24th Nov

Can computers do "EVERYTHING"?

A. TURING.

Today:

1. An intuitive explanation why computers possibly can not solve all pbms
2. An example with pf that computers indeed cannot solve everything.

$x^n + y^n = z^n$, $n \geq 3$ is not terminated

Remark.

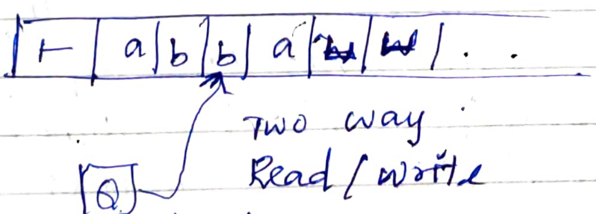
If one can write a program which can scan any program input pair (P, I) & decide if the program P with input I will print "Hello, world" then we are going a proof of Fermat's Last Thm

HIGHLY UNLIKELY.

Russell's Paradox:

Naive Set Theory.

Turing Machine



Informal Description:

A TM has a finitely many states Q, a semi finite tape that is delimited on the left by endmarker ⊢ & infinite on right, and a head that can move left & right over the tape reading & writing symbols

The input string is of finite length & is initially written on the tape in continuous tape cells sung up against the left endmaker. The infinitely many cells to the right of the input contains a special blank symbol $\sqcup$.

The machine starts at its start state s with its head scanning the left endmaker. In each step it reads the symbol on the tape under in head depending on that symbol & the current state it writes a new symbol on that cell & move to left or right.

It accepts its input if it reaches to accept state t, reject if it reaches the reject state r or it runs infinitely causing a "loop".

Formal Defn:

A deterministic one way TM is a 9-tuple

$$M = (Q, \Sigma, \Gamma, \vdash, \sqcup, \delta, s, t, r)$$

Where,

1. Q is the finite set of states
2. Σ finite set of input alphabets.
3. Γ finite set of tape alphabets with $\Sigma \subseteq \Gamma$
4. $\sqcup \in \Gamma \setminus \Sigma$ blank symbol.
5. $\vdash \in \Gamma \setminus \Sigma$ left marker.
6. $\delta : Q \times \Gamma \longrightarrow Q \times \Gamma \times \{L, R\}$
   The transition fn where L & R denotes left & right
7. $s \in Q$ start state
8. $t \in Q$ accept state.
9. $r \in Q$ reject state $r \neq t$

&

A. $\delta(p, a) = (q, b, d)$ means when in state p & scanning symbol a write b on that tape cell & move to direction d & enter that q.

B. Endmarker is never over written
i) $\forall p \in Q, \exists q \in Q$ s.t $\delta(p, \vdash) = (q, \vdash, R)$.

· Once TM enters accept state it never leaves; same for reject state
ii) $\forall b \in \Gamma, \exists c, c' \in \Gamma$ & $d, d' \in \{L, R\}$ s.t
$\delta(t, b) = (t, c, d)$, $\delta(r, b) = (r, c', d')$.

Exc
1. Try to construct an example of a Turing Machines.
2. S.T "does a program Q, given input y, ever call function foo" is undecidable.
3. Read defn of undecidability.

1. Give an example of a TM that accepts $\{a^n b^n c^n \mid n \geq 0\}$
2. Give an example of a TM which accepts all of Σ^*.
3. Give an example of a TM which rejects all strings of Σ^*.

MARIA_MARIA

Defn/Notn: Γ^{ω} = All input strings.

Let z be any input string.

$S_b^n(z)$ = string obtained by replacing n^{th} term z_n of z by b.

4th position changes to b

$S_b^4(\vdash baaacabca\ldots) = \vdash baabcabca.$

Let M be a TM

We define $\xrightarrow[M]{1}$.

M is at,

(q, z, n) means it is reading strings z, is at state q & currantly at n^{th} position.

This is called a configuration of M.

$$(p, z, n) \xrightarrow[M]{1} \begin{cases} (q, S_b^n(z), n-1) & \text{if } \delta(p, z_n) = (q, b, L) \\ (q, S_b^n(z), n+1) & \text{if } \delta(p, z_n) = (q, b, R) \end{cases}$$

Turing Machine:

$M = (Q, \Sigma, \Gamma, \vdash, \sqcup, \delta, s, t, r)$

States, Input alphabets, Tape Alphabets, left end Marker, Blank symbol, Transition fcn, start state, accept state, reject state.

where (p, z, n) implies that the machine is at state p & reading nth cell of string z.

Reflexive closure:

$\xrightarrow[M]{*}$ of $\xrightarrow[M]{1}$.

- $\alpha \xrightarrow[M]{0} \alpha$
- $\alpha \xrightarrow[M]{n+1} \beta$ if $\alpha \xrightarrow[M]{n} \gamma$ & $\gamma \xrightarrow[M]{1} \beta$ for some γ
- $\alpha \xrightarrow[M]{*} \beta$ if $\alpha \xrightarrow[M]{n} \beta$ for some $n \geq 0$.

<u>M accepts.</u>
input $x \in \Sigma^*$ if $(s, \vdash x\sqcup^{\omega}, 0) \xrightarrow[M]{*} (t, y, n)$

for some y & n; it rejects x if
$(s, \vdash x\sqcup^{\omega}, 0) \xrightarrow[M]{*} (r, y', n')$ for some y', n'

- M is said to <u>halt</u> on input x if it either accepts or rejects x, Otherwise it is said to <u>loop</u>.

- A turing Machine M is called <u>total</u> if it halts on all inputs.

- $L(M) = \{x \in \Sigma^* / x$ accept by $M\}$.

- A set of strings is called <u>recursively enumerable</u> (r.e) if it is $L(M)$ for some turing machine M.

- A set of strings is called <u>co-r.e</u> if its complement is r.e.

- A set of strings is called recursive if it is $L(M)$ for some total turing machine.

- A property P of strings is called decidable if $\{x / P(x)\}$ is recursive.

ii) for any set A, A is r.e if "$x \in A$" is semi-decidable.

~~Pbm~~:
S.T /

Pbm:
S.T $A = \{ww \mid w \in \{a,b\}^*\}$ is recursive.

Pbm
Find a total TM that accepts $\{w \mid |w|$ is prime

Remarks:
1. A recursive then so is $A^c = \Sigma^* \setminus A$.
$A = L(M)$.
Let M' be identical to M except

Illu $t_M = r_{M'}$, $r_M = t_{M'}$

2. Every recursive state is recursively enumera
3. If A & A^c both r.e then A is recursive

$i : \mathbb{Z}^k \xrightarrow{\text{surjective}} A$ abelian gp $A = \langle a_1, \dots, a_k \rangle$
$(A, +)$

2

$e_i \mapsto a_i$
extend using homomorphism

$A \cong \dfrac{\mathbb{Z}^k}{\ker i}$

$\mathbb{Z}^k$ is called free abelian gp of rk k = Universal object among abelian gp generated by k elts.

All abelian gps gen by k elts are quotients of $\mathbb{Z}^k$.

If first part of $x \in \{a,b\}^*$ is $a^n b a^k b a^m$ then it says x may represent a TM with states $|\Sigma| = k$, $|\Gamma| = m$.

$$\underbrace{a^n b a^k b a^m \ldots a}_{M_x} \# y$$

TM M_x with input y

Create a TM which takes input x from $\{a,b,\#\}^*$. Then

① Decides if x is $M \# y$ for M

__Theorem 1__

$\{M \# x \mid M$ a TM & x is a string with $x \in L(M)\}$.

is NOT RECURSIVE.

i.e) the problem of checking whether a given turing machine accepts a given string is not decidable.

__Theorem 1'__

$\{M \# x \mid M$ a TM & x a string s.t M halts with x input$\}$ is NOT recursive.

__Proof:__

Let U be a TM that

① decides if the given input is a TM & valid input or not. If no rejects.

If yes goes to step ②

② Applies x on M & does what M does.

i.e) accepts if M accepts x, rejects if M rejects x, Diagonalisation Technique

loops if M loops at x. ii) simulates action of M on x.

[U is called a universal TM].

$L(U) = \{M \# x \mid M \text{ TM } \& \ M \text{ accepts } x\}$

$\forall x \in \{0,1\}^*$ let M_x be the TM over $\{0,1\}$ whose encoding is x

If x not legal encoding then M_x is the trivial TM with 1 state that halts immediately

we get :

$M_\epsilon, M_0, M_1, M_{00}, M_{01}, M_{10}, M_{11}, M_{100}, \ldots$ // all TMs over $\{0,1\}$

	ϵ	0	1	00	01	10	11	100 ...
M_ϵ	H	L	L	H	H	L	-	
M_0	L	H	- - -					
M_1								
M_{00}								
M_{01}								
:								

let $\exists$ a total TM K accepting
$HP = \{M \# x \mid M \text{ TM } \& \ M \text{ halts at } x\}$

ie) (i) K halts & accepts if M halts on x.
(ii) K rejects otherwise.

x,) Consider a machine N that on input $x \in \{0,1\}^*$
(i) Constructs M_x from x & writes $M_x \# x$ on in
(ii) runs K on $M_x \# x$ accepting if K rejects & going to a trivial loop if K accepts !!

Then for any $x \in \{0,1\}^*$, N halts on x.
$\Leftrightarrow$ K rejects $M_x \# x$.
$\Leftrightarrow$ M_x loops on x !!
$\Leftrightarrow$ K # M_x for any x !! &

Exc : Prove Thm 1 (Accept version)

2 A r.e, B r.e . Is $A \cap B$ r.e?

Thm : Regular $\Longrightarrow$ Recursive

Converse not true

Ex $\{a^n b^n c^n / n \geq 0\}$.

$(Q, \Sigma, \Gamma, s, t, r, \delta, \sqcup, \vdash)$

Q Finite $\mathbb{N}$

Σ Finite $\mathbb{N}$

Γ Finite $\mathbb{N}$

$\delta : Q \times \Gamma \longrightarrow Q \times \Gamma \times \{L, R\}$

$\boxed{|\mathbb{N}^3 \times A| = \mathbb{N}}$ $\longrightarrow$ Total no. of TM $\longrightarrow$ 1.

Total no of possible transition

$\{0,1\}^0$

$\cup$

$\{0,1\}^1$

$\vdots$

$\{0,1\}^k$ $\longrightarrow$ 2

{Set 0

$\exists$ 1-1 map. onto map from $X \longrightarrow \mathbb{N}$ $\longrightarrow$ 1.a

One possible enumeration

$0^k 1 0^l 1 0^m 1 0 1 0 1 0^n 1 0^p$
$1 0 1 0^2 1 0 1 0^q 1 0^r 1 0^2$

$|Q| = k$ $|\Sigma| = l$ $|\Gamma \setminus \Sigma| = m$

$\delta(1^{st}$ state, 1^{st} alphabet$)$
$=$ n^{th} state, p^{th} alphabet, left

$\delta(2^{nd}, 1^{st}) = (q^{th}, r^{th}$, right$)$

Universal TM :

Universal TM

Step 1

① + ② + ③ + ④ says that $\exists$ 1-1 map from $T \to X$

Step 2

We construct a TM which takes TMs on alphabet {0,1} [or ^lly any other alphabet) & strings of {0,1} as input then 'simulates the TM on the string'.

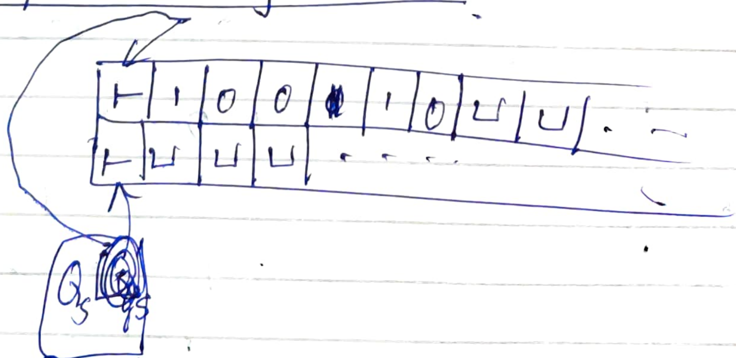
Formally we construct U s.t alphabet of U is {0,1,#}

& (i) U checks whether an input is of the form M # x where M & x are string of {0,1} with M representing a TM under some fixed representation

IF NOT U REJECTS THE INPUT.

(ii) If yes then U acts as M on input x.

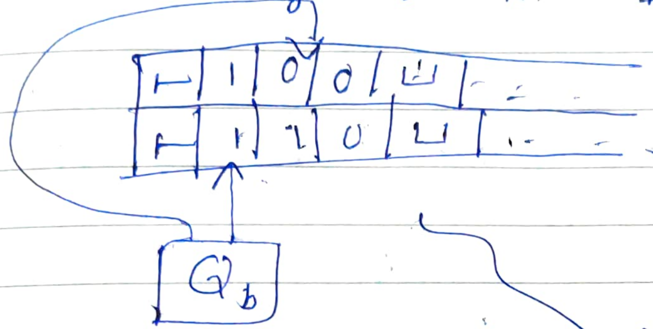
2-Tape Turing Machine:



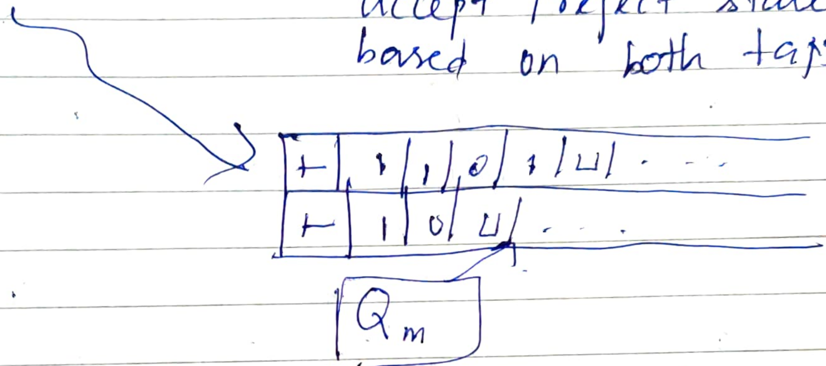
1. Input written on top tap & bottom one is empty
2. Both has left end marker
3. Reads from both.

Time complexity
Reducibility
Incompleteness

tapes & writes on both
states changes as in 1 tape machine.



If halts if it halts on both tapes
i.e) it reaches to an accept / reject state based on both taps



has written 1 on top tape & moved left & has kept the bottom tape unchanged & moved right.

similarly one can defin n-tape Machine.

Remark: 2-Tape machine can be visualized as a 1-tape machine by changing Σ to $\Sigma_{new} = \Sigma \times \Sigma$.

Time complexity:

Ex $A = \{0^k 1^k \mid k \geq 0\}$

'A' not regular

'A' recursive

Q:- How much time does a single tape TM need to decide A?

~~The is not strongly~~ ~~measurably~~:

M_1 - On input string w
1. Scan & reject if a 0 is found to the right of 1
2. Repeat if both 0's & 1's remain on tape
3. Scan accross tape "crossing off" a single 0 & a right 1.
4. If 0's still remain after all 1's are crossed or 1's still remain after 0's are crossed, reject. Otherwise accept.

Defn: If M is a deterministic TM that halts on all inputs, the running time or time complexity of M is the fn $f: N \to N$, where $f(n)$ is the maximum no. of steps that M uses on any input of length n. If $f(n)$ is the running time of M, we say that M runs in time $f(n)$ & that M is an $f(n)$ time TM.

Defn:

$f, g : N \to R_{>0}$, $f(n) = O(g(n))$ if $\exists$ c & n_0 s.t $\forall n \geq n_0$
$$f(n) \leq c\, g(n)$$
When $f(n) = O(g(n))$ we say that $g(n)$ is an asymptotic upper bound of f

Ex $5n^3 + 2n^2 - 6 = O(n^3)$.

Also we say
$$f(n) = o(g(n))$$
if $\lim_{n \to \infty} \frac{f(n)}{g(n)} = 0$

ie) $f(n) = o(g(n))$

$\cancel{g(n) \le c f(n)}$

$\cancel{f(n)}$

means $\forall\, c > 0$, $\exists$ ~~no~~ n_0 s.t $f(n) < c\, g(n)$
$\forall\, n \ge n_0$

Ex $\sqrt{n} = o(n)$

$f(n) = O(g(n))$
$h(n) = o(g(n))$
$f(n) > h(n)$ as $n \to \infty$.

Qn
What is the maximum no. of steps needed by M_1 to halt for a string of length n?
Ans: $O(n^2)$.

Qn
Is there a TTM that accepts A in $o(n^2)$?

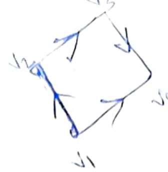
M_2: On input w:
1. Scan across & reject if $\exists$ 0 right of 1
2. Repeat as long as both 0's & 1's on tape:
3. { Scan accross checking total 0's & 1's remaining is even or odd. Reject if odd.
4. Scan accross crossing off every other 0 starting with first 0. Same for 1's }.
5. If no 0's & no 1's remain accept. O.w reject.

Ans: $O(n \log n) = o(n^2)$

Qn

Is there a TTM for A that halts in $O(n)$?
Ans: ~~No~~.
No however one can do it if more than one tape is allowed.

Yes with 2 tape TM.



(v_1, v_2)
(v_2, v_3)
(v_3, v_4)
(v_1, v_4)

~~Thm~~ Any language that is decidable in $o(n \log n)$ by a single tape TM is regular. (small o)

Proof?

Defn:

A language A is said to be in $\text{Time}(f(n))$ for $f: \mathbb{N} \to \mathbb{R}_+$ if A is decided by an $O(f(n))$ TM.

Thm:

Let $t(n)$ be a fn with $t(n) \geq n$. Then every $t(n)$ time multi-tape TM has an equivalent of $O([t(n)]^2)$ single tape TM.

Class P

P is the class of languages that are decidable in polynomial time by some 1-tape TM

$$P = \bigcup_{k \geq 0} \text{Time}(n^k).$$

Examples of Problems in P

1. For a given directed graph decide if $\exists$ a direct path between two vertices is
2. Given 2 integers decide if they are relatively prime

Algorithm:

E = On input $\langle x, y \rangle$
1. Repeat until $y = 0$
2. Assign $x \leftarrow x \bmod y$.
3. Exchange x & y
4. Output x.

$y^2 + 2^2 \cdot p^2 + \ldots + x^2 y^2$ time.

R : On input $\langle x, y \rangle$
1. Run E on $\langle x, y \rangle$
2. If result is 1 accept. o.w reject.

s | 6   y = 6   3
x < 5   n

Q   S.T algorithm (*) is in P but any simple modification into testing composit is not

Reduction & Gödel's Incompleteness

similar

Pbm 1: Does ∃ a TTM that decides if a given TM has atleast 1000 states? (Yes)

Pbm 2: Does ∃ a TTM that decides if a given TM accepts ϵ?. (No)

IMP **Proof**

Reduction Technique.

- We shall show existence of such TTM will render Halting problem decidable leading towards a contradiction
- For a given TM M & an input x construct from a new TM M_x' as follows:
  - (i) M_x' takes any input y erases it & writes x on its tape
  - (ii) Runs M on input x.
  - (iii) Accepts if M halts on x!!

$$L(M_x') = \begin{cases} \Sigma^* & \text{if } M \text{ halts on } x \\ \phi & \text{if } M \text{ doesn't halt on } x. \end{cases}$$

- If a TTM exists that decides if any given TM accepts ϵ then apply that on M_x' & that will decide the halting pbm!!

Why? : M_x' accepts ϵ iff M halts on x!

Godel Incompleteness:

1. Statements in Number Theory are either true or false under natural interpretation
2. One can build a "formal language" for number Theory.
3. $Th(\mathbb{N}) = \{$ True statements about number Theory

Gödel's Incompleteness Thm:

$Th(\mathbb{N})$ is Not r-e.

∴

What is a Thm in some (Finite) Axiom System containing ⟹

Reasonable axioms systems contains ⟹

1. A proof of statement φ_n is a sequence - $\varphi_0 \varphi_1 \cdots \varphi_n$ of formulas s.t each φ_i is either an axiom or follows from formulas earlier in the list by a rule of reference.
2. One example of Axiom system for Number Theory is Peano Arithemetic.

$$\left\{ \begin{array}{l} \bullet\ (\varphi \text{ and } \psi) \Rightarrow \varphi \\ \bullet\ (\forall x\ \phi(x)) \Rightarrow \phi(17). \\ \bullet\ \forall x\ \forall y\ \forall z\ ((x=y) \text{ and } (y=z)) \Rightarrow x=z) \\ \text{\& so on} \end{array} \right.$$

3. A proof system is said "sound" if all Thms are true one can show PA is sound.

$Th(\mathbb{N})$	$\neg Th(\mathbb{N})$
$\mp \varphi$ Thms of	