

Computability theory : problems that

can be solved by a computer.

q)

Computer cannot predict with 100%

accuracy the following problem's solution:

i) Proving correctness of a mathematical

statement

) result of futuristic events (result of

Today's game)

1932 - Alan Turing - Turing m/c - mathematical model.

problem solved by Turing m/c \Leftrightarrow solvable

by any m/c

→ Turing m/c is assumed to have ∞ memory

) models of m/c that can solve a problem

Simple m/c

calculator

(solve some problem)

Complex m/c

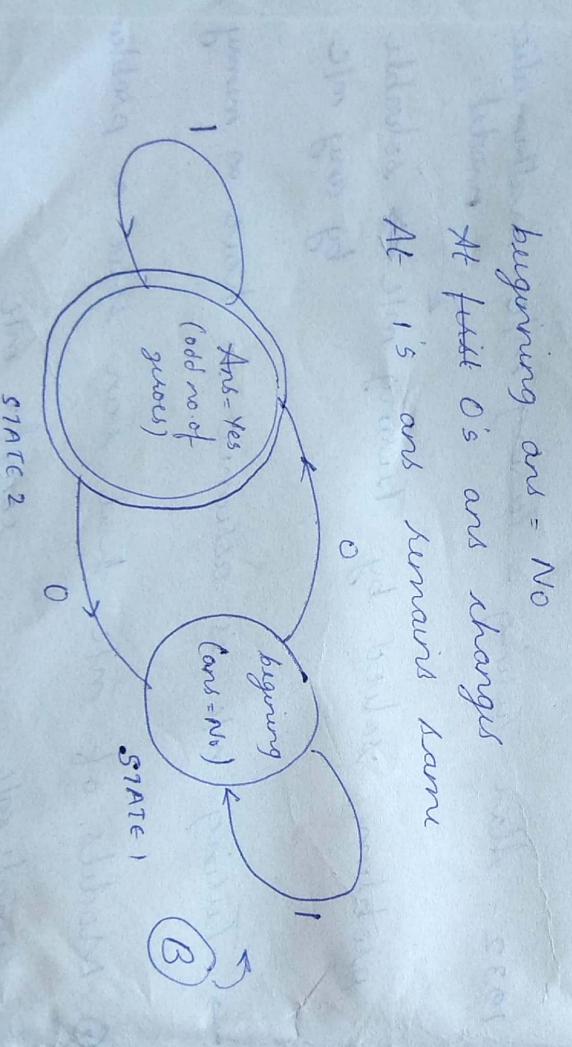
Turing m/c.

or
ent

Automata : Types of mathematical models of m/c

→ Design a m/c that gets a binary string
it gives 'yes' if it has odd no. of 1's
and otherwise 'no.'

A: here lets consider lang A as which
consists of all binary strings with
odd number of zeros.



— finite automata

model

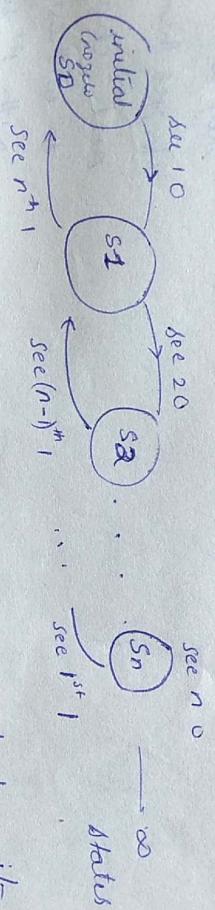
hue m/c B decides ~~var~~ ground

Given a string has 0, 1 or #, check if its of the form $w\#w$ ($: w - \text{any string}$)

ste
t₂: let A' = long has string of the form $w\#w$

w is starting with 0, 1 or ~~2~~

If the M/C doesn't need finite no. of states;



See n
if m/c returns to initial state, it

has same
 $\left(\begin{array}{cc} 00 & 11 \\ 01 & 10 \end{array} \right)$

followed by 'blank' symbols.

A diagram showing a vertical line representing a tape with binary digits (0s and 1s) written on it. A horizontal arrow labeled "tape head" points to the right, indicating the direction of reading or writing.

A Turing m/c has:

(i) # alone is ~~#~~ → ans = yes.
(ii) No # ⇒ ans = no (because one)

(m)

10...#...



Once you see has

compose next char
with 1st one
if same
if crosses both old

say ans = no

if yes then compa

next char after
without cross
with 1st symbol
from 1st without
cross.

skip

for example []
has added a char
Address should be

Turing m/c has:

- ① An input tape : An input string followed by blank symbols (\sqcup)

② tape head : initially points to beginning to tape

③ At a time tape head moves left or right by 1

④ Input alphabet Σ (here $0, 1$)

⑤ Tape alphabet Γ here $\{0, 1, \sqcup\}$

$$\Sigma \subseteq \Gamma \quad \emptyset \subseteq \Gamma$$

⑥ Finite set of states (works for any Tm)

⑦ q_0 - initial state

q_{accept} - accept
 q_{reject} - reject state

⑧ transition function ("moves" / "slips") new state
new symbol

$q(q, a) \xrightarrow{\text{current symbol}} (q', b, \delta)$
current state
of movement

Q

$\delta = \{q_1, q_2, q_3, q_{au}, q_{ay}\}$

$\Gamma = \{0, 1, X, \sqcup\} - \text{symbol that can be seen in tape}$

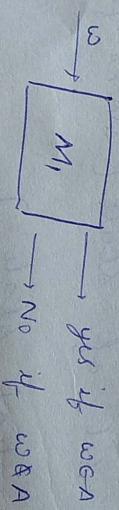
$$f: Q \times \Gamma \rightarrow Q \times \Gamma \times D$$

$\Rightarrow L$

we need to define transition for each state & symbol. If not its assume to stop rule of go to q_{ay} .

(Once rejected tape symbol & direction are not important)

M_1 decides lang A



Now, in some cases this type of m/c may not be designed even if its computable. For those cases:

$$\xrightarrow{\omega} [M_2] \xrightarrow{\quad} \text{yes if } w \in A \\ \text{it doesn't say 'yes' if } w \notin A \\ (\text{i.e., it doesn't say 'no'})$$

Q. Q

①

②

③

if

We

If $\omega \in A$, it may enter a loop and
does not stop.

on t

\Rightarrow lang A is Turing decidable if $\exists M$ which

decides A (recursively)

\nexists lang A is Turing recognizable if $\exists M'$
which recognizes lang A (recursively enumerable)

recognizable

1

set of

Turing decidable \subseteq set of Turing

recognizable

lang

lang

lang

$\omega \in \{0,1\}^\omega$

Q. $\{ \begin{array}{l} \omega \# \omega \\ 0^n 1^n \end{array} \} \quad \omega \in \{0,1\}^\omega$

design m/c that checks these lang.

decides

O^{2n}

We say a turing machine accepts a lang.
if it reaches $\$$ accept at end of tape

if it reaches $\$$ accept at end of tape

turing m/c stop @ q_{right}/q_{accept} no moves
from q_{right}/q_{accept}

Unifor

if a transition is not defined, it
usually moves to q_{reject}

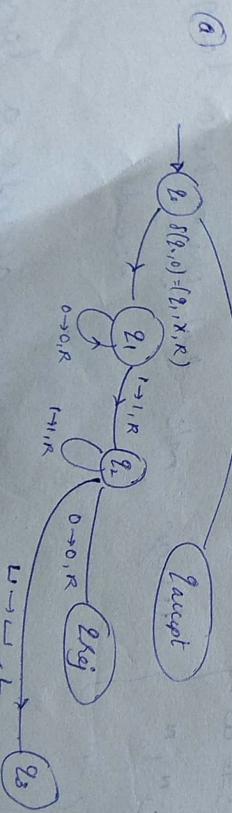
② $0^n 1^n = A$.

a) scan from left \rightarrow right

b) If $T \neq 0$ after 1st 1, reject

c) Come back to left and cross off 0 and
go right if exists of corresponding 1.
If they both end up at same time,
accept.

$$\sqcup, \text{then } R \quad \delta(q_0, \sqcup) = (q_{accept}, \sqcup, R)$$



Now, if $\delta(q_0, 0) = (q_0, 0, R)$ then if $\frac{1}{M}$ is $\boxed{00\sqcup}$
it accepts.

Now, or
we ha
follow



change $0 \rightarrow x$ to know tape head $f(q_0, \sqcup) = (q_{accept}, 0, R)$

$$f(q_0, 0) = (q_1, x, R)$$

$$f(q_1, 0) = (q_2, 0, R)$$

Undefined transition $\rightarrow q_{reject}$

$$f(q_2, 0) = (q_{reject}, 0, R)$$

Now, once it reaches \sqcup in q_2

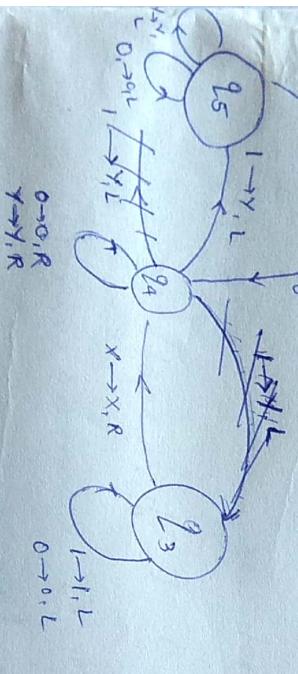
we have to move back and

be

follow step 6

$\sqcup \rightarrow \sqcup, \text{Accept}$

q_{accept}



wrong
if extra 1,
reach q_2 &
 $reject$

if extra 0,
 $reject$ at q_4

$O(n^2)$

① We considered single tape determinisitic turing m/c.

Now what if we had 2 tapes

an t-

1 with input & 1 blank

be

1 101011111111

2 111111111111

① scan through tape 1, copy 0s to un

2nd tape . if 7 0 after 1st 1, reject

only

(forward)

② now compare 1's in 1st tape with 0s

in 2 (backward)

If both finish together, accept.

else reject.

⇒ O(n)

this is called multi tape
turing m/c (k-tape for k tapes)

Ans : (i) Tape 1 has 'p' followed by 'l'

Now,

- Tape 2 - k has 'l' initially
- Set of states $\{q_{\text{start}}, q_p, q_l, q_e\} \subseteq \Gamma$

L

Now, transition fn / moves,

single tape : $f(\varrho, a) \rightarrow (\varrho, b, D)$

k-tape : $f(\underbrace{\varrho, a_1, a_2, \dots, a_k}_{k \text{ tapes}}) = (\underbrace{\varrho, b_1, b_2, \dots, b_k}_{k \text{ tapes}}, \underbrace{D, D_1, D_2, \dots, D_k}_{k \text{ tapes}}$

cur. state
current symbols

$a_i, b_i \in \Gamma$

$\forall i = 1 \rightarrow k$

$\Rightarrow f : \mathcal{Q} \times \overbrace{\Gamma \times \dots \times \Gamma}^{\Gamma^k} \rightarrow \mathcal{Q} \times \overbrace{\Gamma \times \Gamma \times \dots \times \Gamma}^{\Gamma^k} \times D^k$

Theorem

A: N

Γ -tape alphabet

Proof

(ii) it can also be seen as

$$f(\varrho, a, a_1) = (\varrho, (b_1, D_1), (b_2, D_2))$$

$$\Rightarrow f(\varrho \times \Gamma^k) \rightarrow (\varrho \times (\Gamma \times \{L, R\})^k)$$

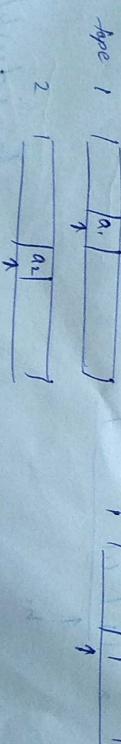
Let f.

Now, Are multi tape turing m/c better
in terms of computational power?
(i.e. do something in multi tape that can't
be done in single tape?)

A: No, Anything solved using multi tape can be
solved in a single tape.

Theorem: If a lang. A is decidable (
recognisable) with a k-tape turing
m/c, then A is decidable (recogni-
sable) with a single tape turing m/c
with a single tape turing m/c
(vice - versa is obvious)

Proof: k type turing m/c (m) single tape (s)



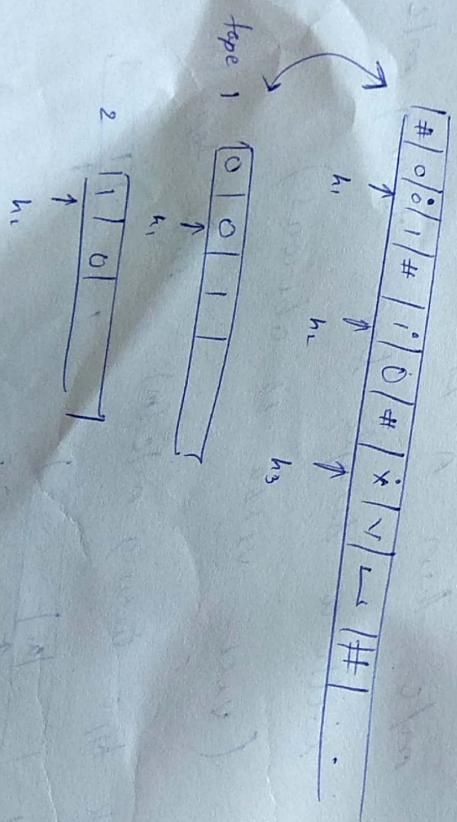
Let $f(g, a_1, a_2, \dots, a_k) = (x, b_1, b_2, \dots, b_n, D, D_1, D_2, \dots, D_k)$ (where g

Now, we can combine all the k tapes into a single tape one after another separated by a delimiter of perform transition on it.

Now, how to keep track of k heads in single tape, ans: use dotted symbol.

$$\text{if } \Gamma = \{a, b, x, y, \sqcup\}$$

$$\text{then } \Gamma' = \{a, b, x, y, \sqcup, a^i, b^i, x^i, y^i, \sqcup^i\}$$



Now, scan the tape one, to identify current symbol a_1, a_2, \dots, a_k , (i^{th} dotted symbol denoted a_i)

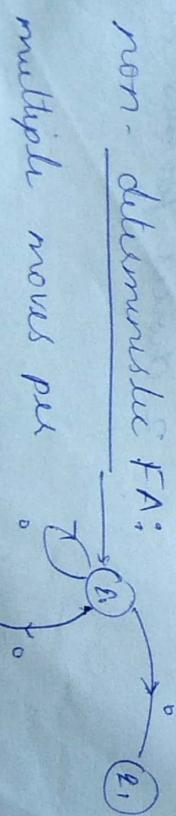
- Now, making changes in symbols or adding symbols may require more than 1 moves in single tape
- if we add a symbol to tape 2, then in single tape, we need to shift all symbols after 2nd dotted symbol to right
- \therefore many moves are required

$$\begin{array}{ccc} O(F(n)) & \xrightarrow{\quad\quad} & O(F(n)^k) \\ \underbrace{\quad\quad\quad}_{k \text{ tape}} & & \underbrace{\quad\quad\quad}_{\text{single tape}} \end{array}$$

Non-deterministic turing m/c



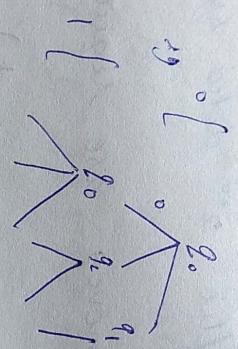
1 move per symbol $\in \Sigma$



situation

then how do we say a turing is
accepted or not in non-deterministic

i.e., till to making all possible choice
parallelly



So, if atleast one of the process
reach accept state, we accept the turing

. : In non-deterministic turing m/c
we have multiple choice of moves

$$S(q, a) = \underbrace{\{(r_1, b_1, D_1), (r_2, b_2, D_2)\}}$$

2 choices for his situation

In turing m/c if 'p' is 0001, we can't say whether its accepted or rejected by looking at states at 4th move as turing m/c may take multiple moves for a single operation

∴ If any path leads to 1/ends at

concept, then we accept the string.

But can't say, if any path leads to

q₄ ⇒ string is rejected

Since some path may never terminate, here we can't say if it eventually reach q₄ / q₅.

If all path leads to q₅, we reject the

string

If we have set of states of each tape
have its own fns?

$$Q_1 \xrightarrow{\text{fns}} S_1(Q_1, a_1) \rightarrow (q_1, b_1, D_1)$$

$$Q_2 \xrightarrow{\text{fns}} S_2(Q_2) \rightarrow ()$$

$$Q_n \xrightarrow{\text{fns}} S_n(Q_n) \rightarrow ()$$

Now, can we convert it into normal
multi tape turing machine? Yes.

Let $\langle q_1, q_2, \dots, q_n \rangle$ be a single state

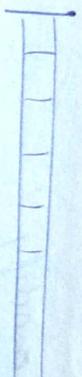
$$\in (Q_1 \times Q_2 \times \dots \times Q_n)$$

$$\therefore f(\underbrace{\langle q_1, q_2, \dots, q_n \rangle}_{n \text{ states}}, a_1, a_2, \dots, a_n) = (\langle x_1, x_2, \dots, x_n \rangle, b_1, b_2, \dots, b_n, D_n)$$

assumed as 1

state of $Q = Q_1 \times Q_2 \times \dots \times Q_n$

Also anything done in 2-way infinite
tape turing m/c can be done in
single tape. how?



single

How to make effects on this half reflect in single tape without affecting other half.

- A raw 2 way tape can be wrapped around to make a 2-track tape that carry

2 symbol in one cell.

$$\text{new } T^2 = r \times r$$

a_1	a_1	a_3	a_4
a_5	a_6	a_7	a_8

\therefore of the form $SL(a_1, a_5) =$

Now, how to modify

- ① Transition for δ
(if needed)
- ② Set of states

① Now how do we know if had made
a move in right half or left half
of 2-way tape?

* put it along ℓ in $(q', 0)$

\nearrow right
new states $Q' = Q \times \{0, 1\}$ left

(case (i)) : original move in right half to right

$$\delta(q, a_1) = (r, b_1, R^R)$$

$$\text{then new fn} \Rightarrow \delta'((q, 0), (a_1))$$

$$= ((r, 0), (b_1))$$

(case (ii)) : original move in left half
towards left

$$\delta(q_2, a_2) = (r, b_2, \leftarrow)$$

$$\Rightarrow \delta'((q_2, 1), \begin{pmatrix} a_2 \\ a_3 \end{pmatrix}) = ((r, 1), \begin{pmatrix} b_2 \\ a_3 \end{pmatrix})$$

case (iii): a] Original move in right half .

Not in left most cell, dis. left.

here let symbol with '̄' above (y: a_i)

represent element at the beginning

of left / right track

$$eg. \quad \delta'((q_2, 0), \begin{pmatrix} a_2 \\ a_3 \end{pmatrix}) = ((r, 0), \begin{pmatrix} a_i \\ a_3 \end{pmatrix})$$

(still but at
on right leftmost
half and as
element is a_i ,

11^th case iv: a] Original move in left half
not in rightmost end, to right

10

For case (iii) to $\phi_{(iv)}$, to move from left half to right ϕ via versa, how to

represent that move using
new transition for?

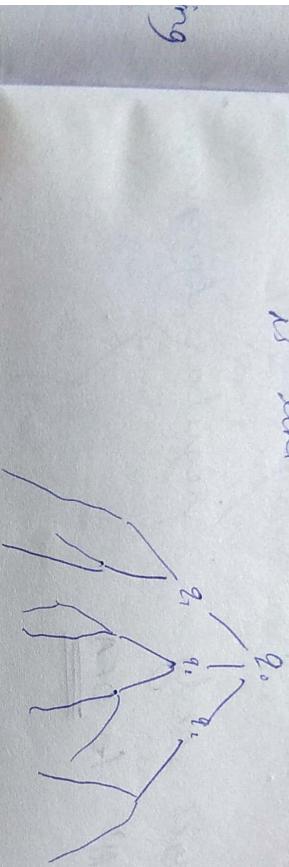
A: Use ' as symbol above dot in lightest cell of ϕ as that above element in leftmost cell.

DF

208

4

Now, if our non-det. tuning m/c
is like



how to traverse each process.

DFS /BFS?

Can't use DFS because if the process
we chose never ends, the traversal also
never ends & we don't get O/P.

If we use BFS, we go level by level.
if any of the states in a level is
except, we accept it or else we go to
next level.

\Rightarrow A turing m/c with right of stay pulse.

as direction.

A: languages only

$M \rightarrow$ DFA

"regular language"

Give a proof: DFA \Leftrightarrow above, turing m/c

both sides

Q. Design a turing m/c to decide following

languages

(a) $0^{2^n} (n \geq 0)$

0
00
0000

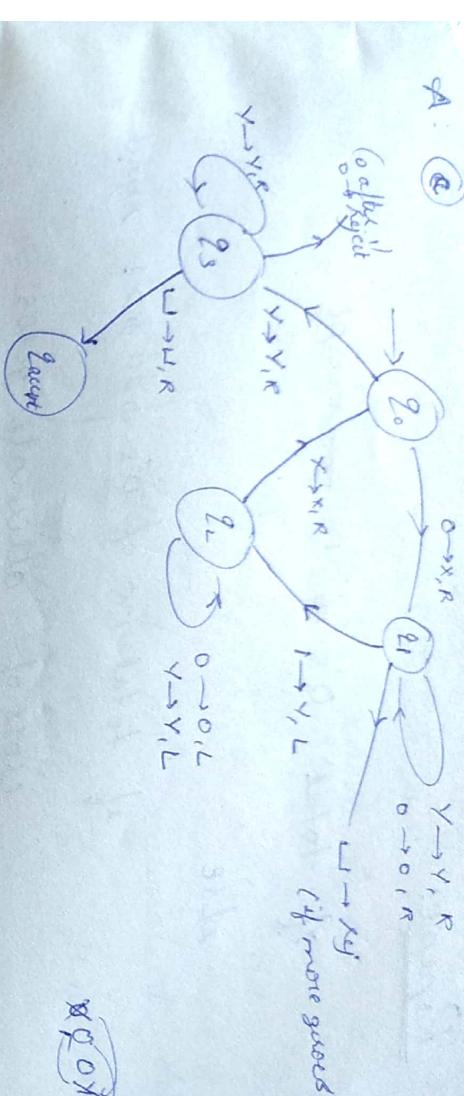
↑
more



b) w reflected by w'g 001/100

c) $0^n 1^n n \geq 0$

d) string is palindrome



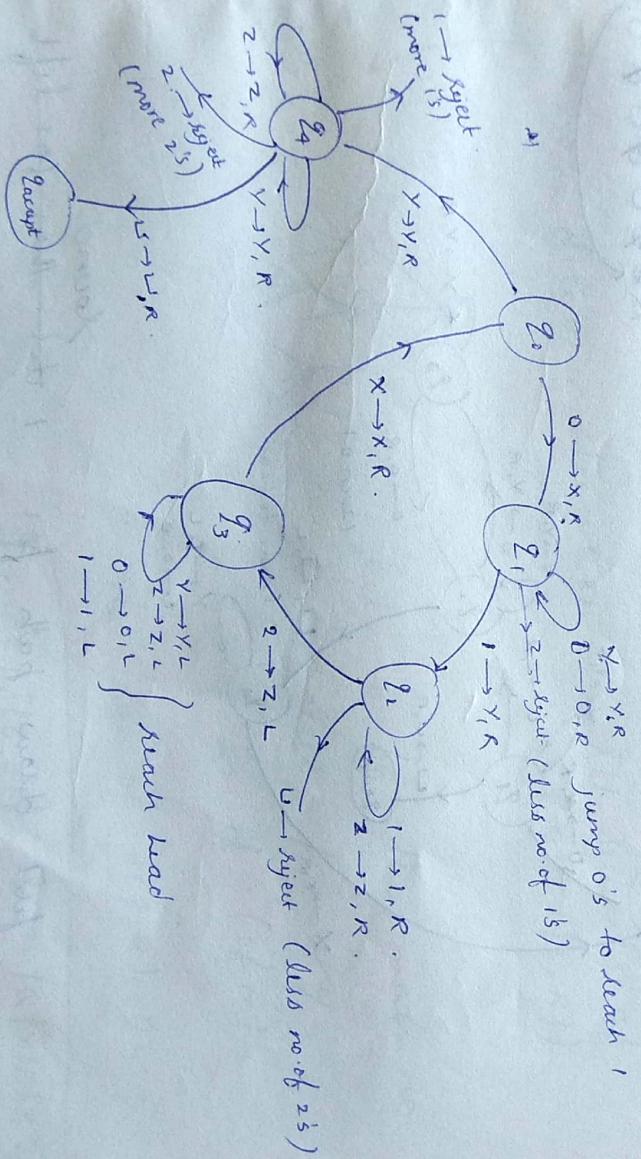
\therefore for $0^n, 1^n 2^n$ add extra symbol

$Y \rightarrow Y, R$

$0 \rightarrow 0, R$ jump 0's to search 1

$2 \rightarrow 2, \text{reject}$ (less no. of 1's)

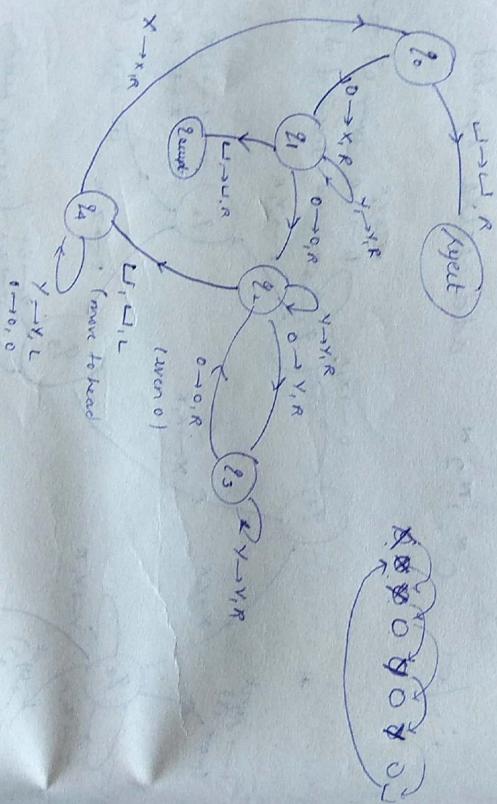
a)



(a) Repeat:

if total no. 0's is 1 → accept
else

if total no. of 0's odd & > 1 reject
else of attenuate zeros.



first draw path for 1 sound,
then for next

C_2 rounds alternating → multiple of 4
at end)

Non - deterministic turing m/c.

- recognize lang A

- If N is a non-det. turing m/c, if an equivalent deterministic turing m/c M that recognizes the same lang. A.

deterministic : $\{q \times r \rightarrow q \times r \times \{l, R\}\}$.

non-deterministic : $\{q \times r \rightarrow \text{all subsets of } (\{q \times r \times \{l, R\})\}$

PROOF:

Let 'b' be the max. no. of choices for any move in d (of N)

• Max size of $S(q, a)$ or

$$\max |S(q, a)| = b = |\{q\}| \times |\{r\}| \times 2$$

as $S(q, a) = \frac{\{q \times r \times \{l, R\}}{|\{q\}| \times |\{r\}| \times 2}$
possibility

M - deterministic

3-tape turing m/c $b \in \mathbb{Z}^+$

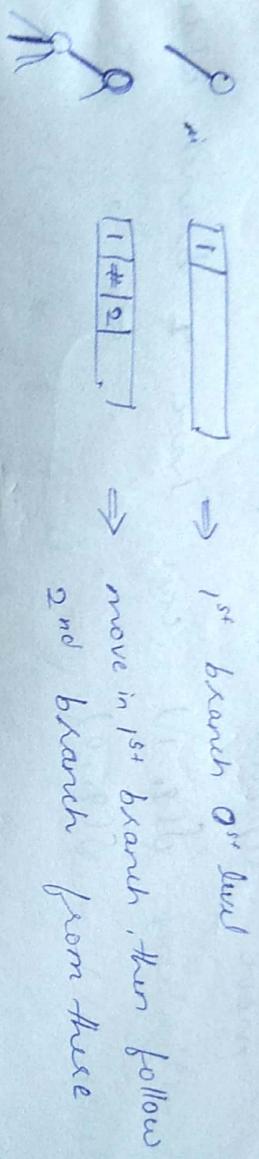
Tape 1 : $\boxed{-1 \downarrow 1 \uparrow 1}$ → input tape
→ never changed

Tape 2 : $\boxed{\text{moves occur here}}$ → simulation /
execution tape
 \hookrightarrow in NFA

Tape 3 : $\boxed{1}$ → current branch of
execution tape /
address tape

Assume we go breadth first in NFA
of our number $1, 2, \dots$ for each branch
from a state. This branch no. is in Tape 3
To represent 2nd level branch we use
 $(1, 2)$ like representation

\therefore Tape 3 is like a dictionary with
alphabets $\in \{1, 2, \dots, b\}$



\therefore in kth level we have

$$(i_1, i_2, \dots, i_k) \quad ij \# j \in \{1, 2, \dots, b\}$$

\Rightarrow ijth choice in jth move.

Now based on tape 3 we make moves in 2

after each move we make value of tape 2
to rapid using tape 1 to carry out the
move from 1st move to current
move sequentially. And if any move
not done of move. Then stop & accept

reaches Paupt, then stop & accept

\therefore Non-det. turing m/c can be implemented

using a 3 tape det. turing m/c
which return can be done using
single tape turing m/c.

here the proof:

Actual no. of branches / choices for a

move depends on 1/p

e.g. $S(2, 1)$

$S(2, 0)$

"

"

"

"

"

"

"

"

"

"

"

"

"

Now, if a move for tape 3 is not there
(tape 3 updates through all combination
of branches) just jump to next move.

Note: We can use any representation

in a turing m/c

for Σ^* of number

→ for Σ^* of basis

e.g. $s = 1111$ (unary representation)

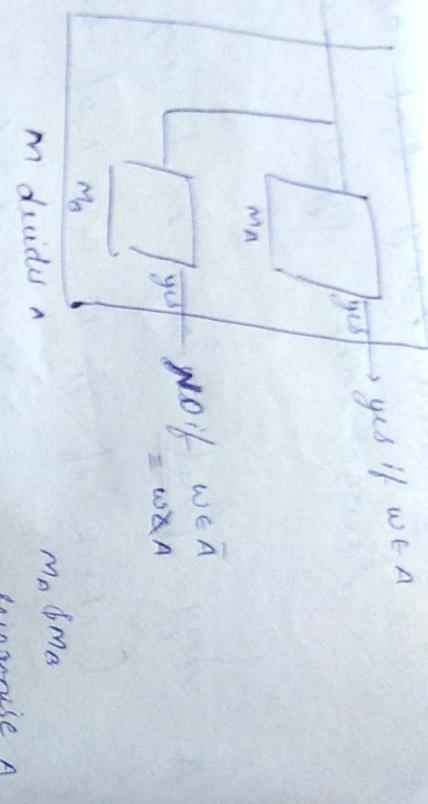
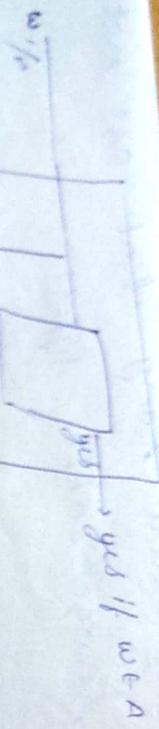
- 0101 (binary " ") .

Undecidability

- what lang are Turing decidable
- Are there lang that are not decidable? (yes)

→ Are there lang. that are not recognizable (yes)

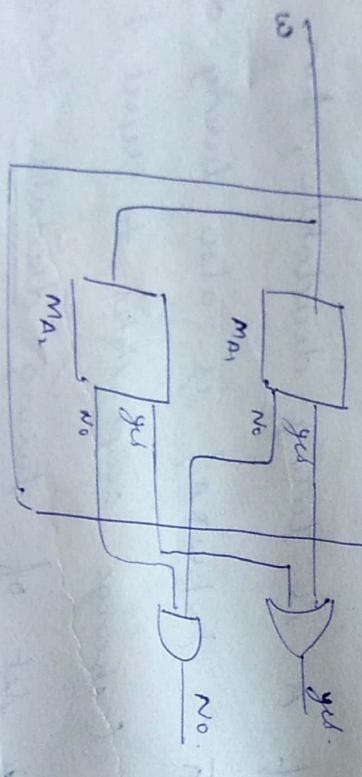
- A: Any finitely lang. is turing decidable
(check if there is any 1 of them (recursively))
of recognizable also (recursively enumerable)
- ⇒ A turing m/c that recognizes A and halts on all 1/pis is turing decidable
- ⇒ If A is turing decidable, then
 - ① $\bar{A} = \{\omega | \omega \notin A\}$ is also turing decidable
(change accept status to reject & vice versa)
i. set of turing decidable lang is closed under complement
 - If A & \bar{A} - are turing recognizable then A is turing decidable



\rightarrow If $A_1 \neq A_2$ are decidable lang.

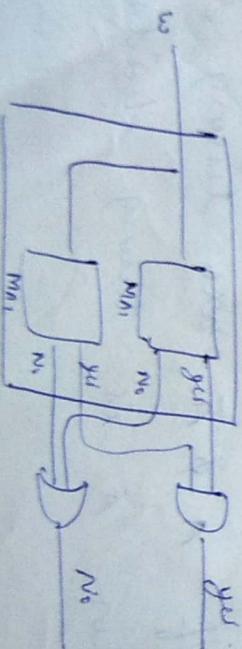
① Then $A_1 \cup A_2$ is turing decidable

except if either A_1 / A_2 is empty



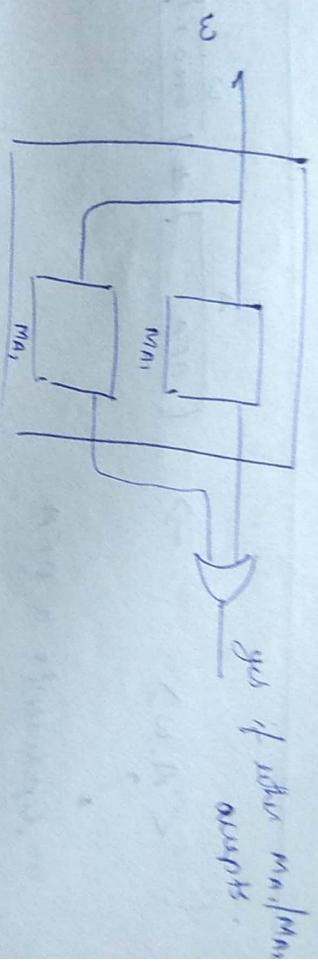
m₁

② Then $A_1 \cap A_2$ is also turing decidable



\Rightarrow If A_1, φ_{A_2} are Turing recognizable

- ① $A_1 \cup A_2$ is Turing recognizable

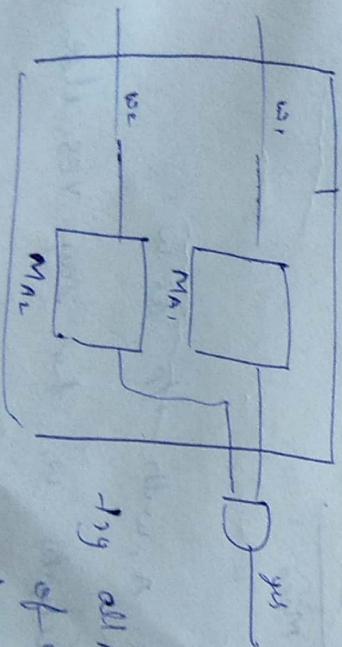


- ② $A_1 \cap A_2$ is Turing recognizable

in accept if M_{A_1}, M_{A_2} accepts
(don't worry about 'no')

\Rightarrow If A_1, φ_{A_2} are decidable / recognizable

A_1 concatenated with A_2 ($A_1 A_2$) is Turing
decidable / recognizable



try all possible combination
of $w_1, w_2 = w$ & check
if atleast 1 gives yes

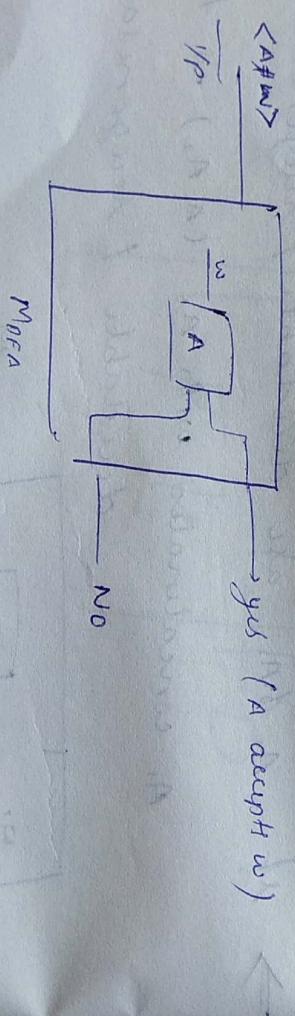
pass the head of tape when machine reaches q_{acc} .
 To make diff both accepts, then accept w

$$\langle A, w \rangle \rightarrow \boxed{A \text{ DFA}} \# \boxed{\overbrace{01001}^w}$$

\downarrow
represents a DFA.

\Rightarrow does A accepts w .

$$L_{DFA} = \left\{ \langle A, w \rangle : A \text{ accepts } w \right\} \xrightarrow{\text{using}} \text{decidable}$$



Algorithm:

- Run DFA A , with $1/\rho$ as w
- If A accepts w , then say YES, else NO

$$L_{Tm} = \{ \langle M, w \rangle : M \text{ accepts } w \} \rightarrow \text{universal lang}$$

M is a turing m/c.

L_m is not turing decidable

(m may be non terminating for some w)

Theorem: L_m is undecidable (no way to design it)...

Proof: ① We define a lang. such that there is no turing m/c that recognises that lang. (diagonalisation technique)
 (if L_m is decidable, then above statement is false)

Idea: ① "Number" all turing m/c s.t each number represent a turing m/c

② Number each 1/0 string (over $\{0, 1\}$)

How to number 1/0 string

$y:$	0	10
	00	1110
	010	?
	0001	

Let $w_i = \underbrace{0}_1 \} \text{ length } 1 \text{ string}$

$w_2 = \underbrace{00}_2 \} \text{ length } 2 \text{ string}$

$w_3 = \underbrace{01}_2 \} \text{ length } 2 \text{ string}$

$w_5 = \underbrace{10}_2 \} \text{ length } 2 \text{ string}$

$w_6 = \underbrace{11}_2 \} \text{ length } 2 \text{ string}$

Exercise: Given a number i , what is w_i

$$\Rightarrow \lfloor \frac{\log(i+1)}{\text{no. of alphabets}} \rfloor$$

- ⑥ Given a string w_i , ($\forall i$ we know its length : $\#$ number n) how to find i s.t. $w = w_i$?

$$i = (2^{k-2}) + (n+1)$$

Flow to number fusion m/c.

$$eg: f(\varrho_0, 0) = (\varrho_1, \sqcup, R)$$

$$f(\varrho_1, 1) = (\varrho_3, \sqcup, R)$$

$$f(\varrho_3, 0) = (\varrho_{\text{out}}, 0, R)$$

$$f(\varrho_0, 1) = (\varrho_0, \sqcup, R)$$

Find order of state

$$\begin{matrix} & \varrho_1 & \varrho_2 & \varrho_3 & \varrho_4 \dots \\ \xrightarrow{\quad} & \downarrow & \downarrow & \downarrow & \downarrow \\ \text{Kurane states s.t.} & \varrho_0 & \varrho_{\text{out}} & \varrho_1 & \varrho_2 \\ & \varrho_1 & \varrho_2 & \varrho_3 & \varrho_4 \end{matrix}$$

$$\Rightarrow f(\varrho_1, 0) = (\varrho_3, \sqcup, R)$$

Also let

$$\begin{matrix} \varrho_1 & \varrho_2 \\ \downarrow & \downarrow \\ D_1 & D_2 \end{matrix}$$

$$\begin{aligned} f(\varrho_3, 1) &= (\varrho_4, \sqcup, R) \\ f(\varrho_4, 0) &= (\varrho_2, \sqcup, R) \\ f(\varrho_1, 1) &= (\varrho_1, \sqcup, R) \end{aligned}$$

$$\Rightarrow f(\varrho_1, x_1) = (\varrho_3, x_3, D_2)$$

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

\therefore if we have s numbers i, j, k, l, b_m base

\therefore if we have s numbers i, j, k, l, b_m base
then it corresponds to the move

$$S(\varrho_i, x_j) = (\varrho_k, x_l, D_m)$$

$$\Rightarrow \overbrace{[0^i 1 0^j 1 0^k 1 0^l 1 0^m]}^{\downarrow} \rightarrow \text{code}$$

\downarrow
 \circ times followed by 1 (as delimiter)

$$\therefore S(\varrho_i, x_j) = (\varrho_k, x_l, D_m)$$

$$\Rightarrow 01010001000100$$

\therefore We write code for turing m/c as

$$111 < \text{move}_1 > 11 < \text{move}_2 > 11 < \text{move}_3 > 11 \dots$$

 \downarrow
 2^{15}
as delimiter

\therefore if we arrange the moves, we could
get diff number for same turing m/c

\therefore from a given no. we can get the moves
of a turing m/c.

back to proof: Defining a lang. s.t there is no T.M. that recognise that lang.

Lang.

Put turing m/c & strings in order to get a

Machine	w₁	w₂	w₃	w₄	w₅	w₆	w₇	w₈	w₉	w₁₀	w₁₁	w₁₂	w₁₃	w₁₄	w₁₅	w₁₆	w₁₇	w₁₈	w₁₉	w₂₀	w₂₁	w₂₂	w₂₃	w₂₄	w₂₅	w₂₆	w₂₇	w₂₈	w₂₉	w₃₀	w₃₁	w₃₂	w₃₃	w₃₄	w₃₅	w₃₆	w₃₇	w₃₈	w₃₉	w₄₀	w₄₁	w₄₂	w₄₃	w₄₄	w₄₅	w₄₆	w₄₇	w₄₈	w₄₉	w₅₀	w₅₁	w₅₂	w₅₃	w₅₄	w₅₅	w₅₆	w₅₇	w₅₈	w₅₉	w₆₀	w₆₁	w₆₂	w₆₃	w₆₄	w₆₅	w₆₆	w₆₇	w₆₈	w₆₉	w₇₀	w₇₁	w₇₂	w₇₃	w₇₄	w₇₅	w₇₆	w₇₇	w₇₈	w₇₉	w₈₀	w₈₁	w₈₂	w₈₃	w₈₄	w₈₅	w₈₆	w₈₇	w₈₈	w₈₉	w₉₀	w₉₁	w₉₂	w₉₃	w₉₄	w₉₅	w₉₆	w₉₇	w₉₈	w₉₉	w₁₀₀	w₁₀₁	w₁₀₂	w₁₀₃	w₁₀₄	w₁₀₅	w₁₀₆	w₁₀₇	w₁₀₈	w₁₀₉	w₁₁₀	w₁₁₁	w₁₁₂	w₁₁₃	w₁₁₄	w₁₁₅	w₁₁₆	w₁₁₇	w₁₁₈	w₁₁₉	w₁₂₀	w₁₂₁	w₁₂₂	w₁₂₃	w₁₂₄	w₁₂₅	w₁₂₆	w₁₂₇	w₁₂₈	w₁₂₉	w₁₃₀	w₁₃₁	w₁₃₂	w₁₃₃	w₁₃₄	w₁₃₅	w₁₃₆	w₁₃₇	w₁₃₈	w₁₃₉	w₁₄₀	w₁₄₁	w₁₄₂	w₁₄₃	w₁₄₄	w₁₄₅	w₁₄₆	w₁₄₇	w₁₄₈	w₁₄₉	w₁₅₀	w₁₅₁	w₁₅₂	w₁₅₃	w₁₅₄	w₁₅₅	w₁₅₆	w₁₅₇	w₁₅₈	w₁₅₉	w₁₆₀	w₁₆₁	w₁₆₂	w₁₆₃	w₁₆₄	w₁₆₅	w₁₆₆	w₁₆₇	w₁₆₈	w₁₆₉	w₁₇₀	w₁₇₁	w₁₇₂	w₁₇₃	w₁₇₄	w₁₇₅	w₁₇₆	w₁₇₇	w₁₇₈	w₁₇₉	w₁₈₀	w₁₈₁	w₁₈₂	w₁₈₃	w₁₈₄	w₁₈₅	w₁₈₆	w₁₈₇	w₁₈₈	w₁₈₉	w₁₉₀	w₁₉₁	w₁₉₂	w₁₉₃	w₁₉₄	w₁₉₅	w₁₉₆	w₁₉₇	w₁₉₈	w₁₉₉	w₂₀₀	w₂₀₁	w₂₀₂	w₂₀₃	w₂₀₄	w₂₀₅	w₂₀₆	w₂₀₇	w₂₀₈	w₂₀₉	w₂₁₀	w₂₁₁	w₂₁₂	w₂₁₃	w₂₁₄	w₂₁₅	w₂₁₆	w₂₁₇	w₂₁₈	w₂₁₉	w₂₂₀	w₂₂₁	w₂₂₂	w₂₂₃	w₂₂₄	w₂₂₅	w₂₂₆	w₂₂₇	w₂₂₈	w₂₂₉	w₂₃₀	w₂₃₁	w₂₃₂	w₂₃₃	w₂₃₄	w₂₃₅	w₂₃₆	w₂₃₇	w₂₃₈	w₂₃₉	w₂₄₀	w₂₄₁	w₂₄₂	w₂₄₃	w₂₄₄	w₂₄₅	w₂₄₆	w₂₄₇	w₂₄₈	w₂₄₉	w₂₅₀	w₂₅₁	w₂₅₂	w₂₅₃	w₂₅₄	w₂₅₅	w₂₅₆	w₂₅₇	w₂₅₈	w₂₅₉	w₂₆₀	w₂₆₁	w₂₆₂	w₂₆₃	w₂₆₄	w₂₆₅	w₂₆₆	w₂₆₇	w₂₆₈	w₂₆₉	w₂₇₀	w₂₇₁	w₂₇₂	w₂₇₃	w₂₇₄	w₂₇₅	w₂₇₆	w₂₇₇	w₂₇₈	w₂₇₉	w₂₈₀	w₂₈₁	w₂₈₂	w₂₈₃	w₂₈₄	w₂₈₅	w₂₈₆	w₂₈₇	w₂₈₈	w₂₈₉	w₂₉₀	w₂₉₁	w₂₉₂	w₂₉₃	w₂₉₄	w₂₉₅	w₂₉₆	w₂₉₇	w₂₉₈	w₂₉₉	w₃₀₀	w₃₀₁	w₃₀₂	w₃₀₃	w₃₀₄	w₃₀₅	w₃₀₆	w₃₀₇	w₃₀₈	w₃₀₉	w₃₁₀	w₃₁₁	w₃₁₂	w₃₁₃	w₃₁₄	w₃₁₅	w₃₁₆	w₃₁₇	w₃₁₈	w₃₁₉	w₃₂₀	w₃₂₁	w₃₂₂	w₃₂₃	w₃₂₄	w₃₂₅	w₃₂₆	w₃₂₇	w₃₂₈	w₃₂₉	w₃₃₀	w₃₃₁	w₃₃₂	w₃₃₃	w₃₃₄	w₃₃₅	w₃₃₆	w₃₃₇	w₃₃₈	w₃₃₉	w₃₄₀	w₃₄₁	w₃₄₂	w₃₄₃	w₃₄₄	w₃₄₅	w₃₄₆	w₃₄₇	w₃₄₈	w₃₄₉	w₃₅₀	w₃₅₁	w₃₅₂	w₃₅₃	w₃₅₄	w₃₅₅	w₃₅₆	w₃₅₇	w₃₅₈	w₃₅₉	w₃₆₀	w₃₆₁	w₃₆₂	w₃₆₃	w₃₆₄	w₃₆₅	w₃₆₆	w₃₆₇	w₃₆₈	w₃₆₉	w₃₇₀	w₃₇₁	w₃₇₂	w₃₇₃	w₃₇₄	w₃₇₅	w₃₇₆	w₃₇₇	w₃₇₈	w₃₇₉	w₃₈₀	w₃₈₁	w₃₈₂	w₃₈₃	w₃₈₄	w₃₈₅	w₃₈₆	w₃₈₇	w₃₈₈	w₃₈₉	w₃₉₀	w₃₉₁	w₃₉₂	w₃₉₃	w₃₉₄	w₃₉₅	w₃₉₆	w₃₉₇	w₃₉₈	w₃₉₉	w₄₀₀	w₄₀₁	w₄₀₂	w₄₀₃	w₄₀₄	w₄₀₅	w₄₀₆	w₄₀₇	w₄₀₈	w₄₀₉	w₄₁₀	w₄₁₁	w₄₁₂	w₄₁₃	w₄₁₄	w₄₁₅	w₄₁₆	w₄₁₇	w₄₁₈	w₄₁₉	w₄₂₀	w₄₂₁	w₄₂₂	w₄₂₃	w₄₂₄	w₄₂₅	w₄₂₆	w₄₂₇	w₄₂₈	w₄₂₉	w₄₃₀	w₄₃₁	w₄₃₂	w₄₃₃	w₄₃₄	w₄₃₅	w₄₃₆	w₄₃₇	w₄₃₈	w₄₃₉	w₄₄₀	w₄₄₁	w₄₄₂	w₄₄₃	w₄₄₄	w₄₄₅	w₄₄₆	w₄₄₇	w₄₄₈	w₄₄₉	w₄₅₀	w₄₅₁	w₄₅₂	w₄₅₃	w₄₅₄	w₄₅₅	w₄₅₆	w₄₅₇	w₄₅₈	w₄₅₉	w₄₆₀	w₄₆₁	w₄₆₂	w₄₆₃	w₄₆₄	w₄₆₅	w₄₆₆	w₄₆₇	w₄₆₈	w₄₆₉	w₄₇₀	w₄₇₁	w₄₇₂	w₄₇₃	w₄₇₄	w₄₇₅	w₄₇₆	w₄₇₇	w₄₇₈	w₄₇₉	w₄₈₀	w₄₈₁	w₄₈₂	w₄₈₃	w₄₈₄	w₄₈₅	w₄₈₆	w₄₈₇	w₄₈₈	w₄₈₉	w₄₉₀	w₄₉₁	w₄₉₂	w₄₉₃	w₄₉₄	w₄₉₅	w₄₉₆	w₄₉₇	w₄₉₈	w₄₉₉	w₅₀₀	w₅₀₁	w₅₀₂	w₅₀₃	w₅₀₄	w₅₀₅	w₅₀₆	w₅₀₇	w₅₀₈	w₅₀₉	w₅₁₀	w₅₁₁	w₅₁₂	w₅₁₃	w₅₁₄	w₅₁₅	w₅₁₆	w₅₁₇	w₅₁₈	w₅₁₉	w₅₂₀	w₅₂₁	w₅₂₂	w₅₂₃	w₅₂₄	w₅₂₅	w₅₂₆	w₅₂₇	w₅₂₈	w₅₂₉	w₅₃₀	w₅₃₁	w₅₃₂	w₅₃₃	w₅₃₄	w₅₃₅	w₅₃₆	w₅₃₇	w₅₃₈	w₅₃₉	w₅₄₀	w₅₄₁	w₅₄₂	w₅₄₃	w₅₄₄	w₅₄₅	w₅₄₆	w₅₄₇	w₅₄₈	w₅₄₉	w₅₅₀	w₅₅₁	w₅₅₂	w₅₅₃	w₅₅₄	w₅₅₅	w₅₅₆	w₅₅₇	w₅₅₈	w₅₅₉	w₅₆₀	w₅₆₁	w₅₆₂	w₅₆₃	w₅₆₄	w₅₆₅	w₅₆₆	w₅₆₇	w₅₆₈	w₅₆₉	w₅₇₀	w₅₇₁	w₅₇₂	w₅₇₃	w₅₇₄	w₅₇₅	w₅₇₆	w₅₇₇	w₅₇₈	w₅₇₉	w₅₈₀	w₅₈₁	w₅₈₂	w₅₈₃	w₅₈₄	w₅₈₅	w₅₈₆	w₅₈₇	w₅₈₈	w₅₈₉	w₅₉₀	w₅₉₁	w₅₉₂	w₅₉₃	w₅₉₄	w₅₉₅	w₅₉₆	w₅₉₇	w₅₉₈	w₅₉₉	w₆₀₀	w₆₀₁	w₆₀₂	w₆₀₃	w₆₀₄	w₆₀₅	w₆₀₆	w₆₀₇	w₆₀₈	w₆₀₉	w₆₁₀	w₆₁₁	w₆₁₂	w₆₁₃	w₆₁₄	w₆₁₅	w₆₁₆	w₆₁₇	w₆₁₈	w₆₁₉	w₆₂₀	w₆₂₁	w₆₂₂	w₆₂₃	w₆₂₄	w₆₂₅	w₆₂₆	w₆₂₇	w₆₂₈	w₆₂₉	w₆₃₀	w₆₃₁	w₆₃₂	w₆₃₃	w₆₃₄	w₆₃₅	w₆₃₆	w₆₃₇	w₆₃₈	w₆₃₉	w₆₄₀	w₆₄₁	w₆₄₂	w₆₄₃	w₆₄₄	w₆₄₅	w₆₄₆	w₆₄₇	w₆₄₈	w₆₄₉	w₆₅₀	w₆₅₁	w₆₅₂	w₆₅₃	w₆₅₄	w₆₅₅	w₆₅₆	w₆₅₇	w₆₅₈	w₆₅₉	w₆₆₀	w₆₆₁	w₆₆₂	w₆₆₃	w₆₆₄	w₆₆₅	w₆₆₆	w₆₆₇	w₆₆₈	w₆₆₉	w₆₇₀	w₆₇₁	w₆₇₂	w₆₇₃	w₆₇₄	w₆₇₅	w₆₇₆	w₆₇₇	w₆₇₈	w₆₇₉	w₆₈₀	w₆₈₁	w₆₈₂	w₆₈₃	w₆₈₄	w₆₈₅	w₆₈₆	w₆₈₇	w₆₈₈	w₆₈₉	w₆₉₀	w₆₉₁	w₆₉₂	w₆₉₃	w₆₉₄	w₆₉₅	w₆₉₆	w₆₉₇	w₆₉₈	w₆₉₉	w₇₀₀	w₇₀₁	w₇₀₂	w₇₀₃	w₇₀₄	w₇₀₅	w₇₀₆	w₇₀₇	w₇₀₈	w₇₀₉	w₇₁₀	w₇₁₁	w₇₁₂	w₇₁₃	w₇₁₄	w₇₁₅	w₇₁₆	w₇₁₇	w₇₁₈	w₇₁₉	w₇₂₀	w₇₂₁	w₇₂₂	w₇₂₃	w₇₂₄	w₇₂₅	w₇₂₆	w₇₂₇	w₇₂₈	w₇₂₉	w₇₃₀	w₇₃₁	w₇₃₂	w₇₃₃	w₇₃₄	w₇₃₅	w₇₃₆	w₇₃₇	w₇₃₈	w₇₃₉	w₇₄₀	w₇₄₁	w₇₄₂	w₇₄₃	w₇₄₄	w₇₄₅	w₇₄₆	w₇₄₇	w₇₄₈	w₇₄₉	w₇₅₀	w₇₅₁	w₇₅₂	w₇₅₃	w₇₅₄	w₇₅₅	w₇₅₆	w₇₅₇	w₇₅₈	w₇₅₉	w₇₆₀	w₇₆₁	w₇₆₂	w₇₆₃	w₇₆₄	w₇₆₅	w₇₆₆	w₇₆₇	w₇₆₈	w₇₆₉	w₇₇₀	w₇₇₁	w₇₇₂	w₇₇₃	w₇₇₄	w₇₇₅	w₇₇₆	w₇₇₇	w₇₇₈	w₇₇₉	w₇₈₀	w₇₈₁	w₇₈₂	w₇₈₃	w₇₈₄	w₇₈₅	w₇₈₆	w₇₈₇	w₇₈₈	w₇₈₉	w₇₉₀	w₇₉₁	w₇₉₂	w₇₉₃	w₇₉₄	w₇₉₅	w₇₉₆	w₇₉₇	w₇₉₈	w₇₉₉	w₈₀₀	w₈₀₁	w₈₀₂	w₈₀₃	w₈₀₄	w₈₀₅	w₈₀₆	w₈₀₇	w₈₀₈	w₈₀₉	w₈₁₀	w₈₁₁	w₈₁₂	w₈₁₃	w₈₁₄	w₈₁₅	w₈₁₆	w₈₁₇	w₈₁₈	w₈₁₉	w₈₂₀	w₈₂₁	w₈₂₂	w₈₂₃	w₈₂₄	w₈₂₅	w₈₂₆	w₈₂₇	w₈₂₈	w₈₂₉	w₈₃₀	w₈₃₁	w₈₃₂	w₈₃₃	w₈₃₄	w₈₃₅	w₈₃₆	w₈₃₇	w₈₃₈	w₈₃₉	w₈₄₀	w₈₄₁	w₈₄₂	w₈₄₃	w₈₄₄	w₈₄₅	w₈₄₆	w₈₄₇	w₈₄₈	w₈₄₉	w₈₅₀	w₈₅₁	w₈₅₂	w₈₅₃	w₈₅₄	w<

⑥ If $w_j \in L_d$.

M_j recognises L_d only if it accepts strings of L_d only

$\Rightarrow M_j$ don't accept w_j as $w_j \notin L_d$

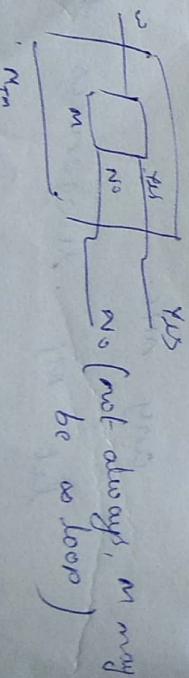
$\Rightarrow w_j \notin L_d$ (by defn of L_d)

\Rightarrow contradiction

L_d is not recognisable by any Turing m/c.

$\Rightarrow L_m$ turing recognisable?

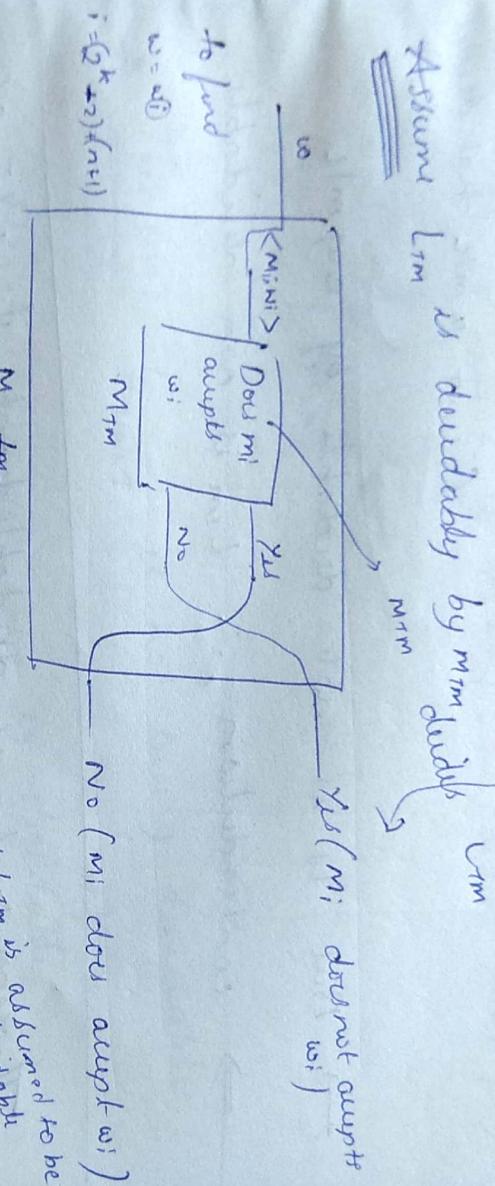
A. Yes. $L_m = \{ \langle m, w \rangle \mid m \text{ accepts } w \}$



$L_d = \{w_i : m_i \text{ does not accept } w_i\}$

Contry $c_{i,i} > 0$.

Algorithm (r/m) for L_d



Algorithm is: (Assume M_m divides w_i & $w_i \in L_d$)

- Find i such that $w = w_i$
- Circ $\langle m_i, w_i \rangle$ as $1/p$ to M_m

If M_m says yes, ans no & vice versa.

$\Rightarrow M_d$ decides L_d but we proved that

No r/m decides L_d

\therefore contradiction $\Rightarrow M_m$ is not deducible in L_m

what we have done is:

→ Assume L_m is decidable. Then

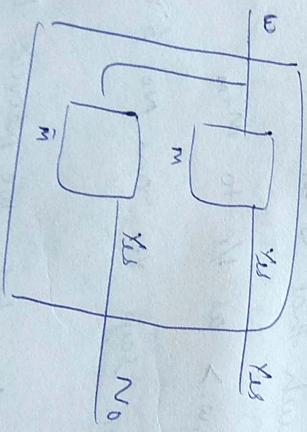
→ Show that L_d is also decidable

(using a m/c that decides L_m)

→ But L_d is not decidable by any m/c

⇒ contradiction. ∴ L_m is non decidable

Theorem: L is decidable if and only if $L \neq L_m$ are recognizable



hence is obvious

* → hence L_m is undecidable. $\overline{L_m}$ is not recognizable

Theo
= f

Th

HALTING Problem

Given a turing m/c $m \notin$ string w

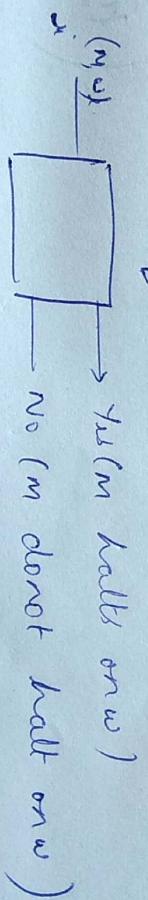
Does m halt on w ? i.e., goes to accept/reject

$$L_{\text{HALT}} = \{ \langle m, w \rangle : m \text{ halts } w \}$$

* Theorem: L_{HALT} is not turing decidable

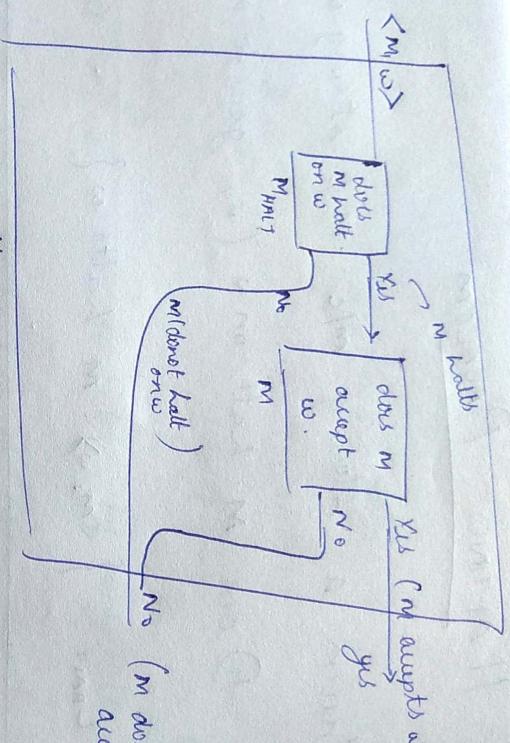
Proof: Assume L_{HALT} is decidable

i.e., \exists a TM M_{HALT} that decides L_{HALT}



M_{HALT}

Then we can make a TM that decides L_{TM} . (its undecidable as we couldn't say if m stops or not)



But L_{TM} is undecidable (so no tm decides L_{TM})

$\Rightarrow M_{HALT}$ is undecidable

