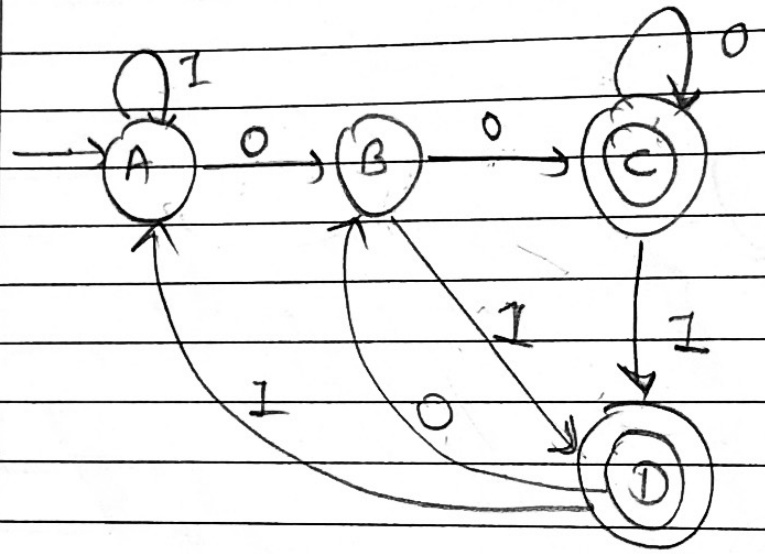


11. $(0+1)^*11$

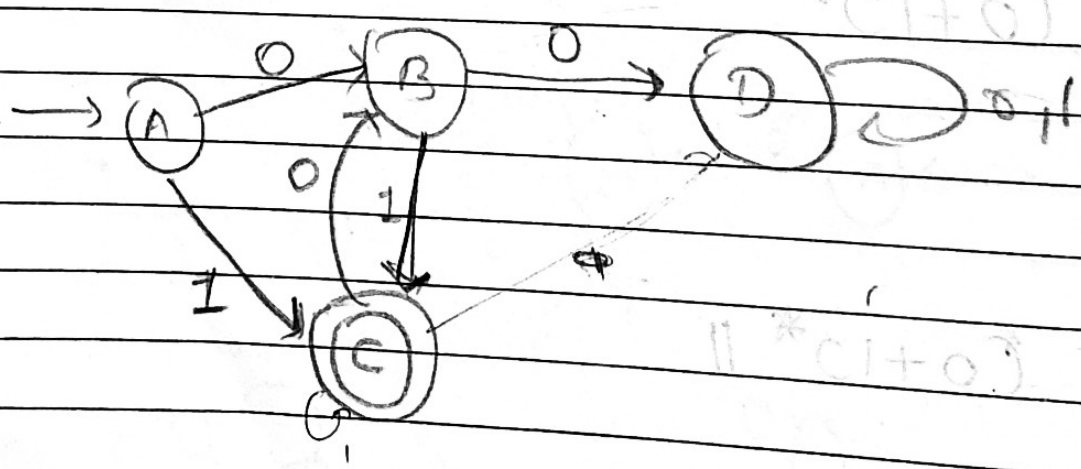


(11) $(0+1)^* 0 (0+1)$

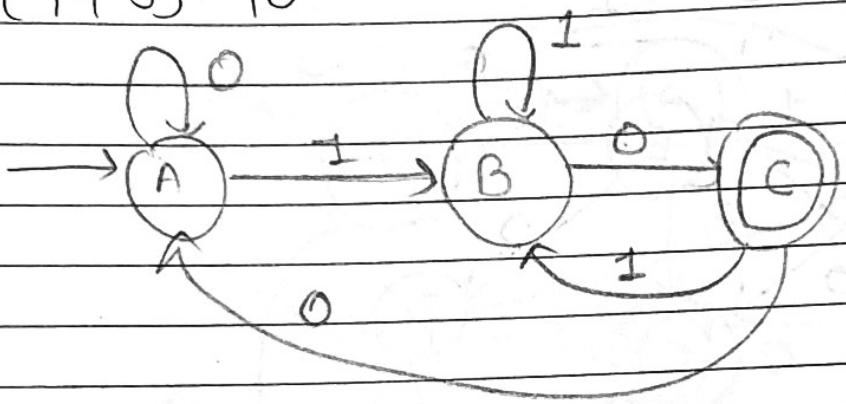
= {00, 01, 000, 001, 100, 101



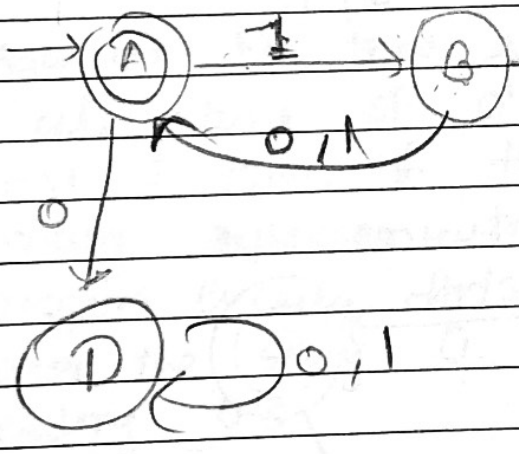
(12) $(1+01)^+$



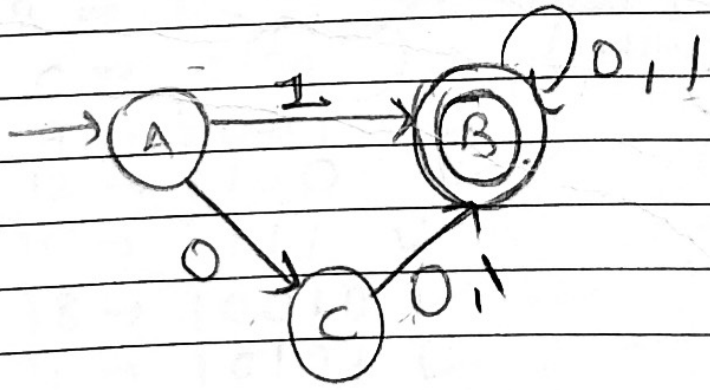
(13) $(1+0)^* 10$



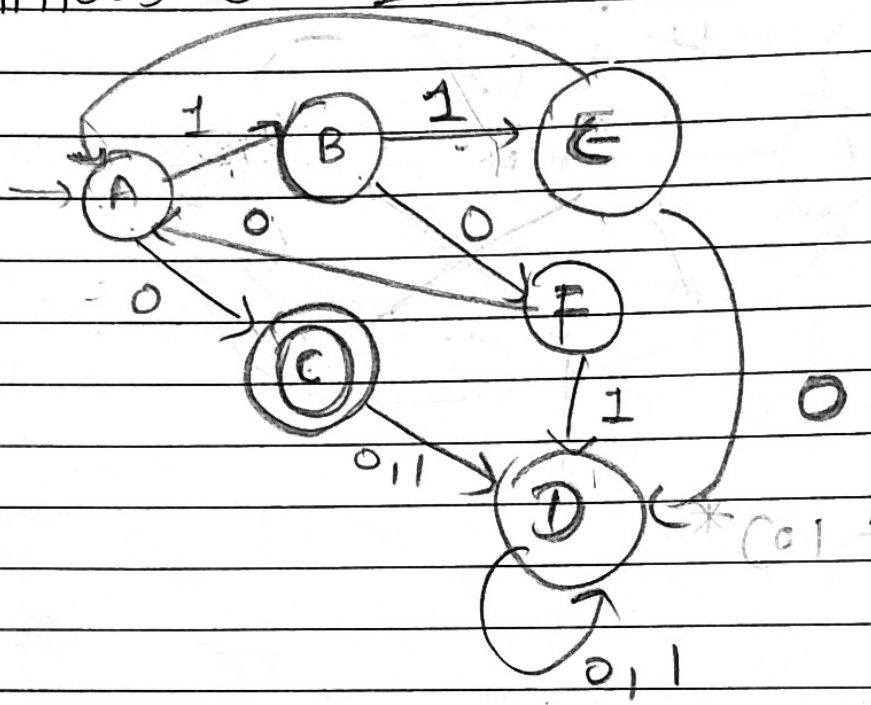
(14) $(11+10)^*$



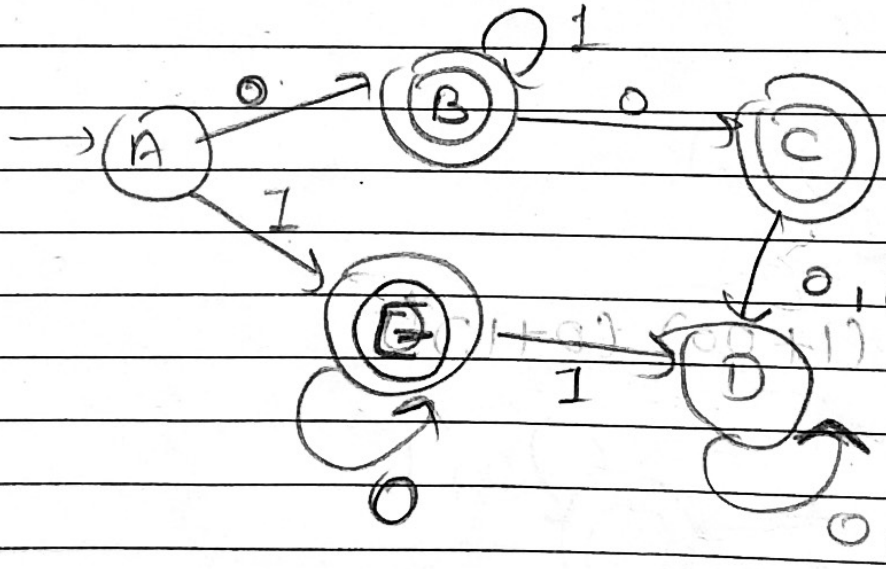
(15) $(0+1)^*(1+00)(0+1)^*$



(16) $(111+100)^* 01$



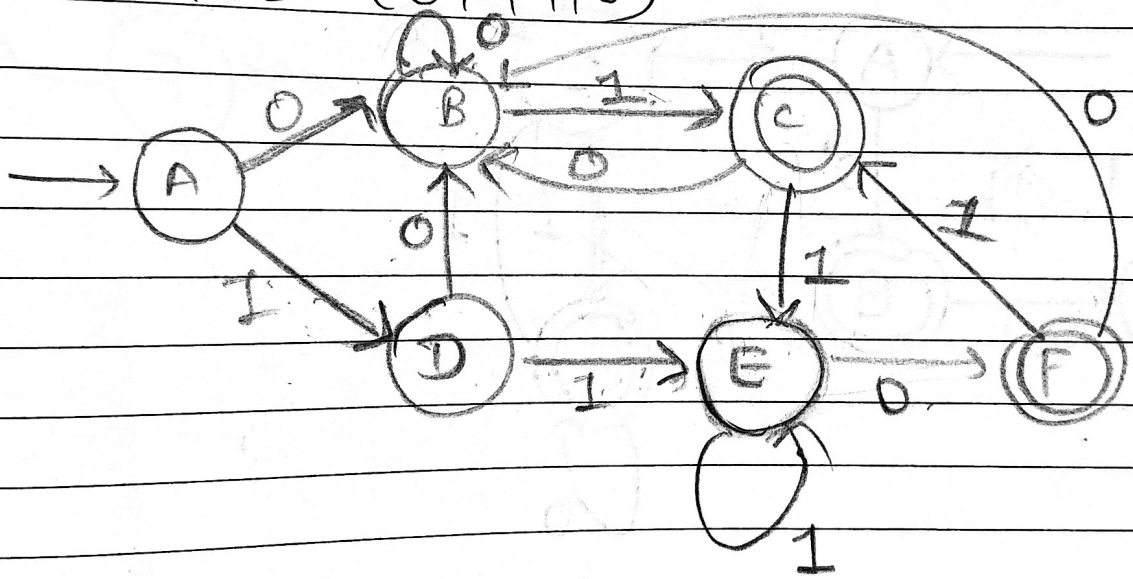
(17) $0+10^*+01^*0$



000 170

DOMS | Page No.
Date / /

(12) $(0+1)^*(01+110)$



DOMS | Page No.
Date / /

Example

(1) Union

Example

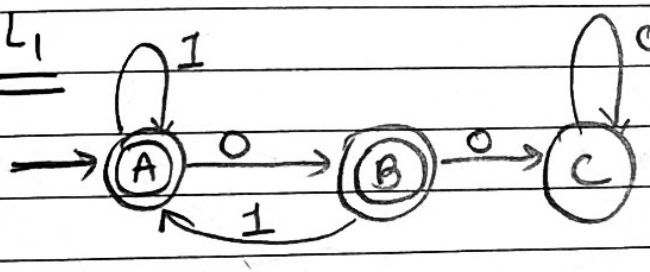
$\Sigma = \{0, 1\}$ [If L_1 & L_2 are regular then L is also regular $[L = L_1 \cup L_2]$]

Suppose

$L_1 = \{ x \mid 100$ is not a substring of $x \}$

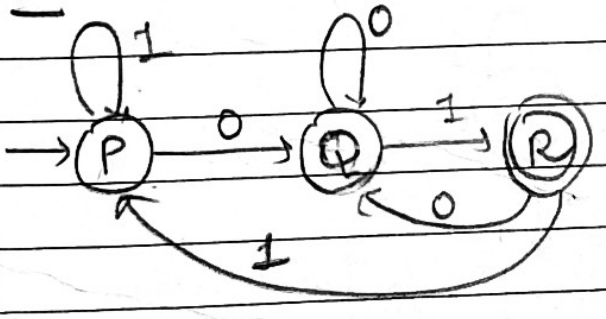
$L_2 = \{ x \mid x$ ends with $01 \}$

L_1

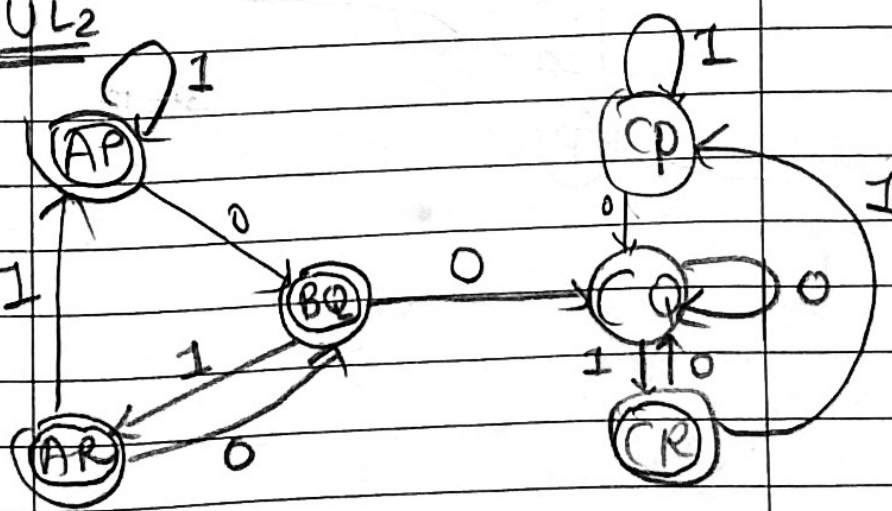


State	0	1
AP	BQ	AP
BQ	CQ	AR
CQ	CQ	CR
AR	BQ	AP
CR	CQ	CP
CP	CQ	CP

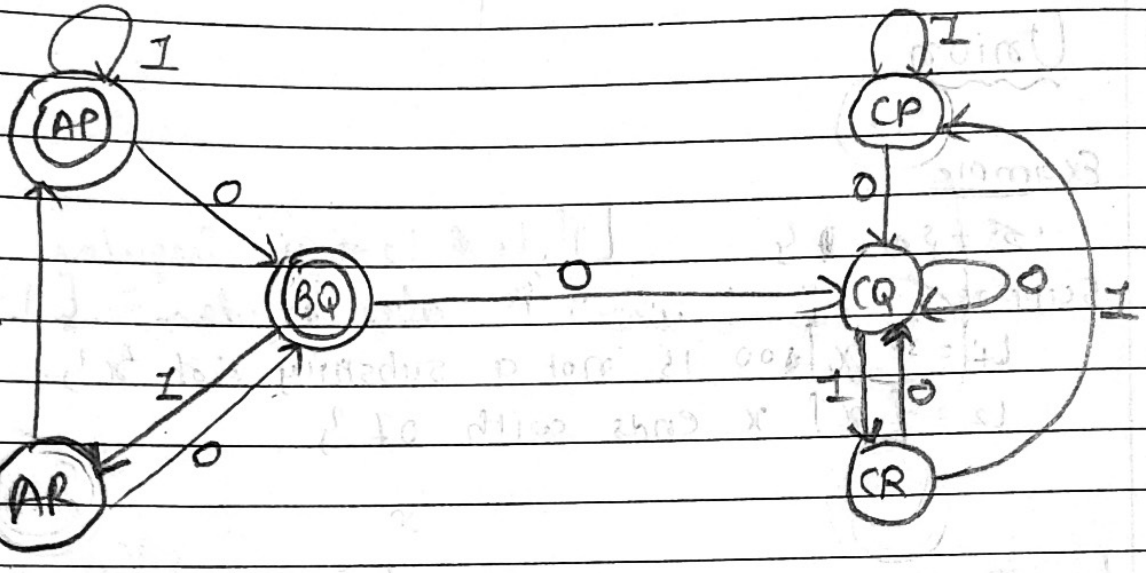
L_2



$L_1 \cup L_2$

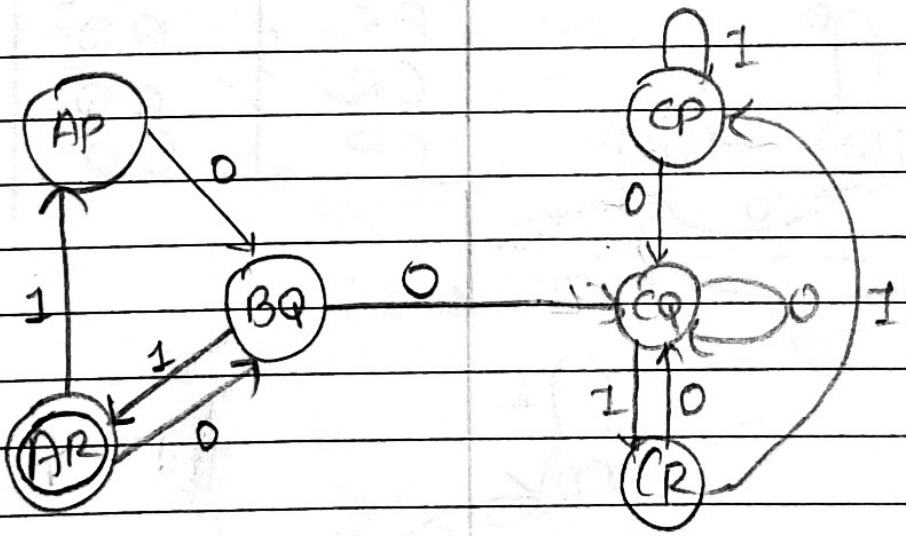


(2) $L_1 - L_2$



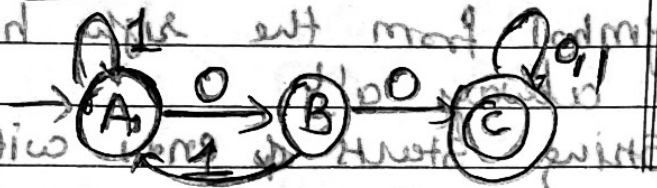
(3) $L_1 \cap L_2$

Only AR will be the final state

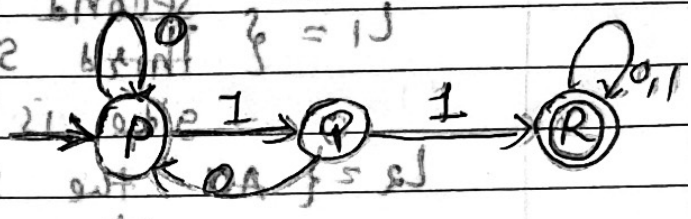


(2) $\Sigma = \{a, b\}$

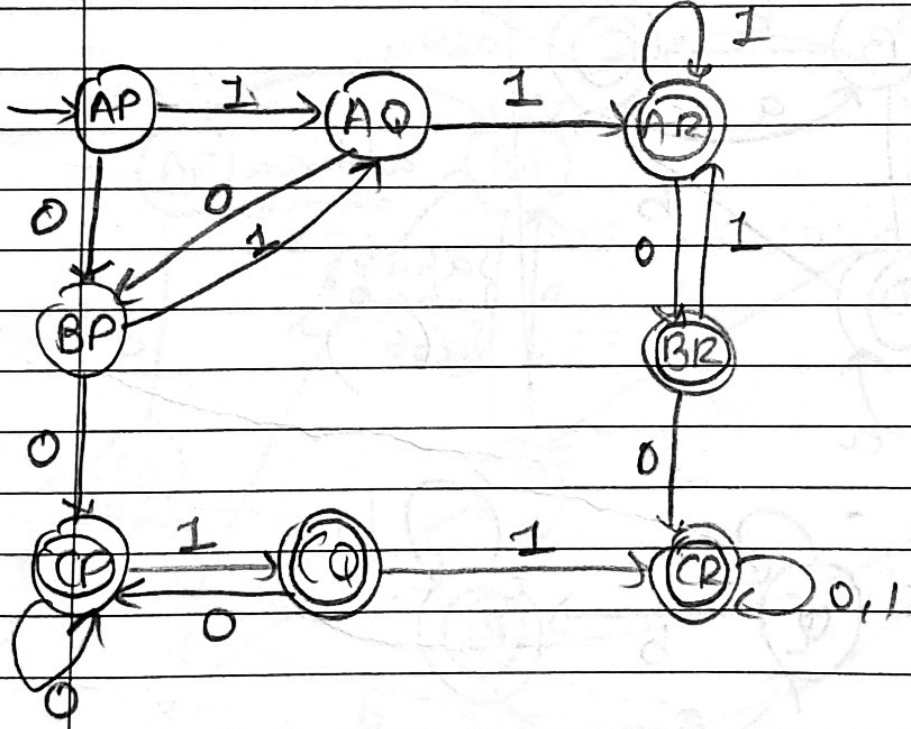
L_1



L_2



$L_1 \cup L_2$



State	0	1
AP	BP	AQ
AQ	BP	AR
AR	BR	AR
BP	CP	AQ
BR	CR	AR
CP	CP	CQ
CR	CR	CR
CQ	CP	CR

$L_1 - L_2 \Rightarrow$ CP, CQ

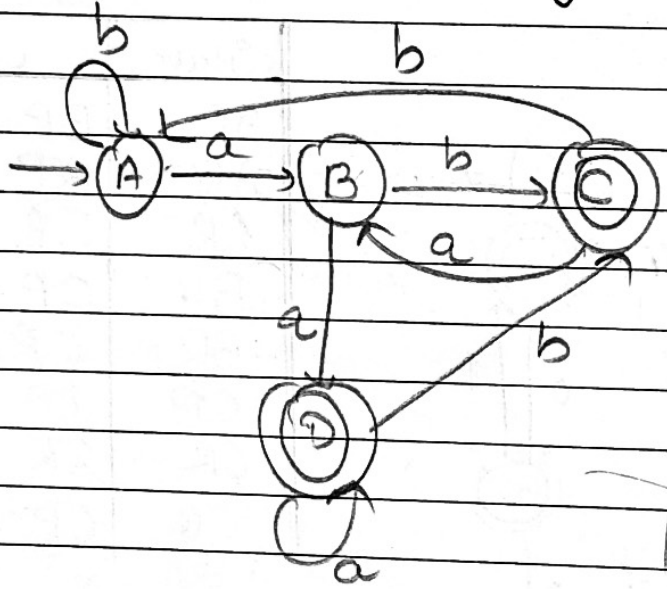
$L_1 \cap L_2 \Rightarrow$ CR

(3) Construct DFA for following languages, that accept all the string of a's & b's where.

L_1 = { second Third symbol from the right hand side is always 'a'}

L_2 = { All the string starts & ends with a different symbol}

L_1

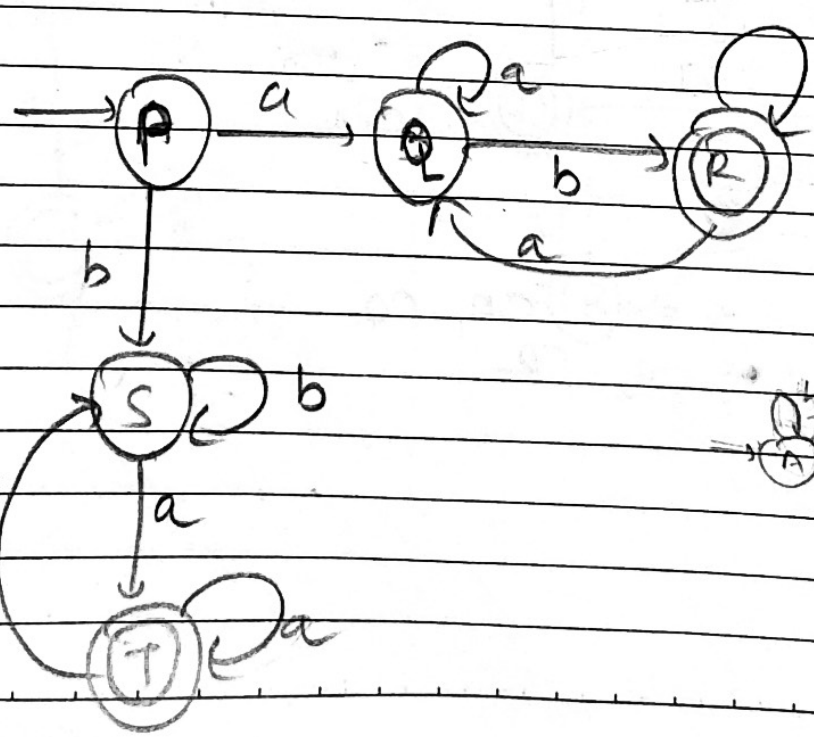


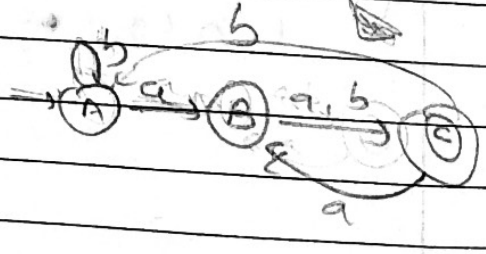
ab, aa, abb, aabaa, ababa,

~~abbaa~~
~~abbab~~

~~{abab~~
~~{abaab~~
{abb

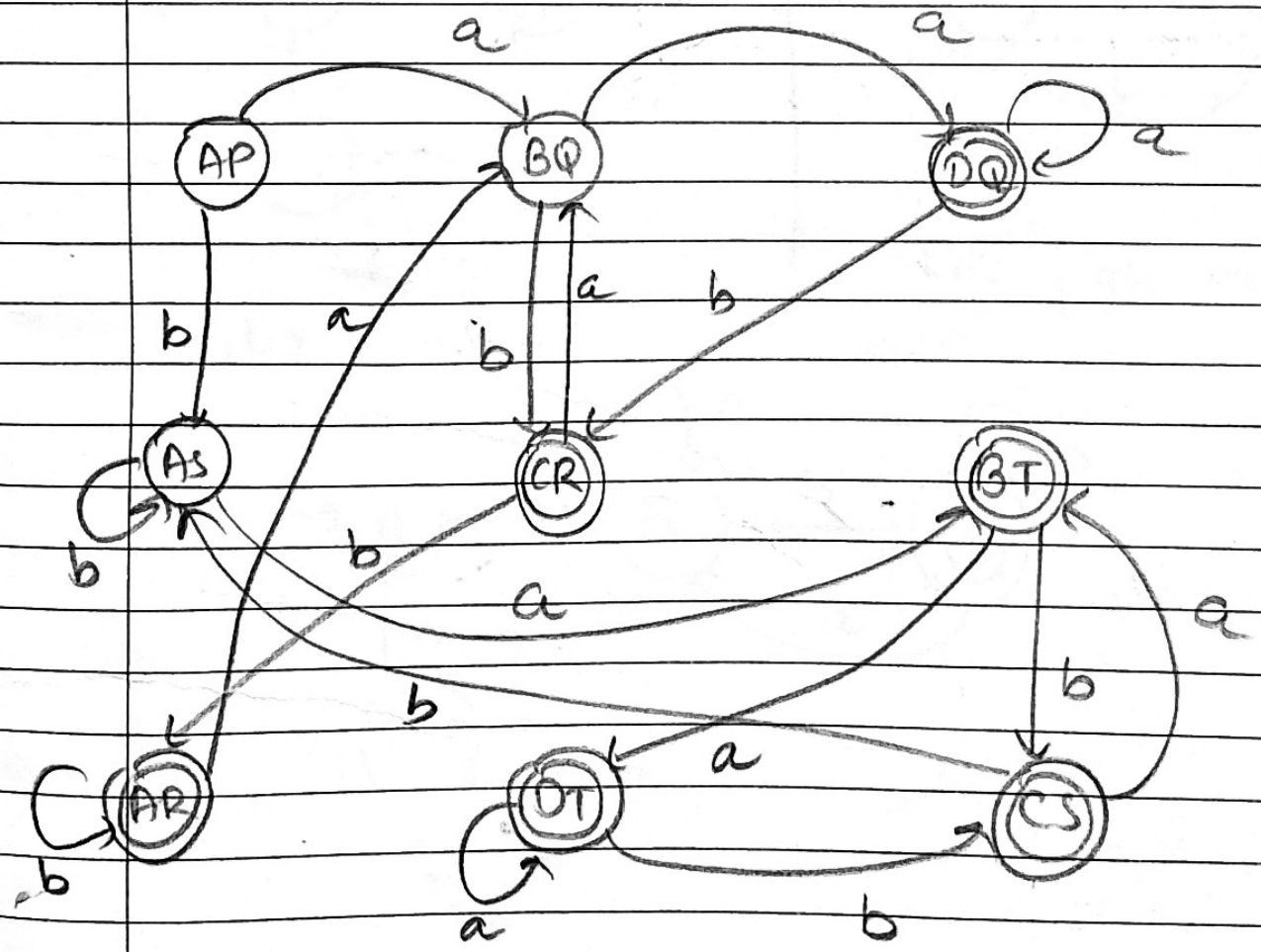
L_2





State	a	b
AP	BQ	AS
BQ	DQ	CR
AS	BT	AS
DQ	DQ	CR
CR	BQ	AR
BT	DT	CS
AR	BQ	AR
DT	DT	CS
CS	BT	AS

4 U L2



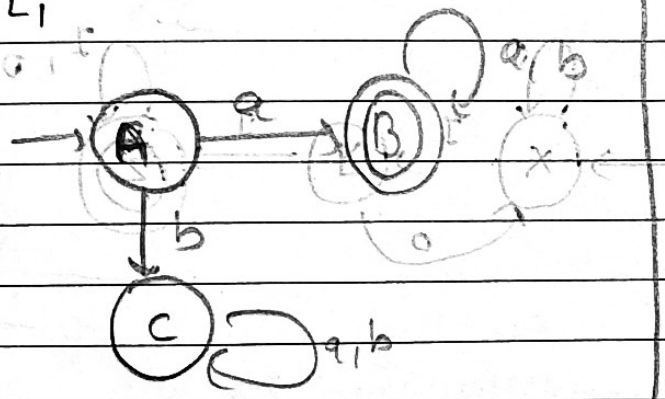
DOMS | Page No.
Date / /

Concatenation

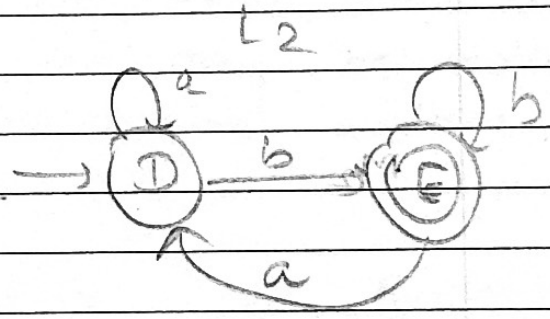
(4) L_1 = {all strings start with a, }
L_2 = {all strings end with b}
$L_3 = L_1 L_2$

$L_1 = \{ a, aa, ab, aba, aab, aaa -- \}$
$L_2 = \{ b, ab, bb, aab, abb -- \}$

L_1

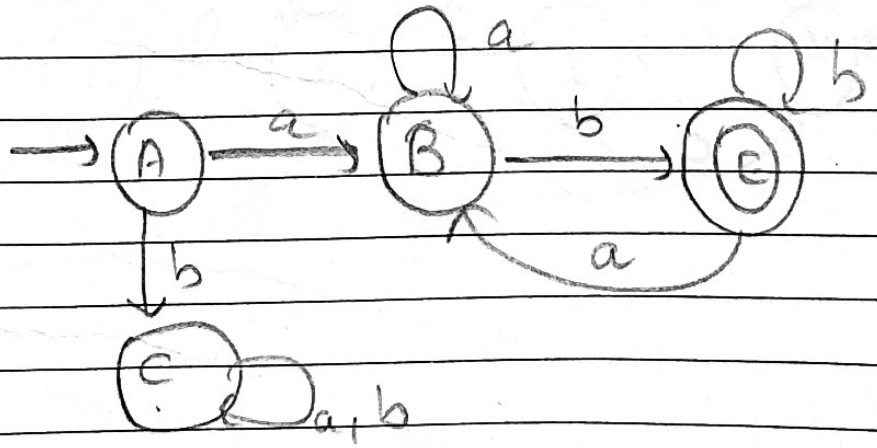


L_2



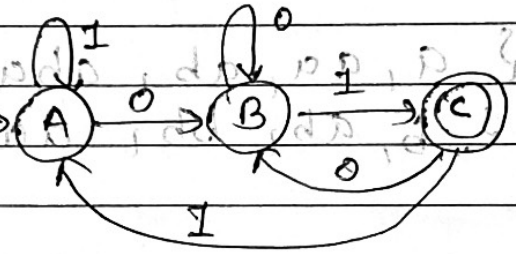
$L_1 L_2 = \{ ab, aab, abb$ ---

$L_1 L_2$

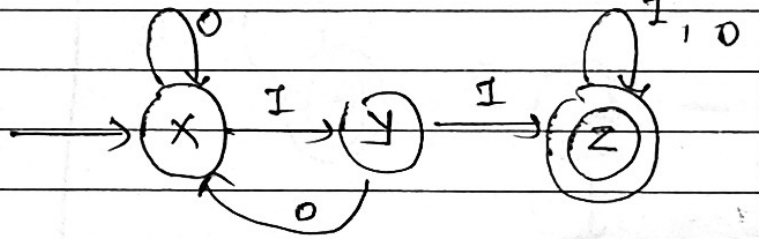


→ Let M_1 & M_2 be the DFA's shown below recognizing the language L_1 & L_2 respectively. Draw the DFA which recognizes language $L_3 = L_1 L_2$.

M_1



M_2



L_1 = { all string ending with 01 }
L_2 = { all string containing [redacted] as a substring }