Date:

CS 205

Course contains. automata theory
- computability theory
- complexity theory

mathematical model related to computation.

- Show some functions are not computable
- segregate solvable and unsolvable problems wrt real world computers
- Build arithematic hierarchy

- P, NP, CoNP, P
- hierarchy among solvable based upon ~~complexity~~ (how easy / hard a problem is)

- interactable, problems taking lot of space or time
- provide evidence that certain problems are interactable even though we are not able to prove they are.

Computability theory + complexity theory gives lower bounds on the problem

approx algo
randomised algo

heuristic

AUTOMATA THEORY

Formal Language

(finite) alphabet $\xrightarrow{\text{denoted by}} \Sigma$

↳ a set of symbols (digits or letters)

eg. $\Sigma = \{a, b, c, d\}$

$\Sigma = \{a, b, c, d, 1, 5, 8\}$

String over a Σ

collection of alphabets from Σ

eg. $\Sigma = \{a, b, c, d\}$

string = acd, accd

eg. daca

daca } prefixes

daca } suffixes

language refers to formal language (throught the course)

language is a set of strings

$L = \{acd, accd, adcc\}$

$L = \{a, ad, dc, bcd\}$

$L = \{w \mid w \text{ has as many 'a's as 'b's}\}$

ab, abba, ...

$w = ab$ $\quad$ $w = abba$

$|w| = 2$ $\quad$ $|w| = 4$

epsilon $\varepsilon \rightarrow$ empty string (" ")

$|\varepsilon| = 0$

$L = \{ \varepsilon, acd, accd, adcc \}$

$L = \{ \}$ → 0 elements

$L = \{ \varepsilon \}$ → 1 element

Machine (M)

Language of a machine $L(M) = \{ , , , \ldots \}$

M recognises $L(M)$

M accepts $L(M)$

$$\text{string } w \rightarrow \boxed{M} \xrightarrow{\text{yes}} w \in L(M)$$
$$\boxed{M} \xrightarrow{\text{no}} w \notin L(M)$$

Advantages of languages and Machines

$$\begin{array}{cccccc} s_1 & s_2 & s_3 & s_4 & & \\ [\,0 & 0 & 1 & 1 & 0 & \ldots\,] \end{array}$$

s_1: not present in $L(M)$

s_3: present in $L(M)$

characteristic vector.

we can easily build characteristic vector in binary form.

language is used for compiling etc.

Machine is easy abstraction (yes/no) wrt languages.

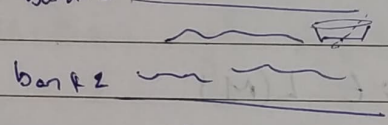
To start with.

- given L .. construct a machine M st L(M) = L
- given M .. determine the L(M)

Model of computation uses automata introducing transistion diagram. (similar to FSM)

M - man G → goat W - wolf C - cabbage

~~goat~~ bank 1 MWGC



to row the boat, man is needed.

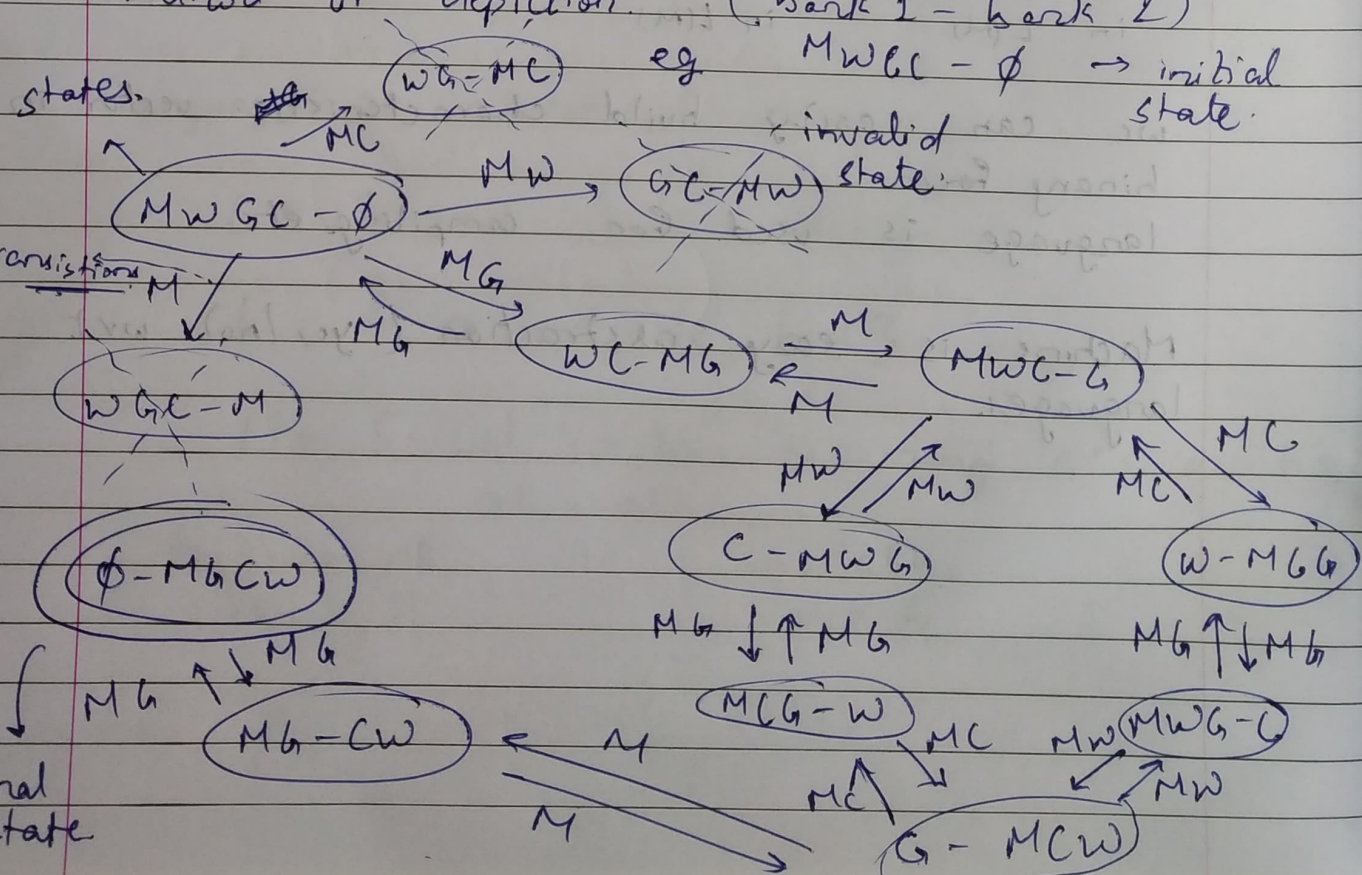
bank 2

If man leaves G, W alone then W eats G and if G, C are left alone then G eats C

Find minimum times man has to cross the river to have MWGC all on other bank

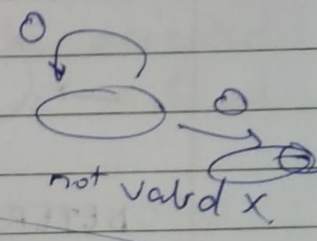
Method of depiction. (bank 1 - bank 2)

eg MWGC - ∅ → initial state.



Minimum 7 times crossing the river
Note, state, transition and transition diagram.

DETERMINISTIC → transition should have a single result state



FINITE AUTOMATA

↓ bcuz states are finite

bcuz we are giving state to state

$(Q, \Sigma, \delta, q_0, F)$ finite state.

Q → set of states

Σ → alphabets

δ → transitions

q_0 → start state

$F \subseteq Q$

transitions.

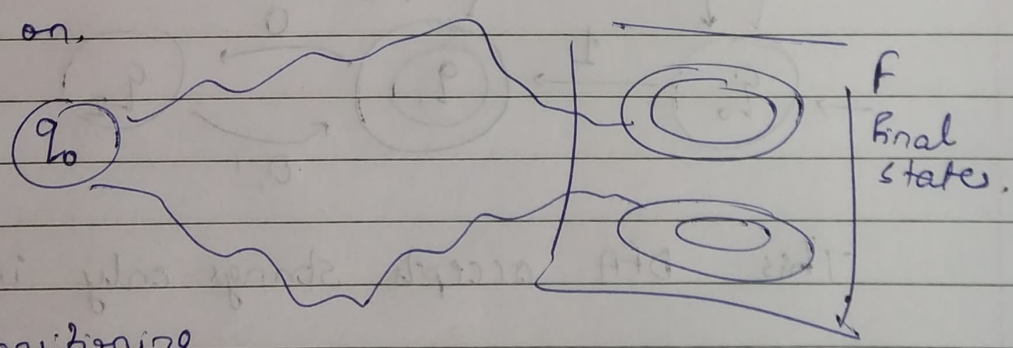
$\delta : Q \times \Sigma \to Q$

eg (a bit vague)

$\Sigma = \{0, 1\}$ $\quad$ w (input string) = | 0 | 0 | 1 | 1 | 0 | 0 |

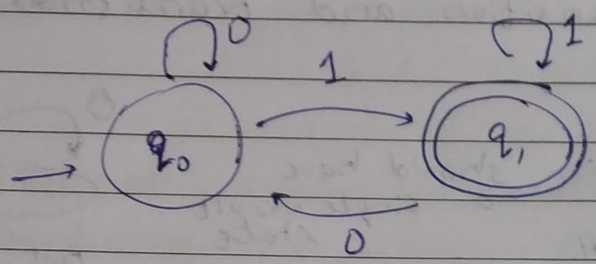
→ only read in forward direction cannot go back

Lets say, machine is in initial state. first it reads 0, goes to other state, again reads 0, goes to some other state and so on.



if upon input strings, ~~fin~~ if we reach "final states" then the string is accepted by the machine or else rejected.

eg. $L(M) = \{ w \mid w \text{ ends with } 1 \}$ $\Sigma = \{0, 1\}$



$q_1 \in F$

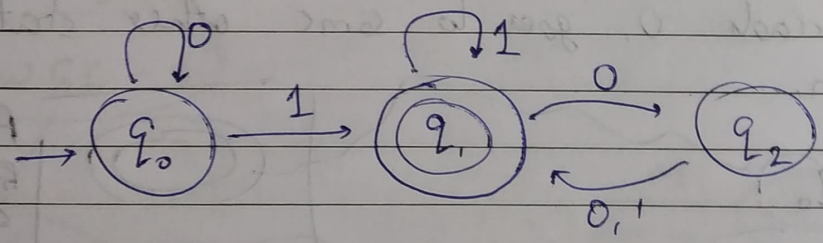
DETERMINISTIC FINITE AUTOMATA (DFA)

$\delta(q_0, 0) = q_0$
$\delta(q_0, 1) = q_1$ $F = q_1$
$\delta(q_1, 0) = q_0$
$\delta(q_1, 1) = q_1$

We can say this DFA is accepting $L(M)$

eg. $L(M) = \{ w \mid w$ contains atleast one "1" and even number of 0's follow the last one $\}$

after having DFA we must have well defined $(Q, \Sigma, \delta, q_0, F)$

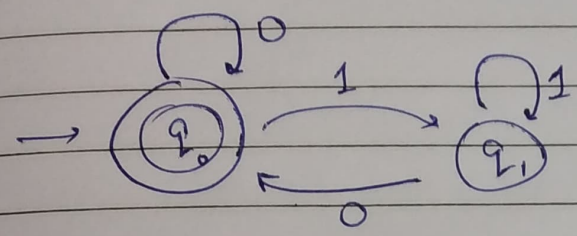


This DFA accepts strings only in $L(M)$

$\delta(q_2, 0) = q_1$ $\delta(q_0, 0) = q_0$ $\delta(q_1, 0) = q_2$
$\delta(q_2, 1) = q_1$ $\delta(q_0, 1) = q_1$ $\delta(q_1, 1) = q_1$

eg. $L(M) = \{ w \mid w \text{ is } \varepsilon \text{ or ends in } 0 \}$

empty string



ε must be accepted so q_0 is a final state.

another string 0011000.

in our model of computation, we only know current symbol of input, not ~~back~~ before and after input symbols are known

$Q = q_0, q_1$

$\delta(q_0, 0) = q_0$ $\quad$ $\delta(q_1, 0) = q_0$

$\delta(q_0, 1) = q_1$ $\quad$ $\delta(q_1, 1) = q_1$

$\Sigma = \{0, 1\}$ $\quad$ q_0, $\quad$ $F = \{q_0\}$.

We learnt, given a language form a DFA, try yourself DFA to language.