Unit I

DATE: / /
PAGE NO.:

Ch 8. Finite Automata with output

Moore Machines (1956) Purpose is to design a mathematical model for sequential circuits, as one component of architecture of a whole computer.

Def^n –

Moore m/c is a collection of 5 things:

1. A finite state of states $q_0, q_1, \dots$, q_0 - start st.
2. Alphabet of letters for forming i/p str. Σ {a b c ...}. (letters)
3. Alphabet of possible o/p char. $\Gamma = \{x\ y\ z \dots\}$. (char.)
4. A transition table that shows for each state & each i/p letter which state is reached next.
5. An o/p table that shows what char. from Γ is printed by each state as it is entered.

Moore m/c always begins by printing the char. dictated by the mandatory start state. If i/p has 7 letters, o/p will have 8 char., because it includes 8 states in its path.

M.m. does not define a lang. of accepted words, because every possible i/p str. creates an o/p str. & there is no such thing as a final state.

Circle depicting the states contains 2 symbols separated by /, left one is name of state & right is o/p from that state.

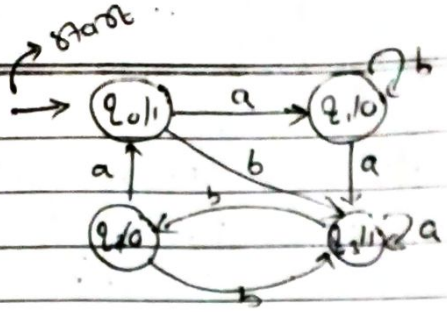
ex. i/p : $\Sigma = \{a\ b\}$

o/p : $\Gamma = \{0\ 1\}$.

Names of states : q_0, q_1, q_2, q_3

Transition table

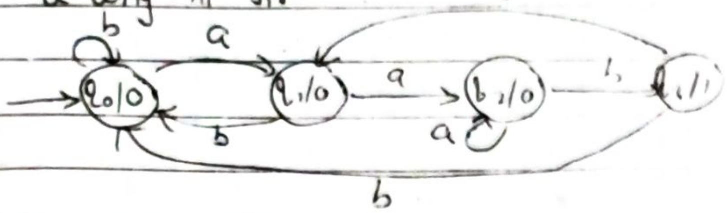
Old state	o/p by old state	New st. After i/p a	New st. After i/p b
$-q_0$	1	q_1	q_3
q_1	0	q_3	q_1
q_2	0	q_0	q_3
q_3	1	q_3	q_2



i/p : abab

o/p : 10010

ex. count how many times the substring aab occurs in a long i/p str.



No. of substr. aab in i/p, will be exactly the No. of 1's in o/p str.

i/p		a	a	a	b	b	a	a	b
state	q_0	q_1	q_2	q_2	q_3	q_0	q_1	q_2	q_3
o/p	0	0	0	0	1	0	0	0	1

Given a lang. L & an FA that accepts it, if we add the printing instrn. 0 to any nonfinal state & 1 to each final state. Above m/c. accepts all words that end in aab.

(1955)

Mealy Machine – is like a Moore m/c except that now we do our printing while we travel along the edges, not in the states themselves. From 1 state to another, if there are 2 edges, a & b edges, then printing instrn may differ.

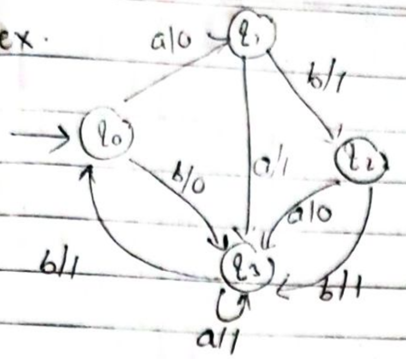
Def.ⁿ – collection of 4

1. finite set of states $q_0, \ldots$
2. $\Sigma = \{a \; b \; \ldots\}$
3. $\Gamma = \{x \; y \; \ldots\}$
4. pictorial representation : states - circle, directed edges - transitions., Edge labeled with i/o, i - i/p, o - o/p

Every state must have exactly one outgoing edge for each possible i/p letter.

(Table defⁿ is not simple as Moore)

ex.



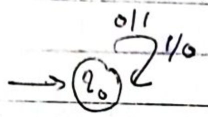
5 ways to arrive in st. q_3 & thus 5 options for o/p.

When we arrive at q_1, there is only 1 option — print 0.

Trace running of m/c on i/p aaab.
o/p → 01110

No. of char. in i/p str. & o/p str. are equal.

ex. 1's compliment



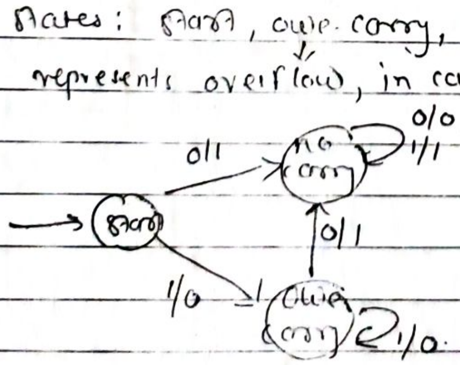
i/p: 001 → o/p: 110
Here i/p & o/p alphabets are both {0 1}.

ex. increment m/c: binary number is the i/p & prints a bin number that is one larger.

→

i/p str is fed in backward, i.e. units digit first, then 2's, 4's digits. o/p will be number 1 greater & is generated right to left

3 states: start, owe-carry, no-carry.
represents overflow, in case of 1+1 = 0, carry = 1.



ex. i/p 1011 (11)
fed as 1101
o/p : 0011
reverse : 1100 (12).

l(o/p str.) = l(i/p str.) ∴ if i/p = 1111, o/p = 0000,
∴ interpret the owe carry state as an overflow situation if a str. ever ends there.

Above 2 m/c's are valuable in computing.

Incrementer — addition,

1's complement — subtraction.

ex. If a & b are binary then a-b = .

① add 1's comple. of b to a, ignore any overflow digit.

② increment results by 1.

ex. 14-5 = 1110 - 0101

= 1110 + 1010 + 1

= [1]1001 = 9

18-7 = 10010 - 00111

= 10010 + 11000 + 1

= [1] 01011 = 11

in decimal, use 9's complement

replace each digit d in 2nd No. by (9-d).

ex. 46-17 → 46 + 82 + 1 → [1]29 → 29

[9-1] [9-7]

ex. Mealy m/c does not accept or reject an I/P str., it can recognize a lang. by making its O/P str. answer some questions about ~~str~~ I/P.

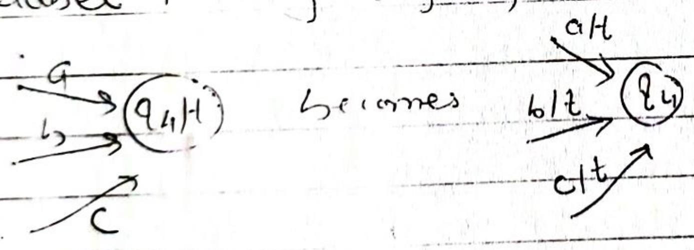
ex. lang. has double letters.

Equivalence: 2 moore m/c's are equal (or 2 mealy) if they give same o/p if presented with same i/p.
But a Moore m/c can never be directly eqvt. to a Mealy m/c because of difference in o/p str. length.
∴ we ignore the o/p of the automatic start ~~stat~~ symbol for Moore m/c.

★ ✓ Def^n – Given Mealy m/c Me & Moore Mo, which prints automatic start state char. x, Me & Mo are eqvt. if for every i/p str., o/p from Mo is exactly x concatenated with o/p from Me.

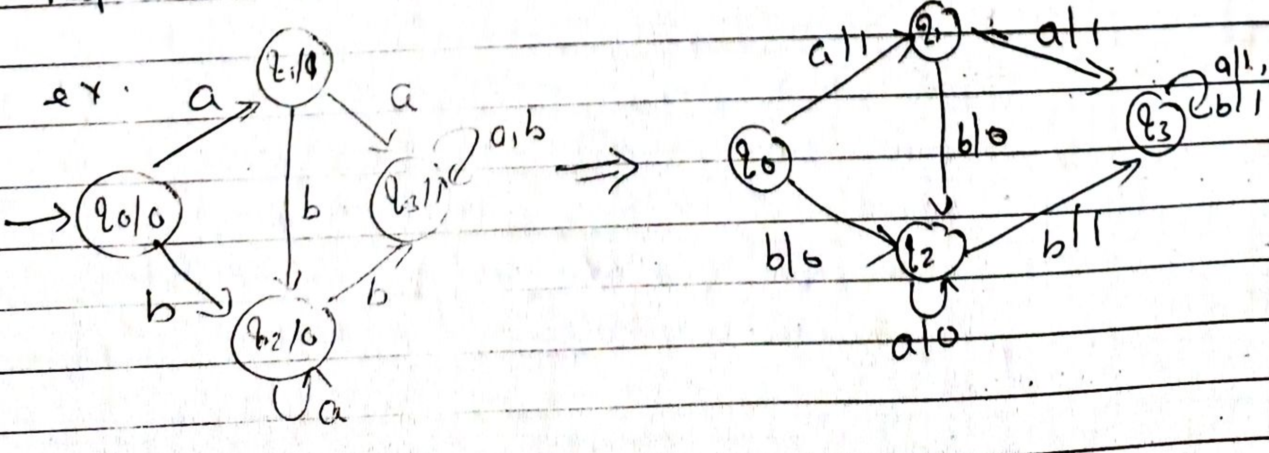
Th^m – If Mo is a Moore m/c, then there is a Mealy m/c that is eqvt. to it.
→ In Mo,
for state q_i where o/p char. is t,
relabel incoming edges, add t to it.

ex:



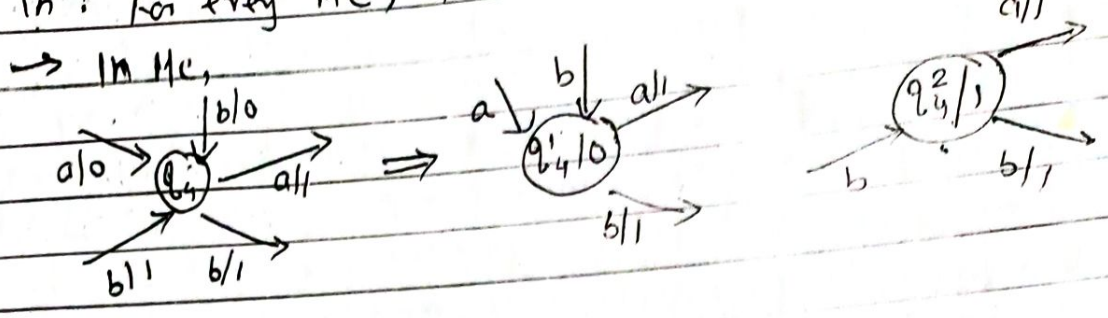
Outgoing edges will get relabeled when we consider state to which they lead.
Repeat above for all states, & we get eqvt. Me.

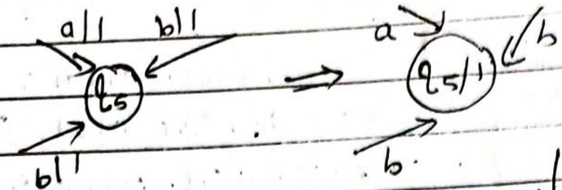
ex.

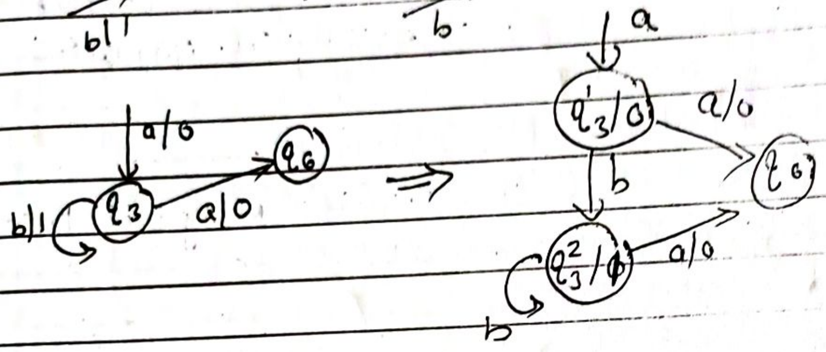


Th^m: For every Me, there is Mo eqvt- to it.

→ In Mc,







If a state has no incoming edges, we can assign it any pointing instrn. we want, even if this state is the start state.

If we have to make copies of start state in Me, we can let any one of them be the start state. in Mo because they all give the identical directions for proceeding to other states.

(Thus conversion of Me into Mo is not unique.)

Any Me is eqvt. to more than one Mo.

It is eqvt. to Mo with automatic start symbol 0 or to the Mo with auto. st. sym. 1, ...

When we start up the m/c initially, we print some unpredictable char., specified by start state, that does not correspond to any o/p from Me, because Me never prints before reading an i/p letter.

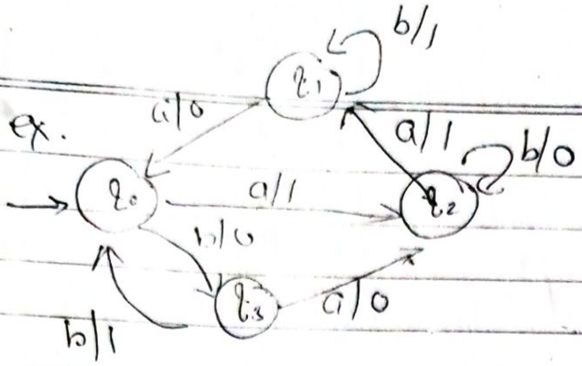
But the def^n of equivalence takes care of this.

Thus Me = Mo.

Mo → Me – same no. of states & edges.

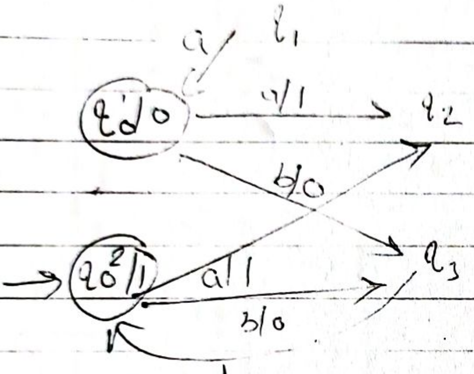
Me → Mo → no. ~~can~~ increases drastically.

ex.

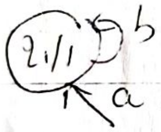


→

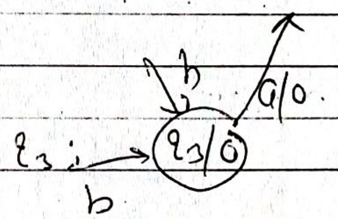
q_0 : 2 incoming edges a/0, b/1 ∴ need 2 copies.
connect both these st. to q_2 & q_3.



q_1 :

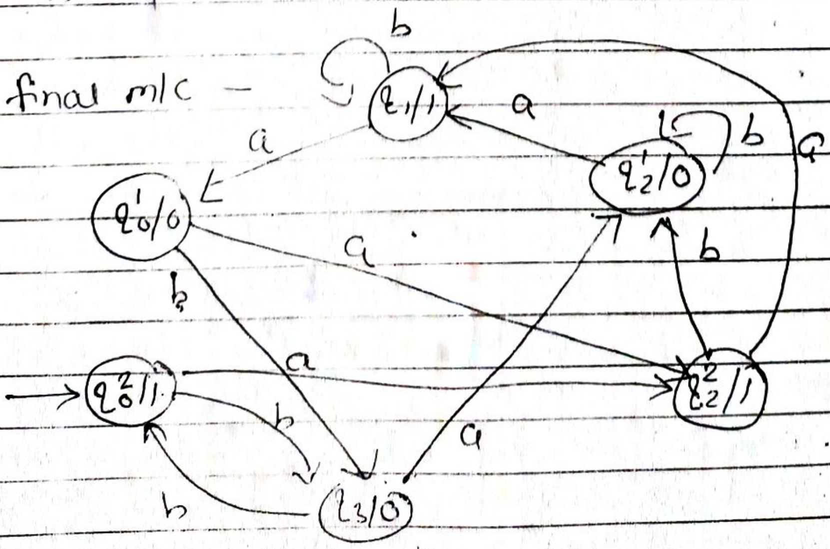


q_3 :



q_2 : 2 0-printing & 2 1-printing edges.
∴ 2 copies — q_2^1 & q_2^2 — 1st print 0 & 2nd — 1.
copies connected by b edge going from q_2^2 to q_2^1
b-loop at q_2^1.

final m/c —



ex. $L = (0+1)^*(00+11)$ — last 2 symbols are same.

This lang. is accepted by no DFA with fewer than 5 states.

We define a 3-state Mealy m/c.

Sequence of y's & n's emitted by M.m. corresponds to the sequence of accepting & nonaccepting states entered by a DFA on the same i/p,

however m.m. does not make an o/p prior to any i/p, while DFA rejects the string ϵ, as its initial state is nonfinal.

→ m/c is drawn on previous pages.

— o —

Examples →

Def^n Moore $(Q, \Sigma, \Delta, \delta, \lambda, q_0)$

Δ : o/p alphabet

λ : m/c f^n : $\lambda : Q \to \Delta$

① mod 3 for binary No.

Remainder values :	0	1	2
add bit 0 :	0	2	1
" 1 :	1	0	2

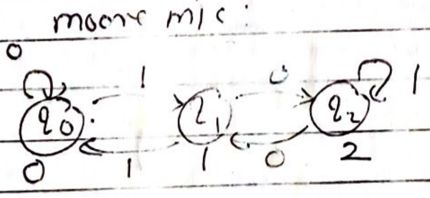
m/c f^n λ :

Q	q_0	q_1	q_2
Δ	0	1	2

Transi. table δ:

Q \ Σ	0	1
q_0	q_0	q_1
q_1	q_2	q_0
q_2	q_1	q_2

moore m/c :

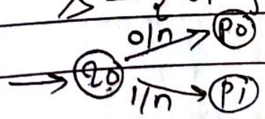


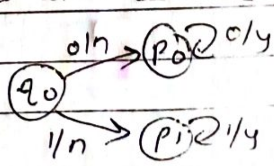
or $q_0/0$ $q_1/1$ $q_2/2$

Mealy : $\lambda : Q \times \Sigma \to \Delta$

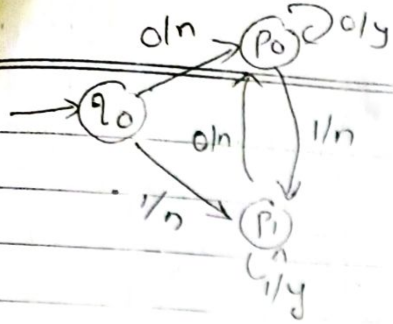
ex. lang. - words end with double letters. $\Sigma = \{0, 1\}$.

$\Delta = \{y, n\}$.





DATE: / /
PAGE NO.:

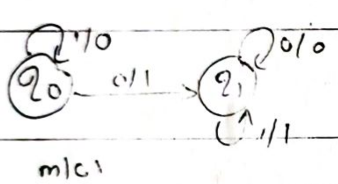


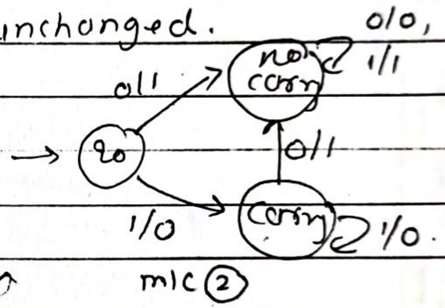
l(i/p str.) = l(o/p str.).

ex. mealy - incrementer i.e. bin. No. + 1.

~~1011~~ i/p	1011	·1011 i/p	1111
~~0100~~ 1's complement	+ 1	~~0100~~ o/p	+ 1
+ ~~1~~	1100	1100 o/p.	10000
0101 - 2's complement			

proc^n - read bit from LSB.
replace 1s by 0's.
" first 0 by 1.
keep remaining bits unchanged.





to consider carry generated ↗
for m/c (1) :-

δ:

$Q \backslash \Sigma$	0	1
q_0	q_1	q_0
q_1	q_1	q_1

λ:

$Q \backslash \Sigma$	0	1
q_0	1	0
q_1	0	1

see previous Moore to Mealy conversion —

$M_1 = (Q, \Sigma, \Delta, \delta, \lambda, q_0)$ - moore
$M_2 = (Q, \Sigma, \Delta, \delta, \lambda', q_0)$ - mealy.

$\lambda'(q, a) = \lambda(\delta(q, a))$. $\forall q \in Q, \forall a \in \Sigma$.

ex. moore for % 3

$\lambda'(q_0, 0) = \lambda(\delta(q_0, 0)) = \lambda(q_0) = 0$

$q_0, 1 = 1$ $\quad$ $q_2, 0 = 1$
$q_1, 0 = 2$ $\quad$ $q_2, 1 = 2$.
$q_1, 1 = 0$

equi. Mealy -