* Turing machine

$M = (Q, \Sigma, T, \delta, q_0, B, F)$.

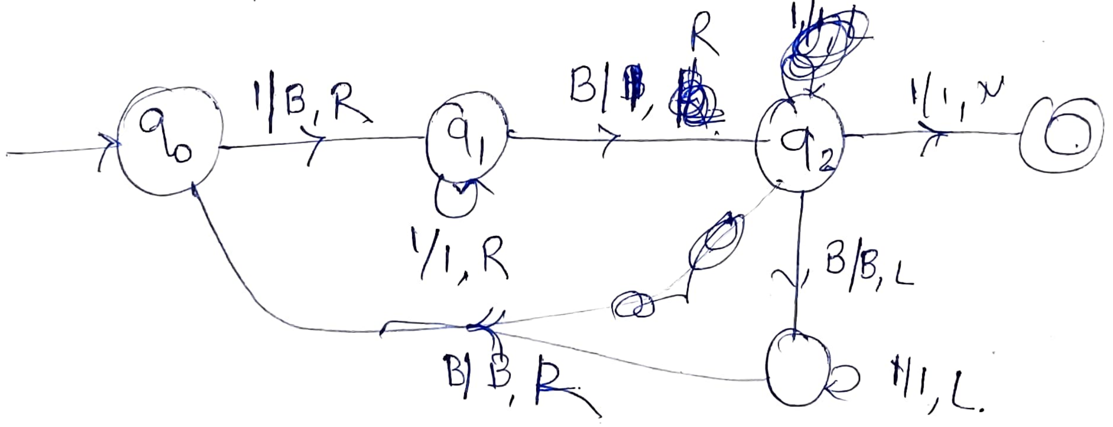
1. Q is finite set of states.

2. δ
$Q \times T \rightarrow Q \times T \times D (L, R, N)$

① Design TM, ~~fini~~ initially contains 2 finite blocks of 1's seperated by blanks. The m/c should delete the block of blanks betn 2 blocks of 1's.

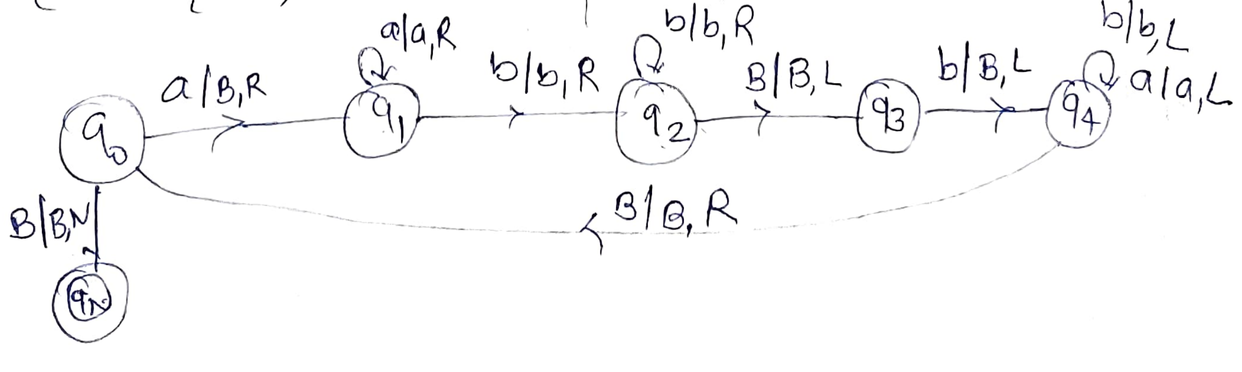
→

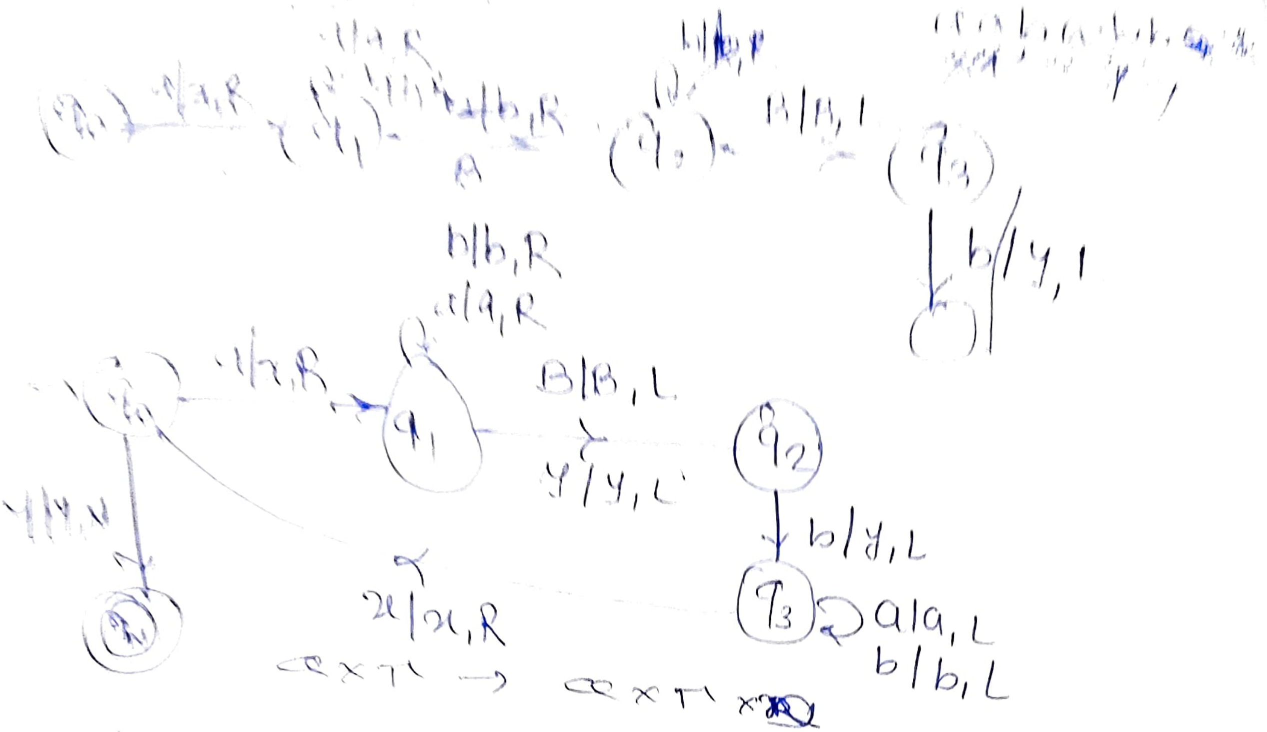
B	1	1	1	B	B	B	1	1	1	1	B



eg.2 $L = \{a^n b^n \mid n \geq 1\}$.

$L = \{ab, aabb, aaabbb, \text{------} \}$

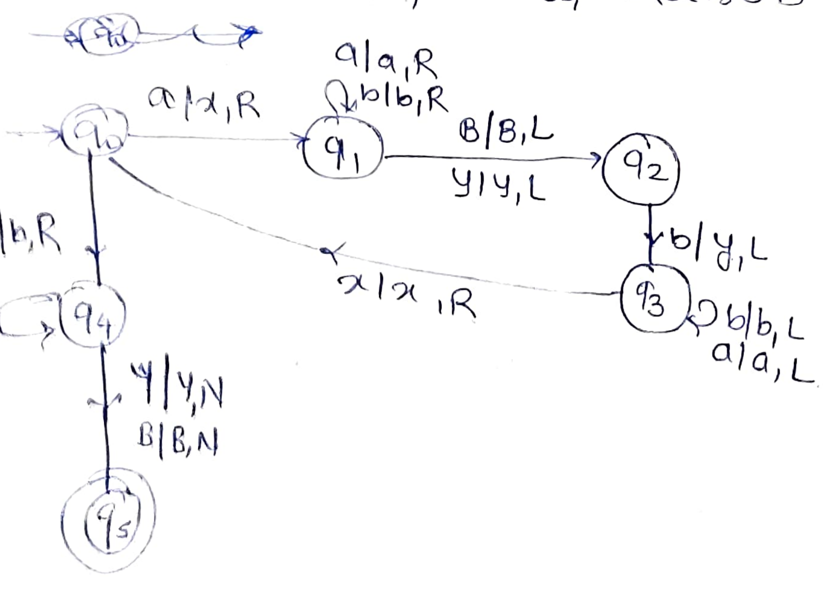




	a	b	x	y	B
→ q0	(q1, x,R)	–	–	(qN, y,N)	–
q1					
q2					
q3					
* qN					

eg.3 $L = \{ a^i b^j \mid i < j \}$

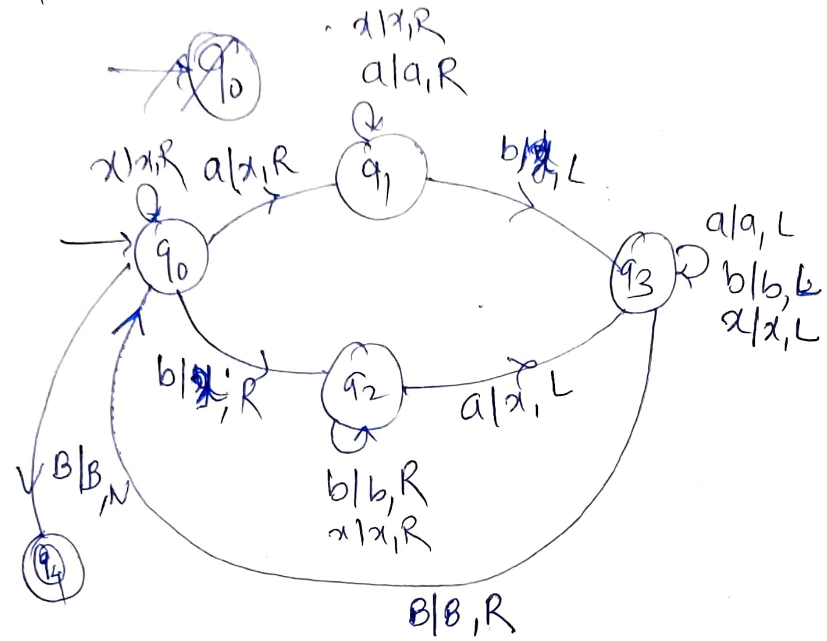
$L = \{b, bb, bbb, abb, abbb, aabbb$



4. Design TM that contains equal no. of a's and b's.

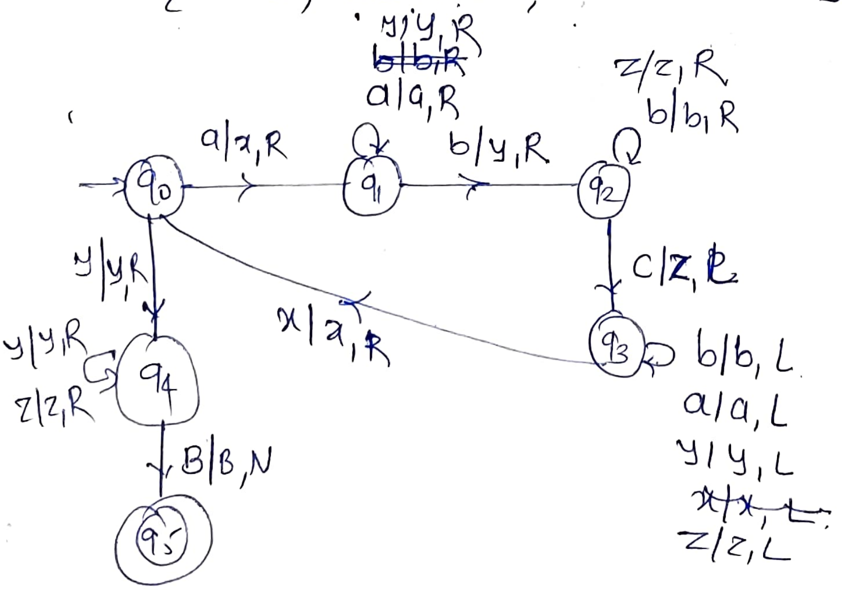
$L = \{ ab, ba, aabb, abab, baba, aabbaabb, \ldots \}$

→



eg. 5 $L = \{ a^n b^n c^n \quad n \geq 1 \}$

$L = \{ abc, aabbcc, \ldots \}$

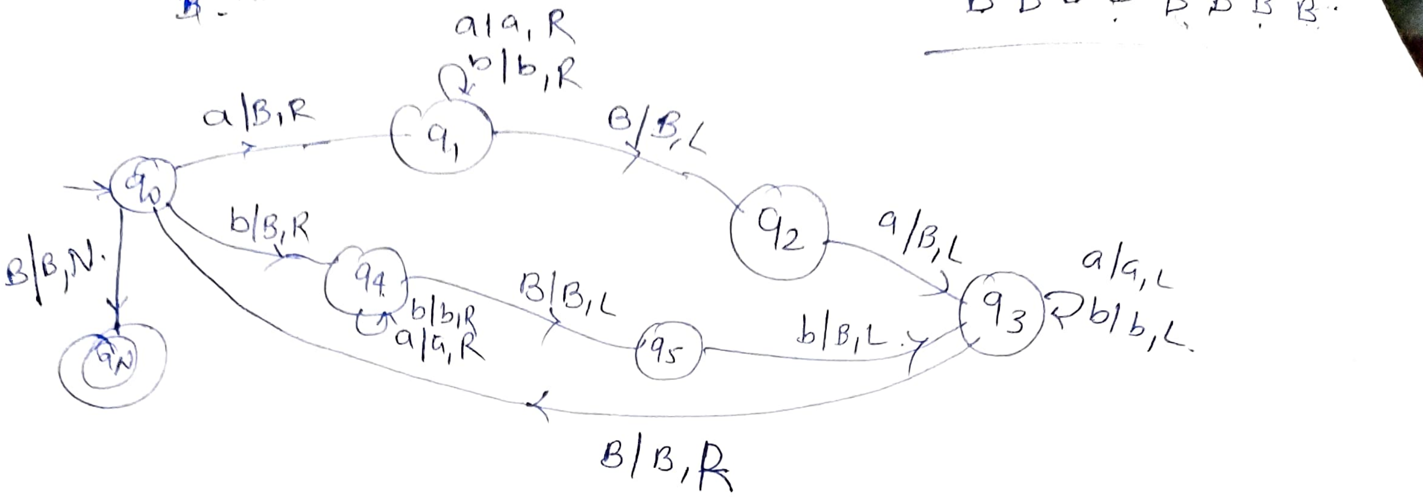


eg 6

Design TM over ww^R

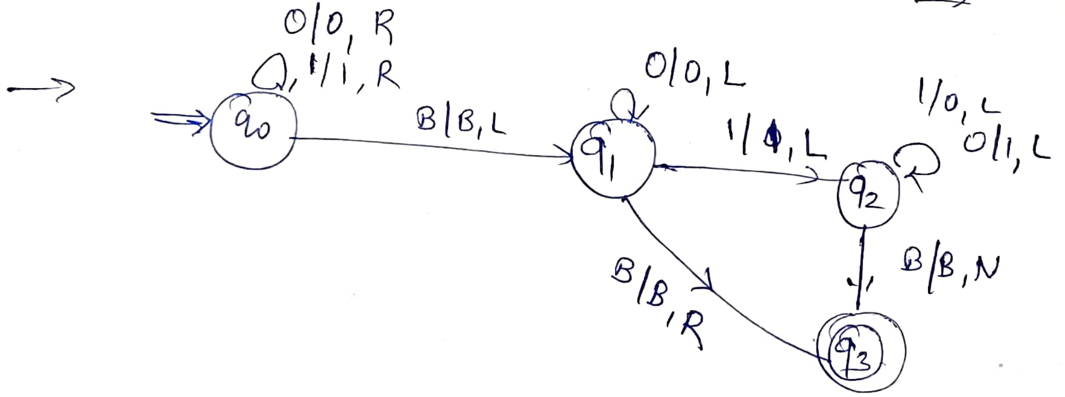
abba
B.

aaba abaa
B B B B B B B B.



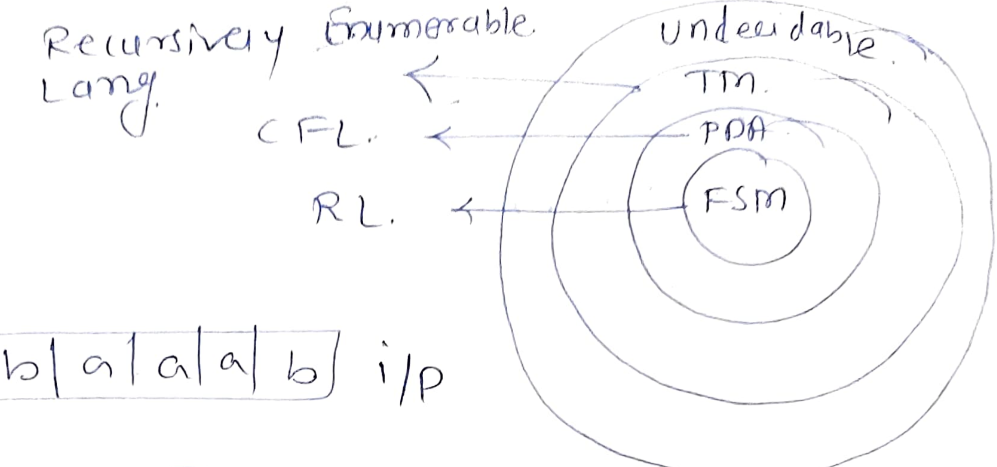
eg.7 Design TM for 2's comp. of binary

0 1 0 1 1 → 1 0 1 0 0 ⇒ 1 0 1 0 1

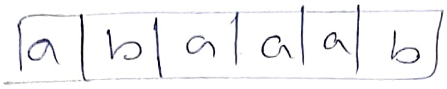


~~ch. for TM~~ construction.

TM

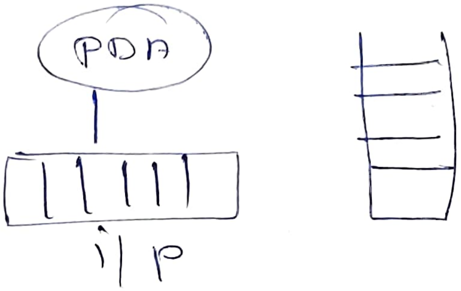


FSM →



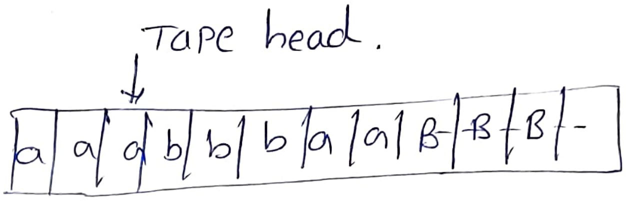
i/p

PDA →



TM.

a Tape



Σ = {.

Opⁿ on the tabe.

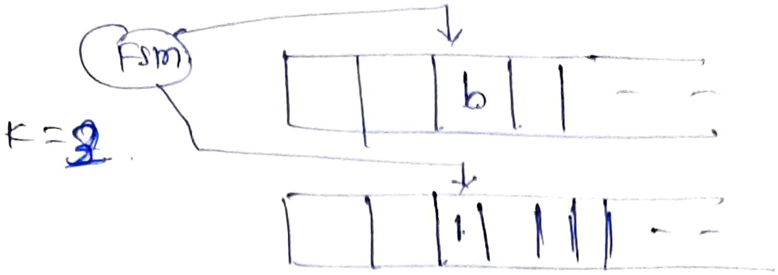
① Read ② update ③ move L, R.

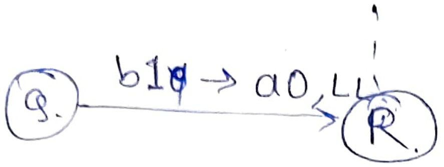
* Techniques of TM.

↳ Diff. example.

Variants of TM.

(1) Multitape TM





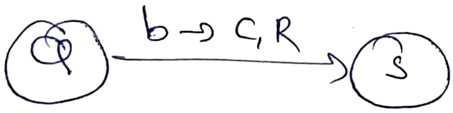
(2) Non-Deterministic TM.

more powerful.

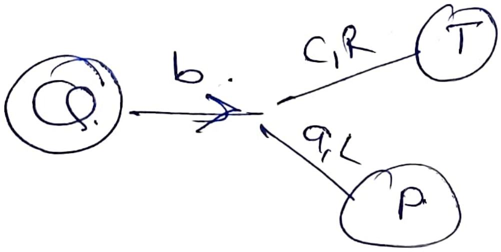
$\delta \rightarrow Q \times \Sigma \rightarrow Q \times \Gamma \times (R,L)$ - D.TM.

$\delta \rightarrow Q \times \Sigma \rightarrow P\{Q \times \Gamma \times (R|L)\}$

D.TM.



NDTM



Lang.

A lang 'L' is said to be recursive if there exists a TM which will accept all the string in 'L' & reject all the strings not in 'L'

T.M. will halt every time & give an answer (accepted or rejected for each & every string i/p.

- not go in loop., Always halt

R.E.L

A lang 'L' is said to be R.E. if there exist TM which will accept (& halt) for all i/p string which are in 'L'.

But may or may not halt for all i/p string which are not in 'L'.

- not guarantee of halt.

Decidable Lang. - "A language 'L' is decidable if it is recursive lang All decidable languages are recursive & vice-versa."

i.e. TM always halt. by either accepting or rejected

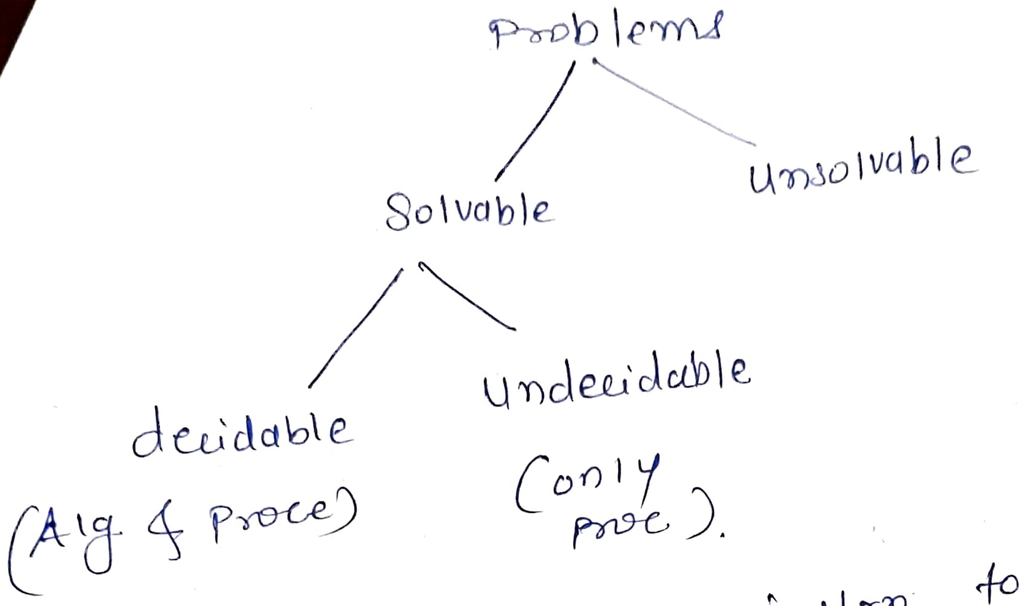
Partially decidable Lang: "A lang 'L' is partially decidable if 'L' is a recursively enumerabe Lang"

Undecidable Lang :-

- if it is not decidable.
- it may sometimes be partially decidable but not decidable.
- If a lang is not even partially decidable, then there exists no T.M. for that lang.

Recursive Lang	- TM will always to Halt
Recu. Enumerable Lang	- TM will halt sometimes & may not halt sometimes
Decidable Lang	- Recursive Lang
Partially Decidable Lang.	- Recursively Enumerable Lang
Undecidable	- No TM for that lang.

ecidable and undecidable problems



<u>Proce.</u> Step by step instrn to solve a prob.

<u>Algo</u> : proce + Approximate time in which a prob. can be solved

<u>eg.</u> Bubble sort Algo. → Time comp. $O(n^2)$ worstca

Insertion sort " - " $\Theta(n^2)$

Heapsort " - " $O(n\log n)$

Linear search. " $O(n)$

Binary search. = " $O(\log n)$.

[If time is approximately predicatable then it is Algo.]

- proce.	Algo.
① can halt or need not halt	①. Always halt & give output.

eg. First Rank in CAT? <u>solvable</u> ① go & study ② write exam ③ check the result. ④ if rank=1 then stop otherwise go to step I no time constraint so Procedure

Decidable → Sol^n is definite (either Y or N)

eg 1 → Does sun rises in the east?

Yes

.2 → Does earth moves around the sun?

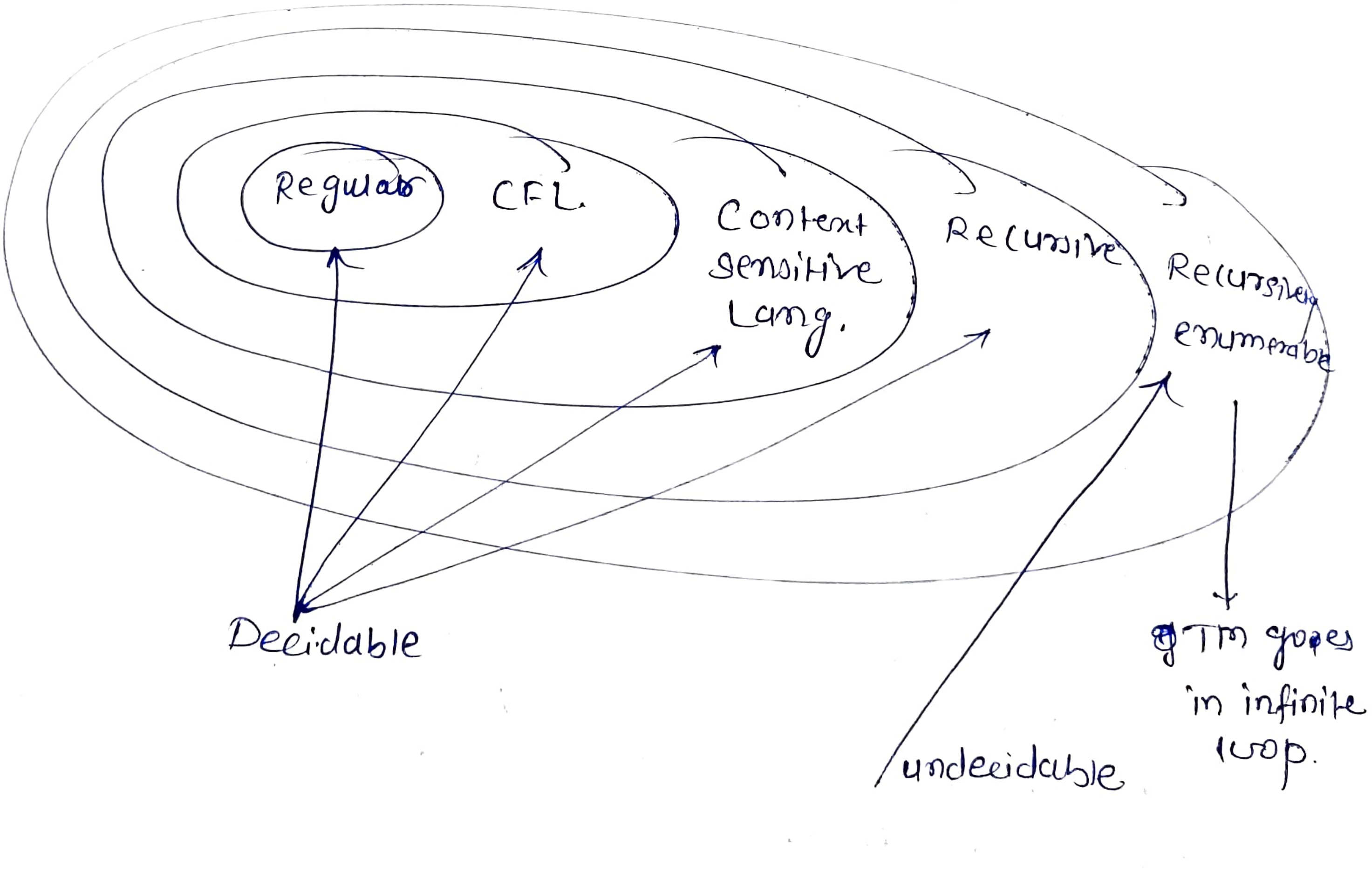
Yes

Undecidable Sol^n is indefinite (sometimes Y & sometimes N)

eg will tomorrow would be a rainy day?

- undecidable.

: Decidable and undecidable languages



hurch- Turing Thesis

What does computable mean.

- Church- Lambda calculus. - calculated with L.C called computable
- Alan Turning - T.M.
  ↳ what ever computable by these machines called <u>computable</u>.

- <u>Variations</u>

① one Tape or many

② Infinite on both ends

③ Multiple head

④ Universal Turing M.

⑤ Non- Deterministic

* <u>Universal T.M.</u>

- TM for all the TM we have
- $A_{TM} = \{ \langle M, w \rangle \mid M$ is TM & M accepts w $\}$ is Turing Recognizable.

* Recursive Enumerable Lang.

-> L is RE if there is TM.

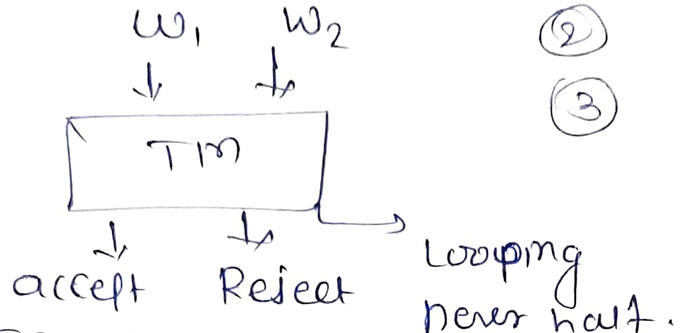
Lang accepted by TM is called RE

3 states

① Halt & accept

② Halt & Reject

③ Never halt.



- ~~In~~ In some case. Lang will ~~b~~ never halt.

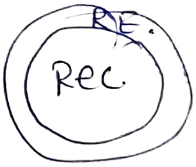
* Recursive Lang:

~~TKM.~~ ~~having~~

- L is Recursive if there is Halting / total TM

2 states - Halt & Accept
Halt & Reject.

- it will give confirm answer.



* Decidability & undecidability

1

alting Problem

- eg of Undecidability
- What is Halting Problem

→ Given a program, will it halt? Halting means prog will accept a halt or reject a halt never goes into a loop

~~Q the~~

Algo Given a TM, will it halt when run on some particular given i/p string? yes or not.

÷ can we design generalized algorithm into which we pass prog ~~of whether that progg~~ then that algo tell us whether prog will halt or not.

- eg. Given some program written in some language (Java|C++ etc) will it ever get into an infinite loop or will it always terminate?

→ can you find out that that prog always goes in an Infinite loop or always terminate?

- we can not design m/c that will not allow to code go in infinite loop.

- Halting problem is undecidable.

Ans

① In general we can't always know.

② The best we can do is run the Pr and see whether it halts.

③ For many programs we can see that it will always halt or sometimes loop.

its not conclusion.

But For Prg in general the ques^{n} is undecidable.

Church Turning Theom.

↳ Anything that is computable has an algo. for that we can design T.M.

i.e if there is TM then that has an algo.

Halting prob. can't have TM so can not be computtable i.e.

inear Bounded Automata

LBA = TM + Input size tape

- TM is infinite tape.
- LBA ~~ω~~ is TM with limited input size tape.

$	a	b	a	b	$

- FA < PDA < LBA < TM ⟶ RE Lang. Recursive

  RL. CFL CSL accepted

- D. N D N D. NO. D & N. D N

- LBA accepting RL, CFL & CSL.

- Lang accepted ex.

  1. $L = \{a^n b^n c^n : n \geq 1\}$ - Not PDA.
  2. $L = \{a^n : n \text{ is prime}\}$
  3. $L = \{a^n : n \text{ is non prime}\}$
  4. $L = \{a^n b : n \geq 0\}$
  5. $L = \{ww w^R : w \in (a,b)^*\}$

Symbols , alphabet, strings over an alphabet,
Σ^* for alphabet Σ, formal lang over Σ.

0, 1 , $\{0, 1\}$ $\{0, 1, 01, 10\}$ $\{0, 1, 01, 10, 11, 111 \ldots\}$

* Automata → Abstract way to represent algo.

* Introduction fundamental course in cs & I)

Prob → I have a comp. prog. & an i/p x, will the prog. terminate when i/p x is fed?

Prob$_2$ → I have a graph G. Can G be colored using 3 colors (RGB)