Page No.
Date : / /

Applications and Variations of FA-

Applications- 1. Lexical Analysis in compiler.

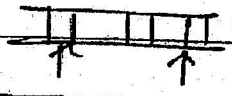
2. Pattern Matching but not string matching.
   ex- Lexical Analyzer
3. Text Editor
   - FIND/ SEARCH/ REPLACE
     ex- (starting with q)
   - Spell checker
     Ex- (R.E)
     ex- complete
     It will come up with menu
     ~~have string~~ starting with compl.
   - GREP (Search command) in UNIX
     (General Regular Expression Processor
4. Finite storage
   Theorem - if m words are there each of size n you will require 2^{mn} states for binary words.
   if ~~th~~ it is ternary = 3^{mn}
5. Sequential Circuit Design.
   (using mealy & moore)
   ex- 1's complement, 2's complement.

- FA is used to modelling a simple machine with less memory finite memory.
  ex- Vending Machine

EXCELLENT

Variations of FA-

1. Multitape FA is equivalent to FA
   they both have same Power.
2. Multihead FA is equivalent to FA
   ie more than one head



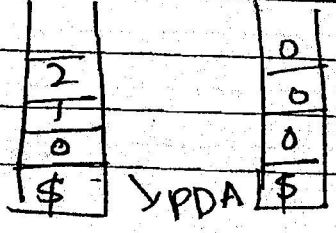
   it now only has finite memory.
3. Two way automata also have same power to FA
   it is also known as 2-NFA & 2-DFA
   ie you can move left or right acc to your requirement
   ie one can move to left and other can move to right
   but it can not accept non-regular language because of limited memory.

$\delta(q_0,1)=(q_1,L)(q_2,R)$

4. FA with stack.
   it has more power than FA as it become PDA of infinite size.

5. FA with finite stack has same power as FA as M/M is limited here.

6. FA + counter has more than FA
   it is known as CA (counter Automata)
   it is similar to FA with stack but the basic difference is stack can have any symbol.
   but counter stack can have only one symbol



Page No.
Date : / /

therefore, we can say that

FA < FA+counter < FA+stack

(it can do counting comparison → (it can do counting f^n but cant do stair matching))

∴ Counter Automata has less power than PDA but has more than power than finite automata.

7. FA + two-stacks is equivalent is TM. it is not equal to T

FA + 2-stack > FA + stack > ~~PDA~~ FA

2-PDA > 1-PDA > FA

So FA + two stack is more powerful than FA + 1-stack.
(it is equivalent to TM as it allows infinite Memory & movem in both direction)

8. FA + 3 stack has equal power as FA + 2 stack have, because Not machine is more powerful than TM.

Ques - n-stack PDA ≅ TM

No - {n ≥ 2}

9. FA + two counter is equivalent to Turing Machine.

FA < FA + 1counter < FA + 1-stack < FA + 2-ccounter
or
FA + 2-stack
|||
TM

To push FA to level of FA, min required = 2 counters

EXCELLENT

Page No.
Date : / /

10. FA + 1 Queue is equivalent to TM. It is also known as Queue Automata (QA)

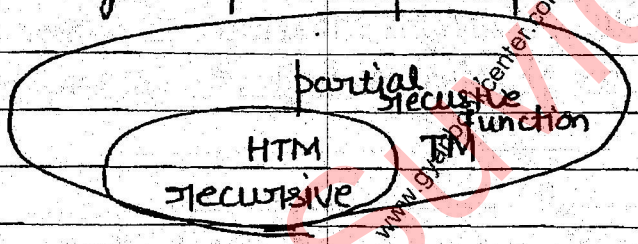
Ques - n-Queue FA $\cong$ TM n?

{$n \geq 1$}

TRANSDUCER - produces an Output

Mealy Machine, Moore Machine

{Every logically computable f^n is partially recursive f^n.}



1. Theory of Mealy & Moore.
2. Mealy/Moore + Input → Output?
3. Mealy/Moore → function
4. Mealy to Moore or Moore to Mealy.

Mealy & Moore Machine - · It is a dfa with output.
· it is not NFA as choice, ε, dc are not allowed but if is NFA as each dfa is NFA.

or Mealy - output f^n. $\Delta_{Q \times \Sigma \to \Gamma^*}$ $\delta(q_0, a) = 001$

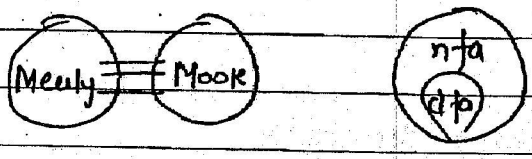
Moore Output f^n. $\Delta_{Q \to \Gamma^*}$ $\Delta(q_0) = 001$

in Mealy O/P is when you exit but in Moore out is when you enter.

EXCELLENT

Page No.
Date : / /

as $\Delta(q_0, a) = 001$
$\Delta(q_0, b) = 01$
but $\Delta(q_0) = 001$ (always same)
Mealy & Moore both have same powers & they are equ



• Each dfa is Nfa but Moore is not Mealy.

Formal Definitions:

Mealy/Moore $(Q, \Sigma, \Gamma, \delta, \Delta, q_0)$

- Q — set of state
- Σ — input symbols
- Γ — output alphabet
- δ — transition function
- Δ — output function
- q_0 — initial state

Differences -

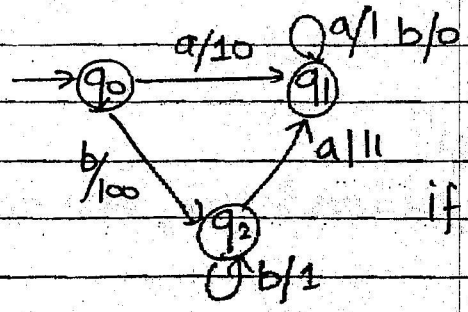
DFA/NFA	Mealy/Moore
• 5 tupled	• 6 tupled
• Final state as it work as acceptor.	• No final states are there.
• Γ, Δ - are not there	• Γ, Δ are there Γ - as O/P is to be specified Δ - output function.

Here, $\delta_{Q \times \Sigma \to Q}$

O/P function in Mealy - $\Delta_{Q \times \Sigma \to \Gamma^*}$ (O/P is fn of both present state & input alphab

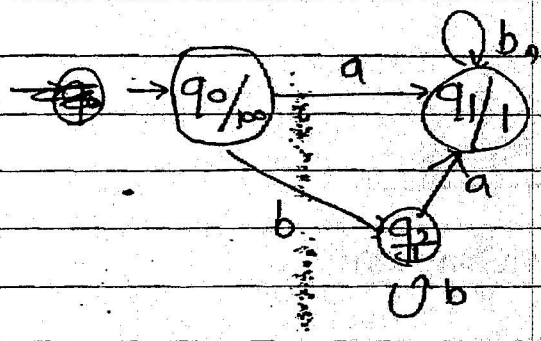
O/P function in Moore - $\Delta_{Q \to \Gamma^*}$ (O/P is fn of present state only

EXCELLENT



— Mealy Machine

if q_0, a comes go to q_1 and give o/p 10.



— Moore Machine

if $\Delta_{Q \times \Sigma^* \to \Gamma^*}$ then it may allow ε as input but it is not the case with transducer.

[We can have o/p as Null as we have o/p alphabet as Γ^*]

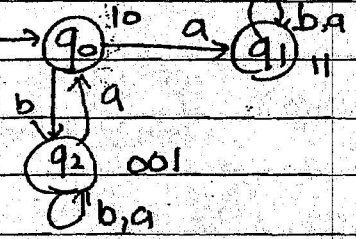
Ques- if you a Mealy machine with m state & n output then Moore will have atmost ~~no~~ how many state?

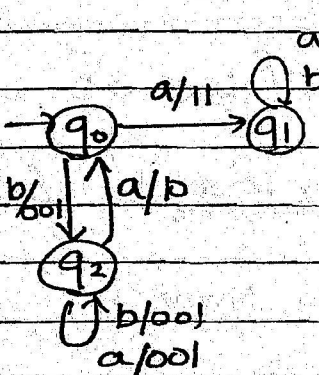
Ans- $\leq mn+1$ state

Ques Moore machine with m state & n output, equivalent Moore will have—

Ans- atmost mn states.

Ex -





but the o/p with transition of the entering states.

EXCELLENT

Page No.
Date : / /

After conversion from Moore to Mealy it is possible to delete some states as well.

	a	b	o/p
→q_0	q_1	q_2	10
q_1	q_2	q_1	11
q_2	q_2	q_0	001

← Moore Machine

Mealy Machine

	a	o/p	b	o/p
→q_0	q_1	11	q_2	001
q_1	q_2	001	q_1	11
q_2	q_2	001	q_0	10

If it is

	a	o/p	b	o/p
→q_0	q_1	11	q_2	001
q_1	q_2	001	q_1	11
q_2	q_2	001	q_1	11

↳ both are equivalent-

∴ equivalent is

→q_0	q_1	11	q_2	001

∴ it will have atmost m states

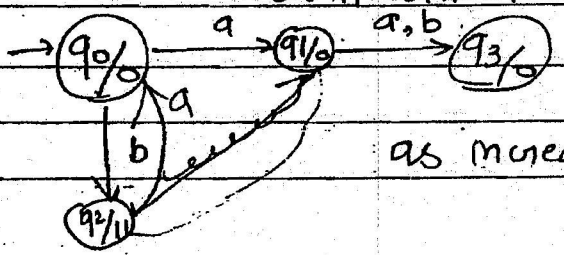
Moo Mealy to Moore-

	a		b	
q_0	q_1	0	q_2	11
q_1	q_2	0	q_1	0
q_2	q_0	0	q_1	11

Here m=3

n=2 ∴ worst case = 7 states

as it will result for many states



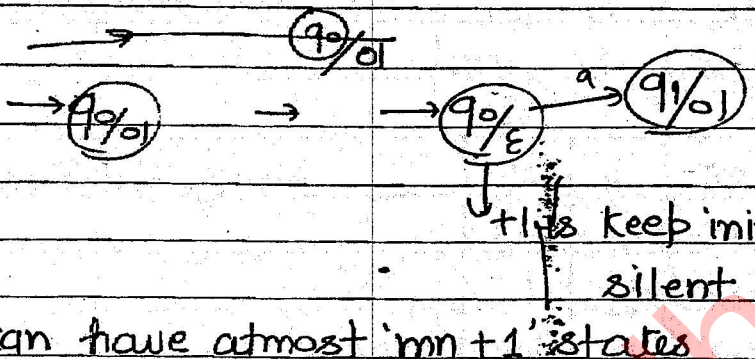
as increase of state occur.

mn cases can happen.

EXCELLENT

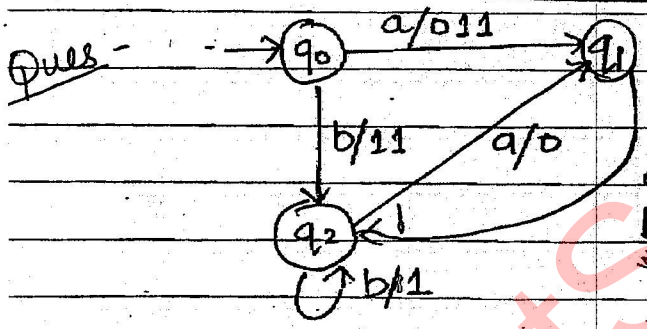
Page No.
Date : / /

+1 occurs as when you enter Moore Machine will also produce o/p even at null state.
So you have keep initial state silent in begining so keep it output for ε at starting & add another state



∴ it can have atmost 'mn+1' states

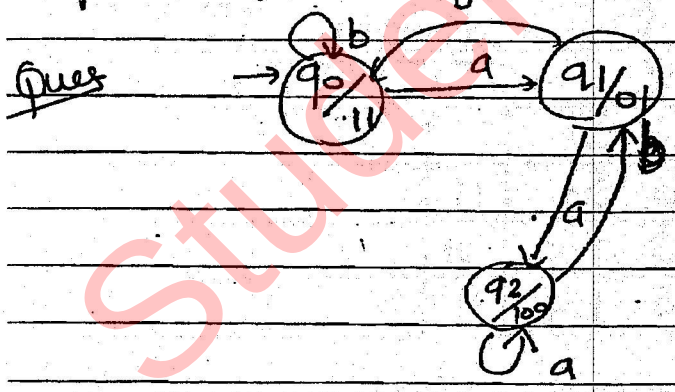
Ques -



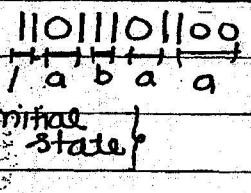
I/P = abaa
O/P = ?
0110001

for every I/P there is one o/p only

Ques



'abaa'



Note - if n size input string is given O/P will have n in Mealy
but in Moore - it is n+1.

EXCELLENT

Page No.
Date : / /

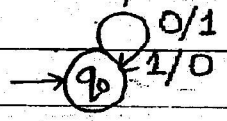
functions -

1. 1's complement
2. 2's complement
3. Binary Full Adder
4. Increment by 1
5. change the sign bit
6. Take d(w) mod 3 (INTEGER DIVISIBILITY TESTER)
7. LOGICAL FUNCTION (NAND, NOR, XOR)
   (Simple Binary logical function)
8. Non-standard functions.

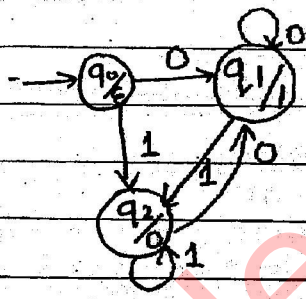
$\Sigma = \{0, 1\}$

Moore is easier than Mealy if o/P are fixed ex-Mod n output is fixed.

1. 1's complement

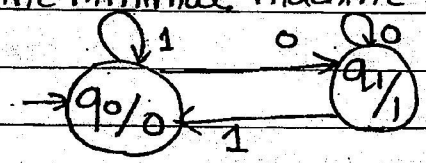


Mealy Machine

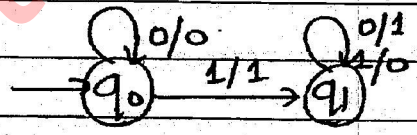


– Moore Machine

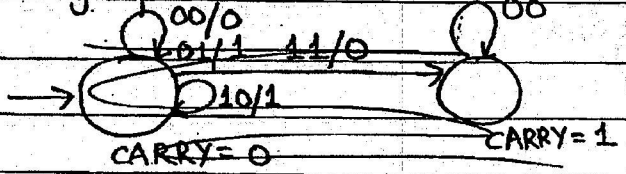
the Minimal machine is



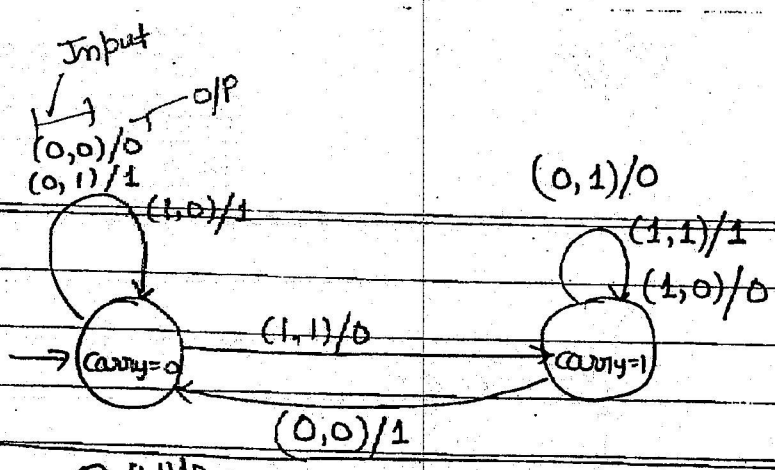
2. 2's complement- • Input from LSB
• Remain as it is till first 1 came.
• Remain 1st 1 same & then complen



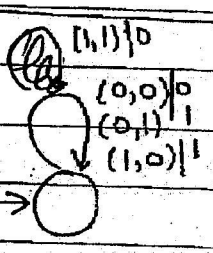
3. Binary full Adder • I/P is pairs to full adder



EXCELLENT

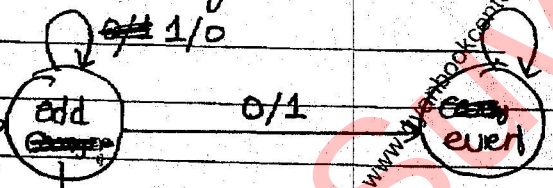


XOR - (half adder or XOR)



5. Increment by 1. even no- 1101**0**
odd no - 1101**1**

for even no - make LSB as 1. • Input from LSB.
for odd no - change has to done

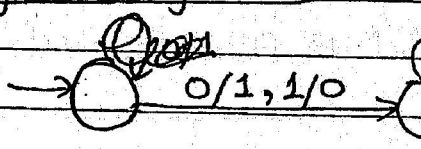


Ques - m state n O/P → total state in Moore- mn+1 or mn

because programmer can give instruction to user to not read starting state output. so no need of putting silent twin.

if mn+1
mn choose mn+1

5. Change the sign bit - • Input from MSB



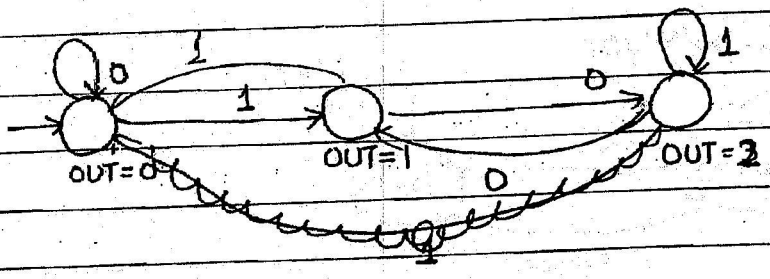
EXCELLENT

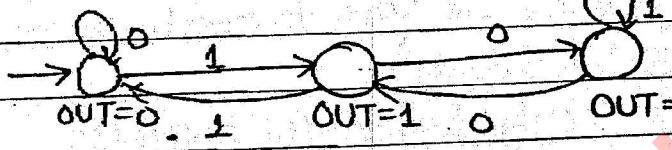
101101

01010110 Ans

Page No.
Date : / /

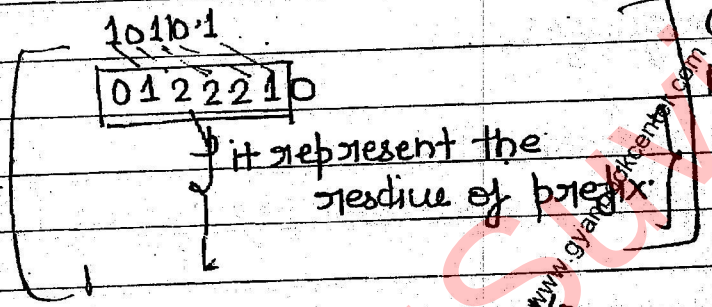
6. d(w) mod 3 –



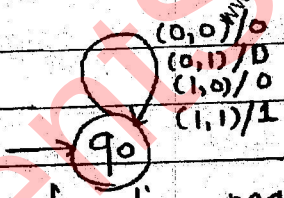


for 101101
0122210

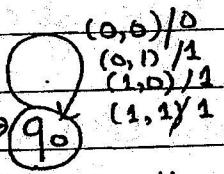
• instruction is given if last b
0 then it is divisible by 3
If 1 → remainder is



7. AND



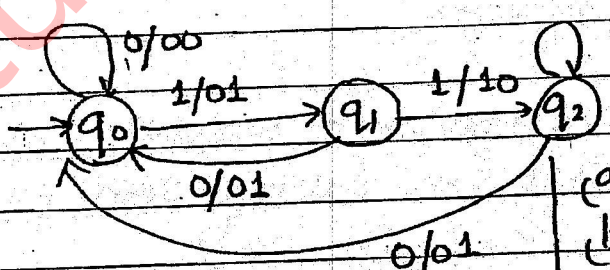
OR



for any logical function map value according to the truth tab

NON-STANDARD FUNCTION

Ques



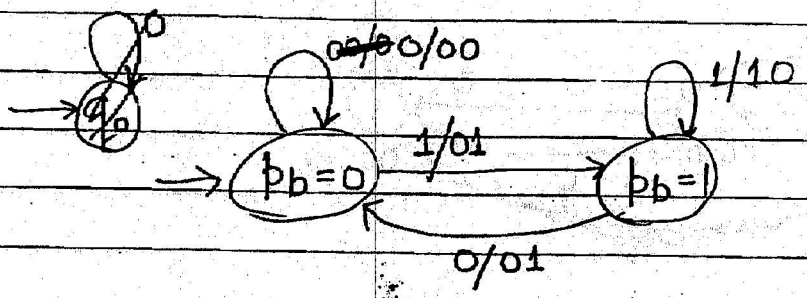
(a) O/P as 01 for every 1 in I/P
(b) O/P as 00 — 0 in I/P
(c) sum of present & current previous bit of input
(d) None of these

• for such question, instead of drawing machine, just take an arbitrary input & check for o/p

EXCELLENT

Page No.
Date : / /

Sum of present & previous bit-



{We have taken 2 states here as we have to ~~remb~~ remember only previous bit & it may be 0 or 1 }
∴ only two states required

Context Free Languages-

{Every Mealy Machine is moore machine - False
but for each Mealy, there exist an equivalent Moore - true }

1. L → G & Types of CFL's (Standard CFL & grammar)
   G → L
2. Derivations: LMD, RMD, Derivation tree, partial DT
   - Right Most Derivation
   - Left Most Derivation
   - Ambiguity of Grammar
   - Ambiguity of Language
3. Membership algorithm- • CYK Algorithm — Complexity - $O(n^3)$
   • Brute Force parsing ~~O(K^n)~~ Algorithm
   (complexity - $O(k^n)$)

• Properties of LL(k) & LR(k) Grammars.
↳ It will take O(n) time to check for membership.

EXCELLENT