classfellow

Date : / / 2022

Page No.

TOC Unit-2

Pumping Lemma

Pumping Lemma is used to prove that a language is NOT REGULAR.

Regular Languages are the languages that can be designed using Finite State Machines. The languages that cannot be designed using Finite State Machines are non-regular lang.

⇒ Pumping Lemma cannot be used to prove that a language is regular.

⇒ If A is a regular language then A has a pumping length 'P' such that any string 'S' where $|S| \geq P$ may be divided into 3 parts

↓ length of string S

$S = xyz$

Such that the following conditions must be true:-

1. $xy^iz \in A$ for every $i \geq 0$
2. $|y| > 0$
3. $|xy| \leq P$

Steps to prove that a language is not regular using Pumping Lemma:-

[We prove using contradiction]

→ Assume that A is Regular
→ It has to have a pumping length (say P)
→ All strings longer than P can be pumped $|S| \geq P$
→ Now find a string 's' in A such that $|S| \geq P$
→ Divide S into xyz
→ Show that $xy^iz \notin A$ for some i
→ Then consider all ways that S can be

divided into x y z.

→ Show that none of these can satisfy all 3 pumping conditions at the same time.

→ S cannot be pumped = = Contradiction.

Ques Using Pumping Lemma prove that the language $A = \{a^n b^n \mid n \geq 0\}$ is not regular.

Theory: In this no. of a and b should be same and for that the finite state machine need to keep the count of a inorder to have same no. of b but finite state machine is not capable of storing the count therefore this cannot be designed using a finite state machine and hence it is not a regular language.

Proof: Assume that A is a regular language

Pumping length = P

$S = a^P b^P$

Divide S into x y z

Let P = 7

$S = a^7 b^7$ = aaaaaaabbbbbbb

Case 1: Y is in 'a' part.

aaa aaaa abbbbbbb
x y z

$xy^i z$ Let i = 2

$xy^2 z$ = aaaaaaaaaabbbbbbb $xy^i z \notin A$

No. of a ≠ No. of b

11 ≠ 7

classfellow

$|y| = 4 > 0$

$|xy| \leq 7 \Rightarrow 6 \leq 7$

Case 2: y is in 'b' part

aaaaaaab bbbbbbb
x y z

Let i = 2

xy^2z = aaaaaaabbbbbbbbbbb $\quad xy^iz \notin A$

No. of a ≠ No. of b

7 ≠ 11

$|y| = 4 > 0$

$|xy| \leq 7 \Rightarrow 12 \leq 7$ Not satisfied

Case 3: y is in 'a' and 'b' part

aaaaaaab bbbbbbb
x y z

Let i = 2

xy^2z = aaaaaaaabbaabbbbbbb

↓

String does not follow pattern

$xy^iz \notin A$

$|y| = 4 > 0$

$|xy| \leq 7 \Rightarrow 9 \leq 7$ Not satisfied

⇒ This language cannot satisfy all the conditions at the same time in any case

∴ Not a Regular Language

Ques. Using Pumping Lemma prove that the language $A = \{yy \mid y \in \{0,1\}^*\}$ is NOT a Regular Language

Proof:-

Assume that A is regular

Then it must have a pumping length P

$S = 0^P 1 0^P 1$

let $P = 7$

$S = \underbrace{0000000}_{x}\underbrace{0}_{y}\underbrace{10000001}_{z}$

xy^iz

let $i = 2$

$S = 00000000010000001$

$xy^2z \notin A$

~~$|xy| \leq P$~~

let $i = 3$

$S = 000000000000010000001$

$xy^3z \notin A$

Also, $|xy| \leq P$ Not satisfied

$\Rightarrow$ A is not Regular

Date : / / 2022

Page No.

Closure Properties of Regular Languages

Closure Properties refer to the ways in which regular languages can be combined and transformed while still remaining regular.

1. Union
If $L1$ and $L2$ are regular languages
$\Rightarrow L1 \cup L2$ is also a regular language.

2. Intersection
If $L1$ and $L2$ are regular languages
$\Rightarrow L1 \cap L2$ are also regular.

3 Concatenation
If $L1$ and $L2$ are regular languages
$\Rightarrow L1.L2$ is also regular.

4. Kleene Star
If $L1$ is a regular language
$\Rightarrow L^*$ is also regular

5. Complementation
If L is a regular language
$\Rightarrow L'$ (all strings not in L) is also regular.

6. Transpose
If L is a regular language
$\Rightarrow L^T$ is also regular.

Minimization of DFA

Minimization of DFA is required to obtain the minimal version of any DFA which consists of the minimum number of states posible.

Two states are said to be equivalent if

$\delta(A, x) \rightarrow F$ and $\delta(B, x) \rightarrow F$

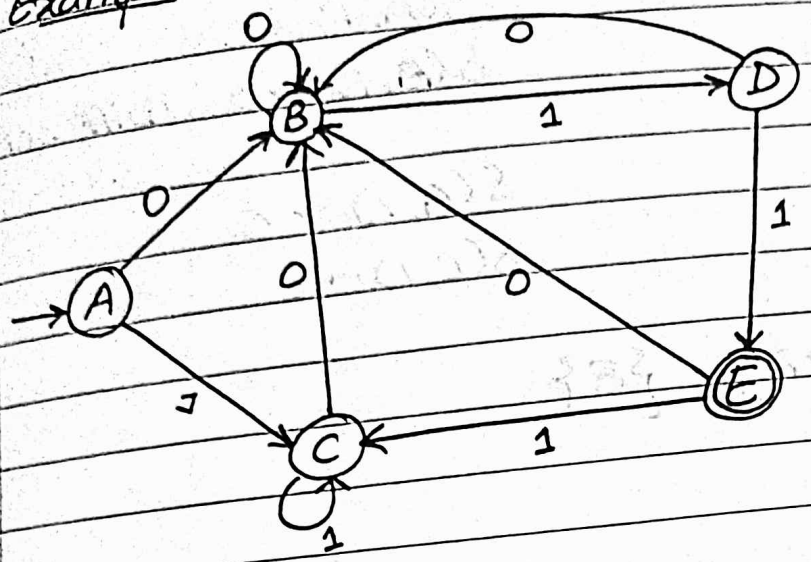
OR

$\delta(A, x) \nrightarrow F$ and $\delta(B, x) \nrightarrow F$ $\quad x \rightarrow \Sigma$

If $|x| = 0$ then A and B are said to be 0 equi
If $|x| = 1$ then A and B are said to be 1 equi
If $|x| = 2$ then A and B are said to be 2 equi
:
If $|x| = n$ then A and B are said to be n equi

classfellow

Example :- [Minimization of DFA]



	0	1
→ A	B	C
B	B	D
C	B	C
D	B	E
(E)	B	C

0 Equivalence

{A, B, C, D} {E}

1 Equivalence

$\delta(A,0) \rightarrow B$

$\delta(B,0) \rightarrow B$

$\delta(A,1) \rightarrow C$

$\delta(B,1) \rightarrow D$

C, D belong to same set

⇒ A, B are equivalent

$\delta(A,0) \rightarrow B$

$\delta(C,0) \rightarrow C$

$\delta(A,1) \rightarrow C$

$\delta(C,1) \rightarrow C$

$\delta(A,0) \rightarrow B$

$\delta(D,0) \rightarrow B$

$\delta(A,1) \rightarrow C$

$\delta(D,1) \rightarrow E$ → Different set

{A, B, C} {D} {E}

2 Equivalence

$\delta(A,0) \rightarrow B$ $\quad$ $\delta(A,1) \rightarrow C$

$\delta(B,0) \rightarrow B$ $\quad$ $\delta(B,1) \rightarrow D$ [Diff set]

$\delta(A,0) \rightarrow B$ $\quad$ $\delta(A,1) \rightarrow C$

$\delta(C,0) \rightarrow B$ $\quad$ $\delta(C,1) \rightarrow C$

$\{A,C\}$ $\{B\}$ $\{D\}$ $\{E\}$

3 Equivalence

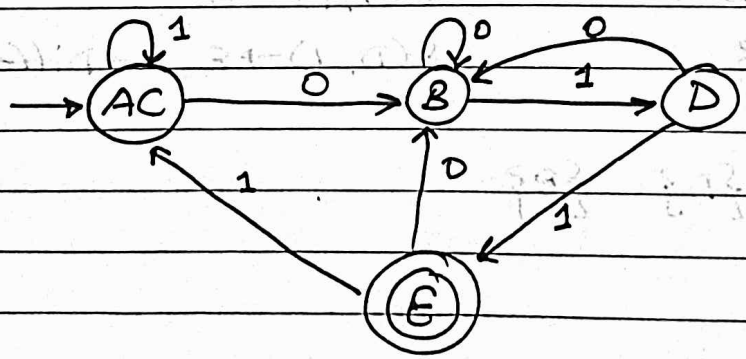
$\delta(A,0) \rightarrow B$ $\quad$ $\delta(A,1) \rightarrow C$

$\delta(C,0) \rightarrow B$ $\quad$ $\delta(C,1) \rightarrow C$

$\{A,C\}$ $\{B\}$ $\{D\}$ $\{E\}$

Transition Table

	0	1
→ AC	B	AC
B	B	D
D	B	E
(E)	B	AC



Date : / / 2022

Page No.

classfellow

Ques. Construct a minimum DFA equivalent to the DFA described by

	0	1
$\rightarrow q_0$	q_1	q_5
q_1	q_6	q_2
(q_2)	q_0	q_2
q_3	q_2	q_6
q_4	q_7	q_5
q_5	q_2	q_6
q_6	q_6	q_4
q_7	q_6	q_2

0 Equivalence

$\{q_0, q_1, q_3, q_4, q_5, q_6, q_7\}$ $\{q_2\}$

1 - Equivalence

$\delta(q_0, 0) \rightarrow q_1$

$\delta(q_4, 0) \rightarrow q_6$

$\delta(q_0, 1) \rightarrow q_5$

$\delta(q_1, 1) \rightarrow q_2$ [Diff set]

$\delta(q_0, 0) \rightarrow q_1$

$\delta(q_3, 0) \rightarrow q_2$ [Diff set]

$\delta(q_1, 0) \rightarrow q_6$

$\delta(q_3, 0) \rightarrow q_2$ [Diff set]

$\delta(q_3, 0) \rightarrow q_2$

$\delta(q_5, 0) \rightarrow q_2$

$\delta(q_3, 1) \rightarrow q_6$

$\delta(q_5, 1) \rightarrow q_6$

$\delta(q_1, 0) \rightarrow q_6$

$\delta(q_7, 0) \rightarrow q_6$

$\delta(q_1, 1) \rightarrow q_2$

$\delta(q_7, 1) \rightarrow q_2$

$\{q_0, q_4, q_6\}$ $\{q_1, q_7\}$ $\{q_3, q_5\}$ $\{q_2\}$

2- Equivalence

$\delta(q_0, 0) \to q_1$ $\quad$ $\delta(q_0, 1) \to q_5$

$\delta(q_4, 0) \to q_7$ $\quad$ $\delta(q_4, 1) \to q_5$

$\delta(q_0, 0) \to q_1$ $\quad$ $\delta(q_0, 1) \to q_5$

$\delta(q_6, 0) \to q_6$ $\quad$ $\delta(q_6, 1) \to q_4$ [Diff set]

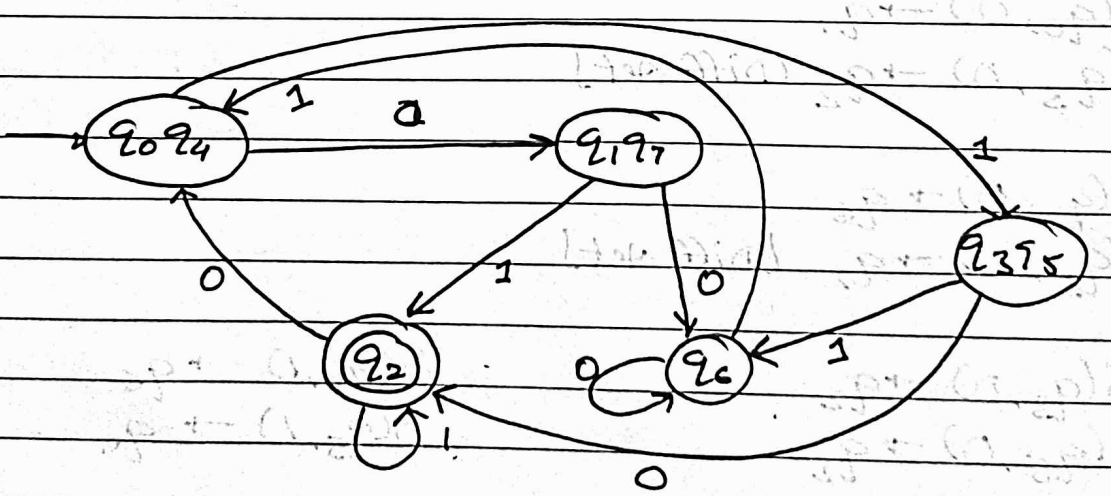
~~$\delta(q_4, 0) \to q_7$~~

~~$\delta(q_6, 0) \to q_5$~~

$\{q_0, q_4\}$ $\{q_6\}$ $\{q_1, q_7\}$ $\{q_3, q_5\}$ $\{q_2\}$

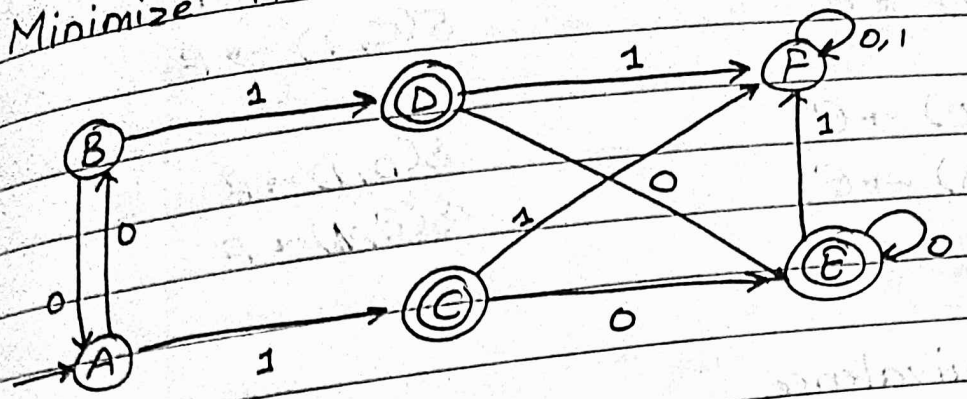
Transition Table

	0	1
$\to q_0q_4$	q_1q_7	q_3q_5
q_1q_7	q_6	q_2
(q_2)	q_0q_4	q_2
q_3q_5	q_2	q_6
q_6	q_6	q_0q_4



classfellow

Ques. Minimize the following DFA



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

0 - Equivalence

{A, B, F} {C, D, E}

1 - Equivalence

$\delta(A,0) \to B$

$\delta(B,0) \to A$

$\delta(A,1) \to C$

$\delta(B,1) \to D$ ~~[Diff set]~~

$\delta(A,0) \to B$

$\delta(F,0) \to F$

$\delta(A,1) \to C$

$\delta(F,1) \to F$ [Diff set]

{A, B} {F} {C, D, E}

$\delta(C,0) \to E$ $\quad$ $\delta(C,1) \to F$

$\delta(D,0) \to E$ $\quad$ $\delta(C,1) \to F$

$\delta(D,0) \to E$ $\quad$ $\delta(D,1) \to F$

$\delta(E,0) \to E$ $\quad$ $\delta(E,1) \to F$

2- Equivalence

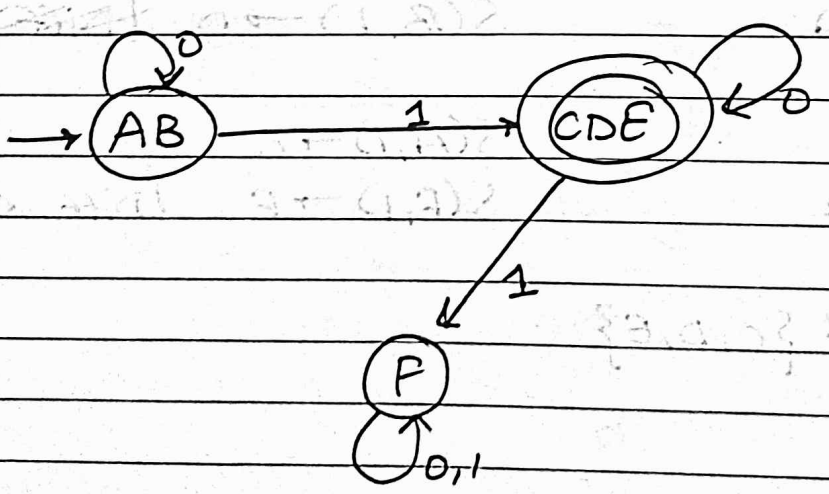
$\delta(A,0) \to B$ $\quad$ $\delta(A,1) \to C$

$\delta(B,0) \to A$ $\quad$ $\delta(B,1) \to D$

$\{A,B\}$ $\{F\}$ $\{C,D,E\}$

Transition Table

	0	1
→{A,B}	{A,B}	{C,D,E}
{F}	{F}	{F}
{C,D,E}	{C,D,E}	{F}



Date : / / 2022

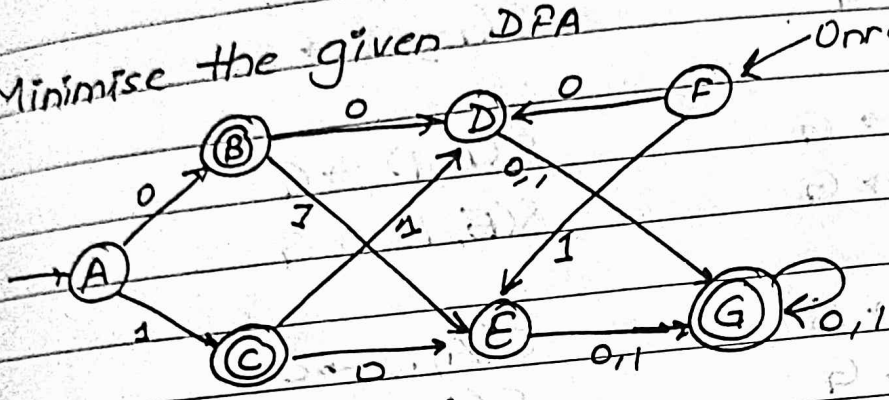
Page No.

classfollow

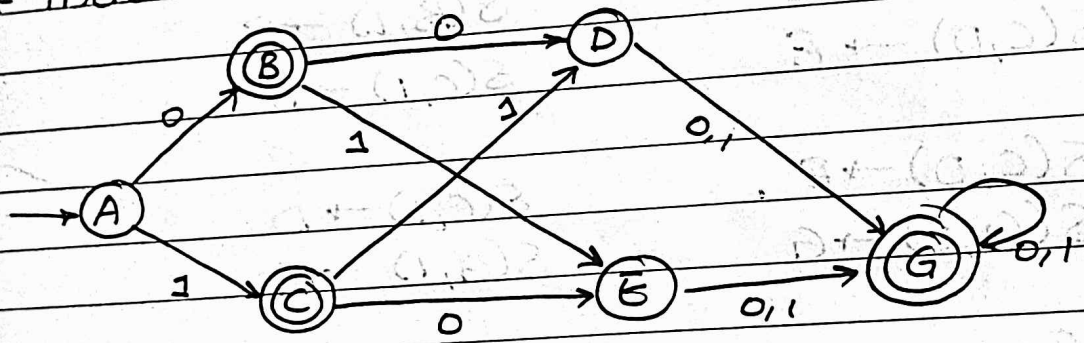
Minimization with Unreachable States

A state is said to be unreachable if there is no way it can be reached from the initial state.

Ques Minimise the given DFA



If there is an unreachable state remove that.



Transition Table

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

0-Equivalence
$\{A, D, E\}$ $\{B, C, G\}$

1-Equivalence

$\delta(A,0) \rightarrow B$ $\delta(A,1) \rightarrow C$
$\delta(D,0) \rightarrow G$ $\delta(D,1) \rightarrow G$

$\delta(D,0) \rightarrow G$ $\delta(D,1) \rightarrow G$
$\delta(E,0) \rightarrow G$ $\delta(E,1) \rightarrow G$

$\delta(A,0) \rightarrow B$ $\delta(A,1) \rightarrow C$
$\delta(E,0) \rightarrow G$ $\delta(E,1) \rightarrow G$

$\delta(B,0) \rightarrow D$ $\delta(B,1) \rightarrow E$
$\delta(C,0) \rightarrow E$ $\delta(C,1) \rightarrow D$

$\delta(C,0) \rightarrow E$ $\delta(C,1) \rightarrow D$
$\delta(G,0) \rightarrow G$ $\delta(G,1) \rightarrow G$

$\{A, D, E\}$ $\{B, C\}$ $\{G\}$

2-Equivalence

$\delta(A,0) \rightarrow B$ $\delta(A,1) \rightarrow C$
$\delta(D,0) \rightarrow G$ $\delta(D,1) \rightarrow G$

$\{A\}$ $\{D, E\}$ $\{B, C\}$ $\{G\}$

3-Equivalence

$\delta(D,0) \rightarrow G$ $\delta(D,1) \rightarrow G$
$\delta(E,0) \rightarrow G$ $\delta(E,1) \rightarrow G$

C P P L
Date : / / 2022
Page No.

classfellow

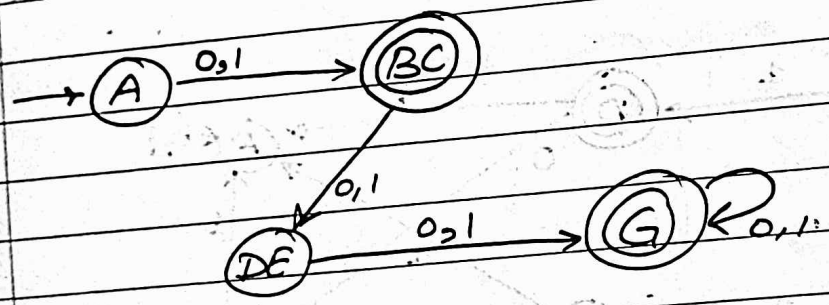
$\delta(B,0) \rightarrow D$
$\delta(C,0) \rightarrow E$

$\delta(B,1) \rightarrow E$
$\delta(C,1) \rightarrow D$

$\{A\}$ $\{D,E\}$ $\{B,C\}$ $\{G\}$

Transition Table

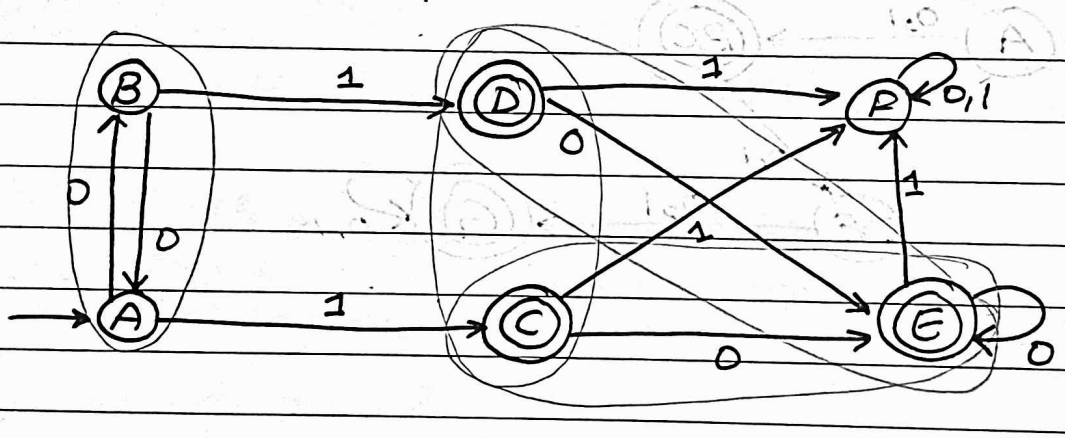
	0	1
→ $\{A\}$	$\{B,C\}$	$\{B,C\}$
(*) $\{B,C\}$	$\{D,E\}$	$\{D,E\}$
$\{D,E\}$	$\{G\}$	$\{G\}$
(*) $\{G\}$	$\{G\}$	$\{G\}$



Myhill – Nerode Theorem

Steps :-

1. Draw a table for all pairs of states (P, Q)
2. Mark all pairs where $P \in F$ and $Q \notin F$
3. If there are any unmarked pairs (P, Q) such that $[\delta(P, x), \delta(Q, x)]$ is marked, then mark (P, Q) x is an input symbol
   Repeat this until no more markings can be made.
4. Combine all the Unmarked Pairs and make them a single state in the minimized DFA



	A	B	C	D	E	F
A						
B						
C	✓	✓				
D	✓	✓				
E	✓	✓				
F	✓	✓	✓	✓	✓	

C P P L
Date : / / 2022
Page No.

classfollow

(A,B)
$\delta(A,0) \rightarrow B$
$\delta(B,0) \rightarrow A$
$\delta(A,1) \rightarrow C$
$\delta(B,1) \rightarrow D$

(D,C)
$\delta(D,0) \rightarrow E$
$\delta(C,0) \rightarrow E$
$\delta(D,1) \rightarrow F$
$\delta(C,1) \rightarrow F$

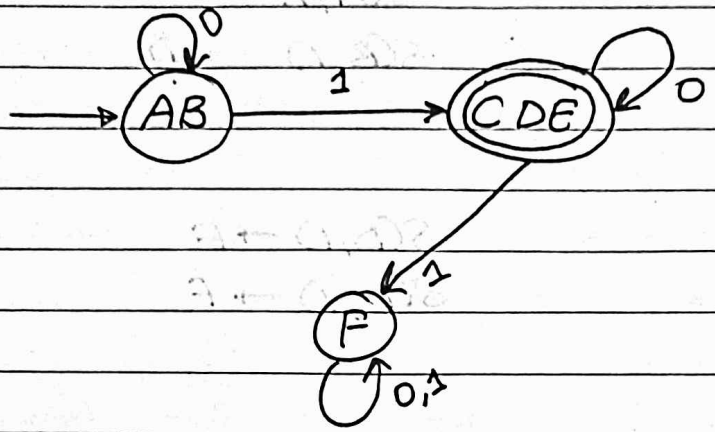
(E,C)
$\delta(E,0) \rightarrow E$
$\delta(C,0) \rightarrow E$
$\delta(E,1) \rightarrow F$
$\delta(C,1) \rightarrow F$

(E,D)
$\delta(E,0) \rightarrow E$
$\delta(D,0) \rightarrow E$
$\delta(E,1) \rightarrow F$
$\delta(D,1) \rightarrow F$

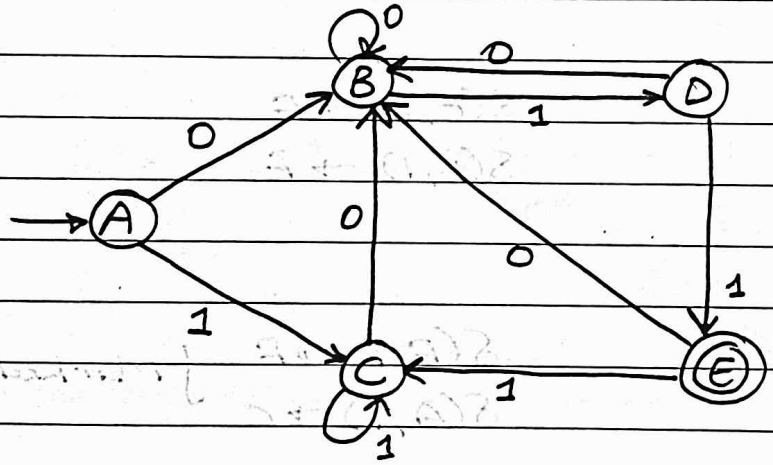
(F,A)
$\delta(F,0) \rightarrow F$
$\delta(A,0) \rightarrow B$
$\delta(E,1) \rightarrow F$
$\delta(A,1) \rightarrow C$
] Marked

(F,B)
$\delta(F,0) \rightarrow F$
$\delta(B,0) \rightarrow A$
$\delta(E,1) \rightarrow F$
$\delta(B,1) \rightarrow D$
] Marked

Unmarked pairs
(A,B) (D,C) (E,C) (E,D)



Ques Minimize the following DFA using Table Filling Method.



	A	B	C	D	E
A					
B	✓				
C		✓			
D	✓	✓	✓		
E	✓	✓	✓	✓	

classfellow

(A, B)
$\delta(A,0) \to B$
$\delta(B,0) \to B$
$\delta(A,1) \to C$] Marked
$\delta(B,1) \to D$

(A, C)
$\delta(A,0) \to B$
$\delta(C,0) \to B$
$\delta(A,1) \to C$
$\delta(C,1) \to C$

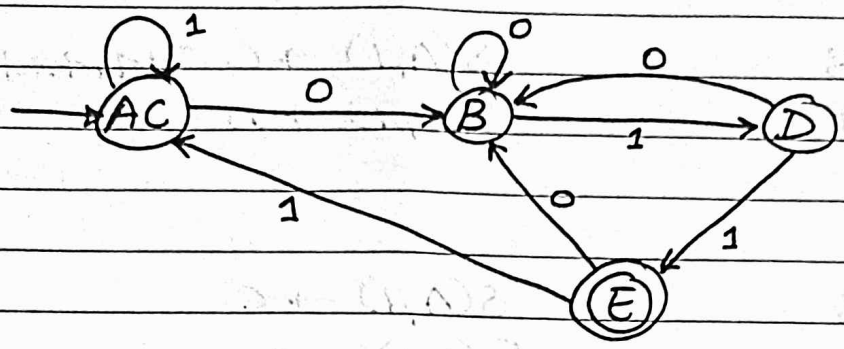
(B, C)
$\delta(B,0) \to B$
$\delta(C,0) \to B$
$\delta(B,1) \to D$] Marked
$\delta(C,1) \to C$

(D, A)
$\delta(D,0) \to B$
$\delta(A,0) \to B$
$\delta(D,1) \to E$] Marked
$\delta(A,1) \to C$

(D, B)
$\delta(D,0) \to B$
$\delta(B,0) \to B$
$\delta(D,1) \to E$] Marked
$\delta(B,1) \to D$

(D, C)
$\delta(D,0) \to B$
$\delta(C,0) \to B$
$\delta(D,1) \to E$] Marked
$\delta(C,1) \to C$

Unmarked Pair
(C, A) B D E



Finite State Machine

A finite state machine is described by 6 typ
$(Q, \Sigma, \Delta, \delta, \lambda, q_0)$

$Q \rightarrow$ Finite and non-empty set of states.
$\Sigma \rightarrow$ Input alphabets
$\Delta \rightarrow$ Output alphabets
$\delta \rightarrow$ Transition function that maps present state and input symbol onto next state.
$\lambda \rightarrow$ Output function
$q_0 \rightarrow$ Initial state, $q_0 \in Q$

Finite State Machine

- Mealy Machine
- Moore Machine

classfellow

Mealy Machine

Described by 6 tuples.

$(Q, \Sigma, \Delta, \delta, \lambda, q_0)$

where

Q = Finite set of states

Σ = Finite non-empty set of Input alphabets

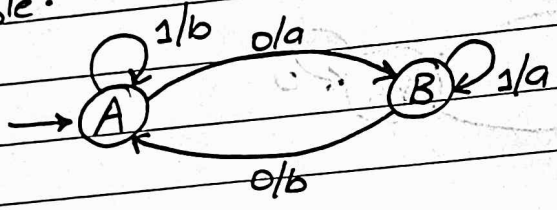
Δ = Set of Output Alphabets

δ = Transition function : $Q \times \Sigma \to Q$

λ = Output Function : $Q \times \Sigma \to \Delta$

q_0 = Initial State

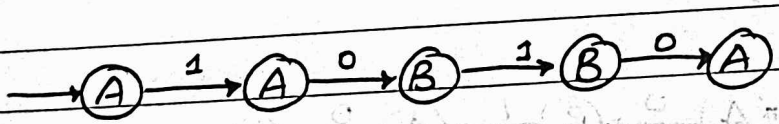
Example :-



When input 1 is given on state A, it gives an output b.

Output can be different inputs even on the same states.

Input = 1010



b a a b

Output = baab

n → n

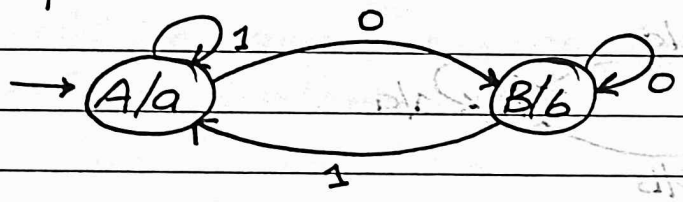
↓

This means n input symbols give n output symbols

Moore Machine

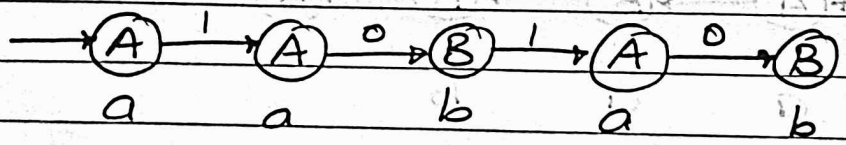
Described by 6 tuples
$(Q, \Sigma, \Delta, \delta, \lambda, q_0)$
Where
Q = Finite set of states
Σ = Finite non-empty set of Input Alphabets
Δ = Set of output Alphabets
δ = Transition function : $Q \times \Sigma \to Q$
λ = Output function : $Q \to \Delta$
q_0 = Initial State

Example :-



When input 1 is given on state A, it gives an output a.
Output remains same for same state even for different input values.

Input = 1010



Output = aabaa

n → n+1
↓
4 input symbols gives an output of 5

Date : / / 2022
Page No.

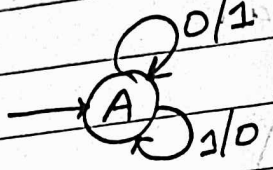
classfellow

Construction of Mealy Machine

Ex1 Construct a Mealy Machine that produces 1's complement of any binary input string.

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

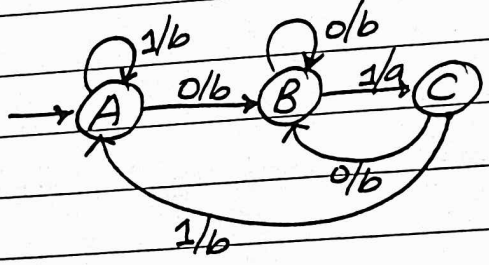
$\Delta = \{0, 1\}$



Ex2 Construct a Mealy Machine that prints 'a' whenever the sequence '01' is encountered in any input binary string.

$\Sigma = \{0, 1\}$ $\Delta = \{a, b\}$

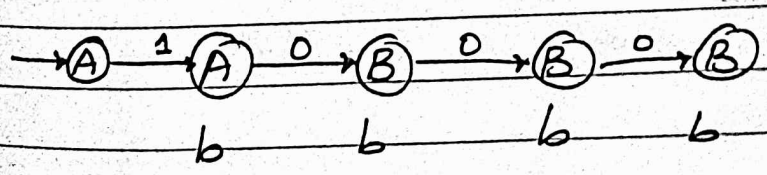
To simplify the problem, try to make DFA of string ending with 01 and then convert to Mealy Machine.



Input = 0110

Output = babb

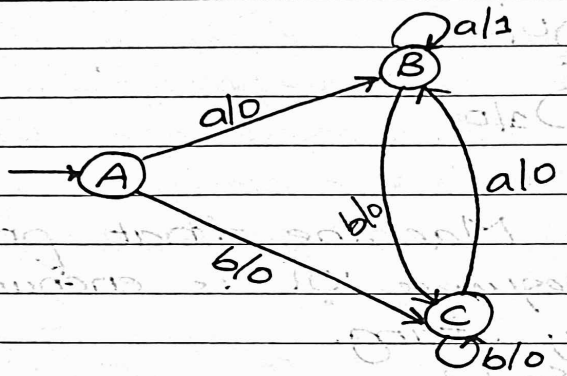
Input = 1000



Output = bbbb

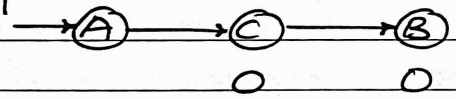
C P P L
Date : / / 2022
Page No.

Ex3 Design a Mealy Machine accepting the language consisting of strings from Σ^* where $\Sigma = \{a, b\}$ and the string should end with either aa or bb



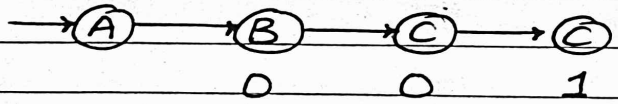
Output 1 is given incase of ending with aa or bb only.

Input = ba



Output = 00

Input = abb



Output = 001.

Date : / / 2022
Page No.

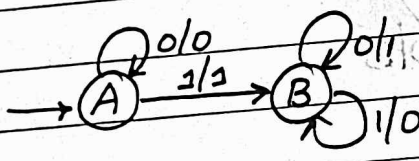
classfellow

Ex4 Construct a Mealy Machine that gives 2's complement of any binary input. (Assume that the last carry bit is neglected)

2's complement = 1's complement + 1

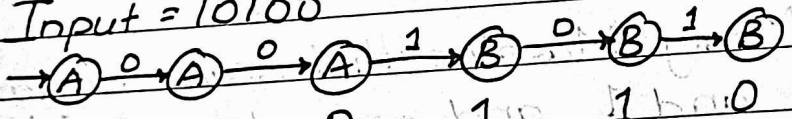
Example:-

10100	11100	1111
1's = 01011	1's 00011	1's 0000
+1	+1	+1
2's = 01100	2's 00100	2's 0001



Input = 10100

Read from LSB to MSB

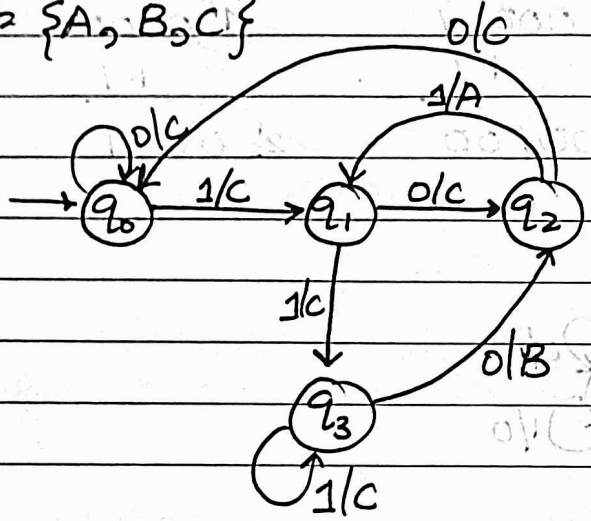


Output = 01100 [Output reversed as input was given from LSB]

Ex15 Design a Mealy machine, for a binary input sequence such that if it has a substring 101, the machine output A, if the input has substring 110, the machine output B. For other strings output will be C.

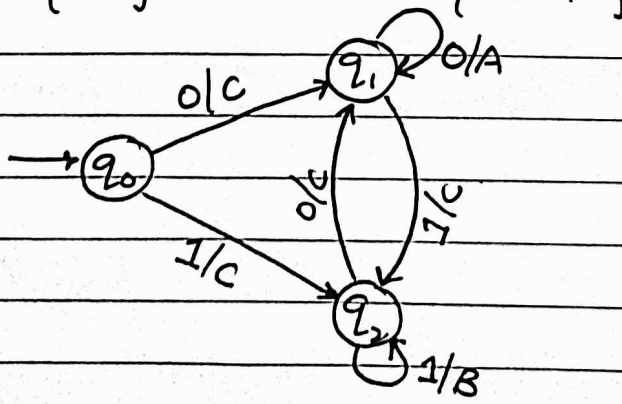
$\Sigma = \{1, 0\}$

$\Delta = \{A, B, C\}$



Ex16 Design a Mealy Machine that scans sequence of input 0 and 1 and generates output 'A' if the input string terminates in 00, output 'B' if the string terminates in 11, otherwise output 'C'

$\Sigma = \{0, 1\}$ $\Delta = \{A, B, C\}$



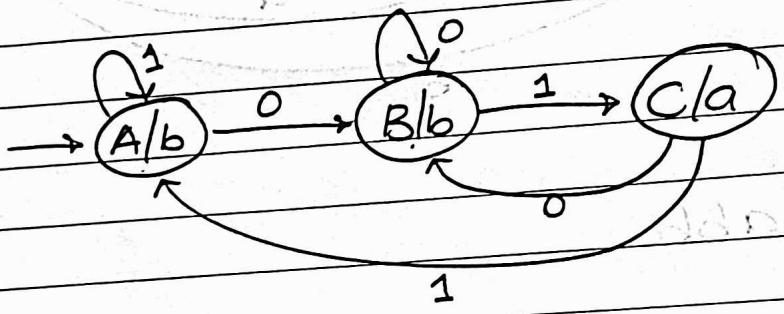
classfellow

Construction of Moore Machine

Ex1 Construct a Moore Machine that prints 'a' whenever the sequence '01' is encountered in any input binary string.

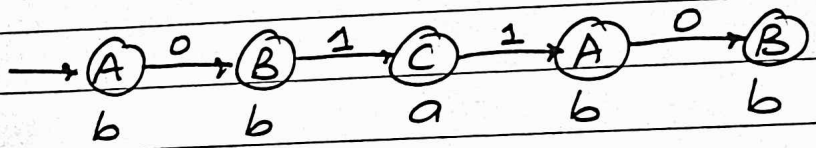
$\Sigma = \{0, 1\}$

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



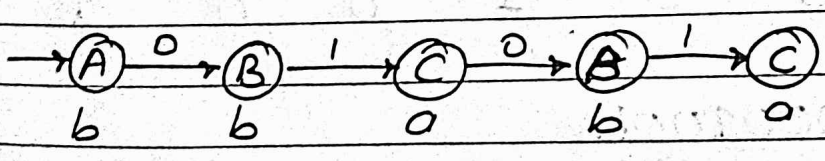
Input = 0110

Output = bbabb



Input = 0101

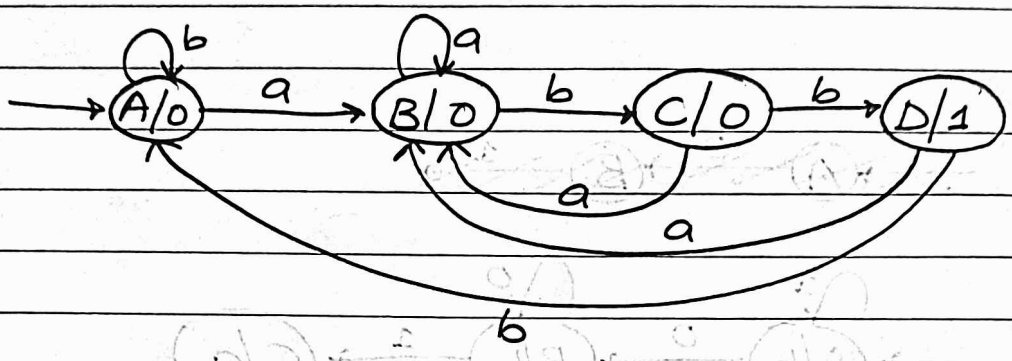
Output = bbaba



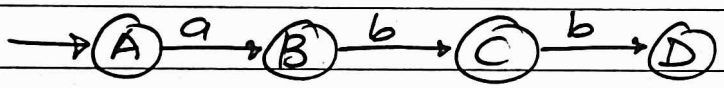
Ex2 Construct a Moore Machine that counts the occurences of the sequence 'abb' in any input strings over {a,b}

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

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

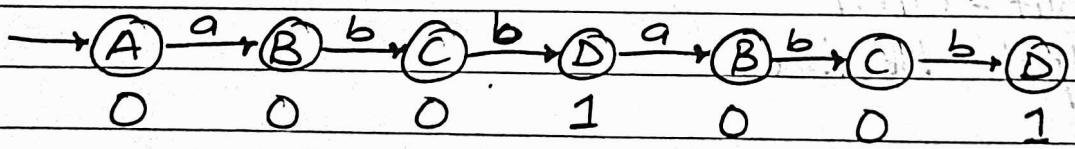


Input = abb



Output = 0001

Input = abbabb



Output = 0001001

Total occurence = 2

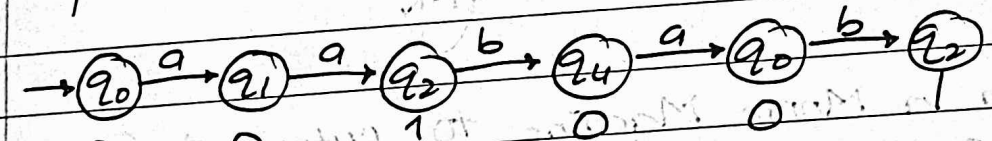
classfellow

Ex3 For the following Moore Machine the input alphabet is $\Sigma = \{a, b\}$ and the output alphabet is $\Delta = \{0, 1\}$. Run the following input sequences and find the respective outputs:

(i) aabab
(ii) abbb
(iii) ababb

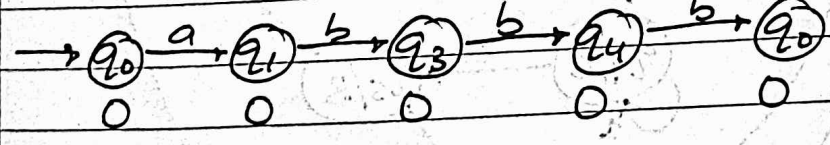
States	a	b	Outputs
$\rightarrow q_0$	q_1	q_2	0
q_1	q_2	q_3	0
q_2	q_3	q_4	1
q_3	q_4	q_4	0
q_4	q_0	q_0	0

(i) Input = aabab



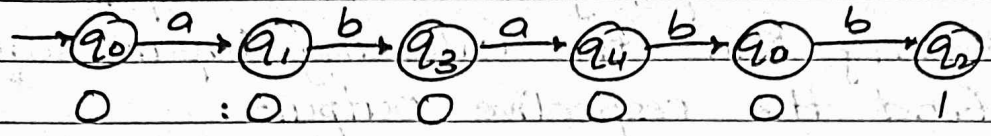
Output = 001001

(ii) Input = abbb



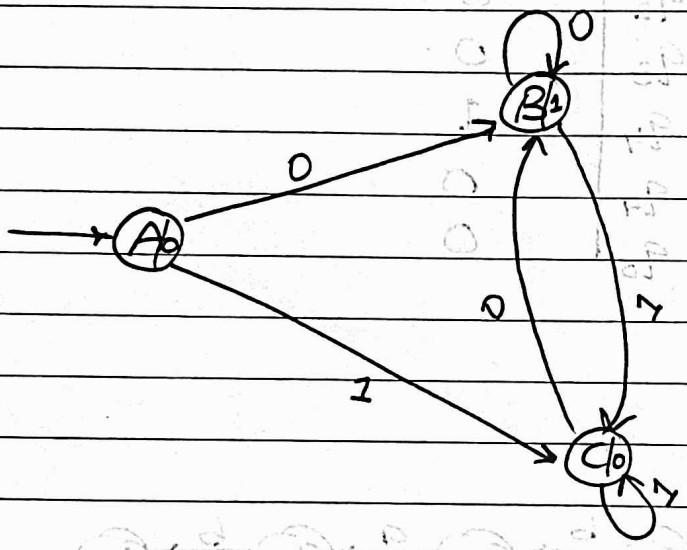
Output = 00000

(iii) Input = ababb

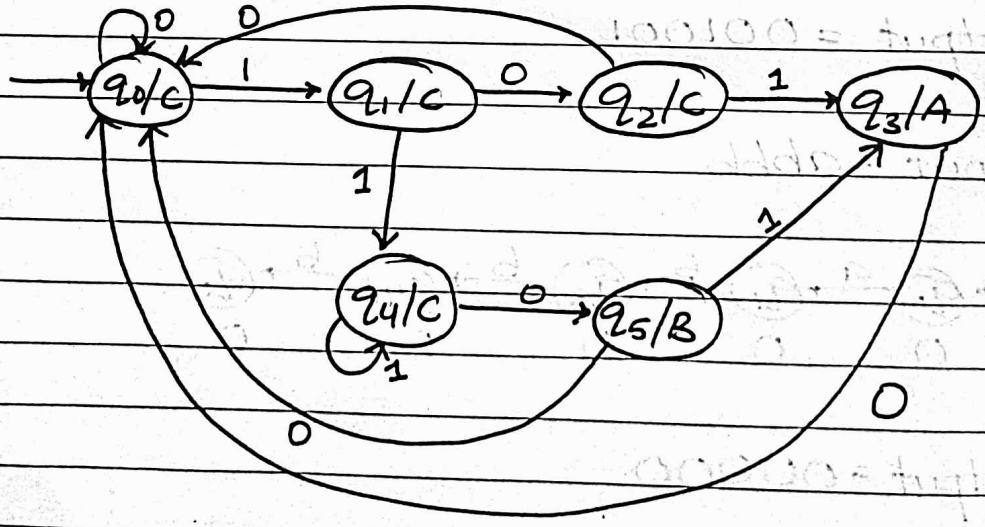


Output = 000001

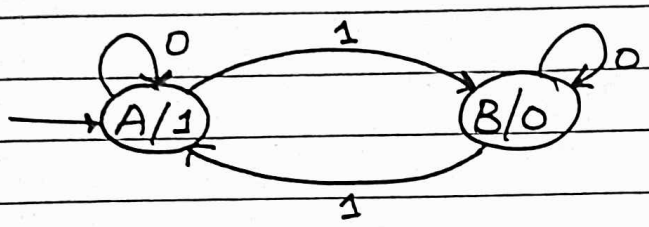
Ex4 Design a Moore Machine to generate 1's complement of a given binary number.



Ex5 Design a Moore Machine to output A for substring 101, B for 110, C for rest.



Ex6 Construct a Moore Machine that determines whether an input string contains an even or odd number of 1's. The machines should give 1 as output if an even number one's are in string otherwise 0.



Ex6 Design a Moore Machine with input alphabets {0,1} and output alphabet {Y,N} which produces Y as output if input sequence contains 1010 as substring otherwise N.