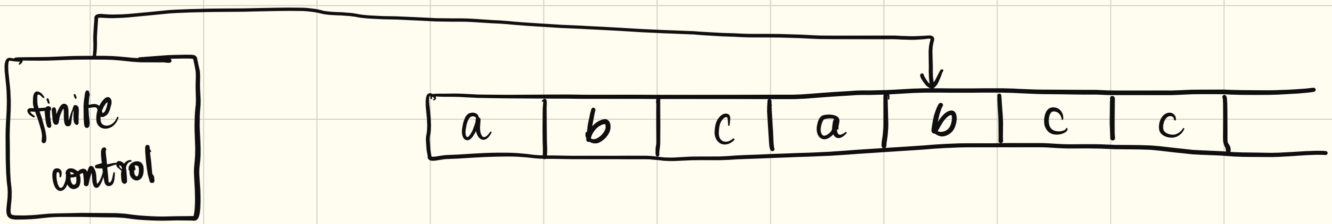
19 Mar 2025 – Theory of Computation – Week 11

Turing Machines



- a state q
- a storage tape (infinite)
- a tape head that can read / write and move left or right (one cell at a time)
- accept or reject by entering $\underbrace{q_{accept} \ / \ q_{reject}}_{\text{halting state}}$

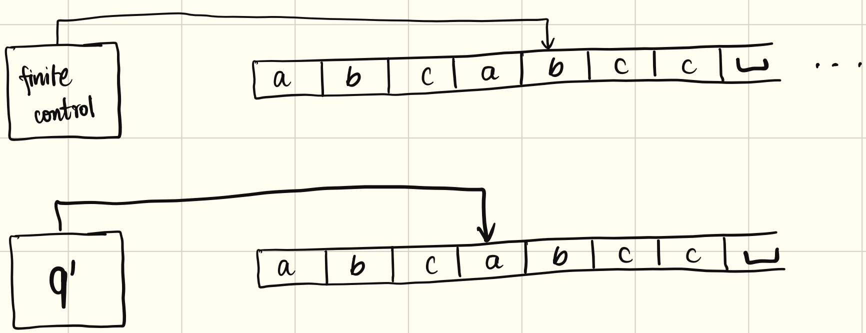
$$(q, b) \longmapsto (q, a, L)$$

write symbol on the cell

move one cell left or right on the tape

DFA → construction makes the final state ~halt~ (?)

" In state q reading b,
then write 'a' at that cell,
move a cell to left and enter the state q' "



$(Q, \Sigma, \Gamma, \delta, q_0, q_{accept}, q_{reject})$

Q := finite set of states

Σ := input alphabet

Γ := tape alphabet s.t. $\Sigma \cup \{\sqcup\} \subseteq \Gamma$

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

q_0 := start state

q_{accept} := accept state

q_{reject} := reject state

Stay on the same cell
L, then R.
(temporary state)

If you reach q_{accept} / q_{reject} the machine halts/stops/terminate

---

Not halting $\Rightarrow$ looping

$\underbrace{\text{Input tape, output}}_{\text{same}}$

only one tape : read/write

Def^n: M accepts an input string $w \in \Sigma^*$ iff M enters the accept state q_{accept} on input w.

Def^n: The language of the machine

$$L(M) := \{ w \in \Sigma^* \mid M \text{ accepts } w \}$$

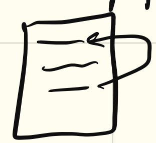
$$\text{"Looping"} := \underbrace{\text{never entering the states } q_{accept} / q_{reject}}_{\text{forever}}$$

$\sqcup \rightarrow$ denotes end of the input string at the beginning of computation

---

How did Turing come up with this?

He thought that this is just how humans compute, on paper



some times answer not possible

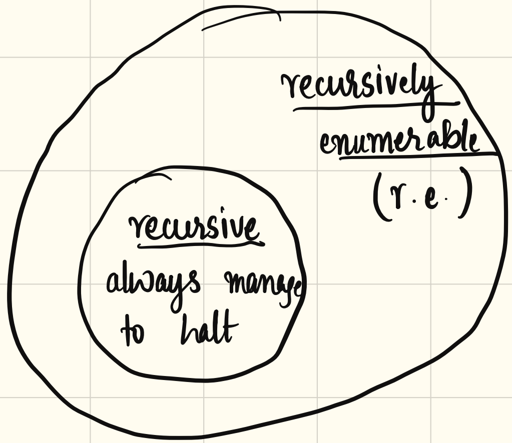
Def^n : We say that a language A is turing recognizable or recursively enumerable

if $\exists$ a M (turing machine), s.t $A = L(M)$

Def^n :- A TM M is said to be "total / decider" if it halts on every input $w \in \Sigma^*$

Def^n :- A language B is said to be recursive or turing decidable if $\exists$ a total TM M s.t $B = L(M)$

In practice, we need total turing machines



- Halting problem, later

$A = \{ ww \mid w \in \{0,1\}^* \}$ ⟶ non-CFL (exam)

How to construct a turing machine?

a^L ——— - ——^{L} | ^{R} - - ——— - b^R

reach
the middle

start maching

(need more
elements in F)

Bug verification

20 Mar 2025

$$M := (\overbrace{Q, \Sigma, \Gamma}^{\text{Finite}}, \delta, q_0, q_a, q_r)$$

where $\quad \Sigma \cup \{\sqcup\} \subseteq \Gamma$

and $\quad \delta : Q \times \Gamma \longrightarrow Q \times \Gamma \times \{L, R\}$ $\quad \}\rightarrow$ finite

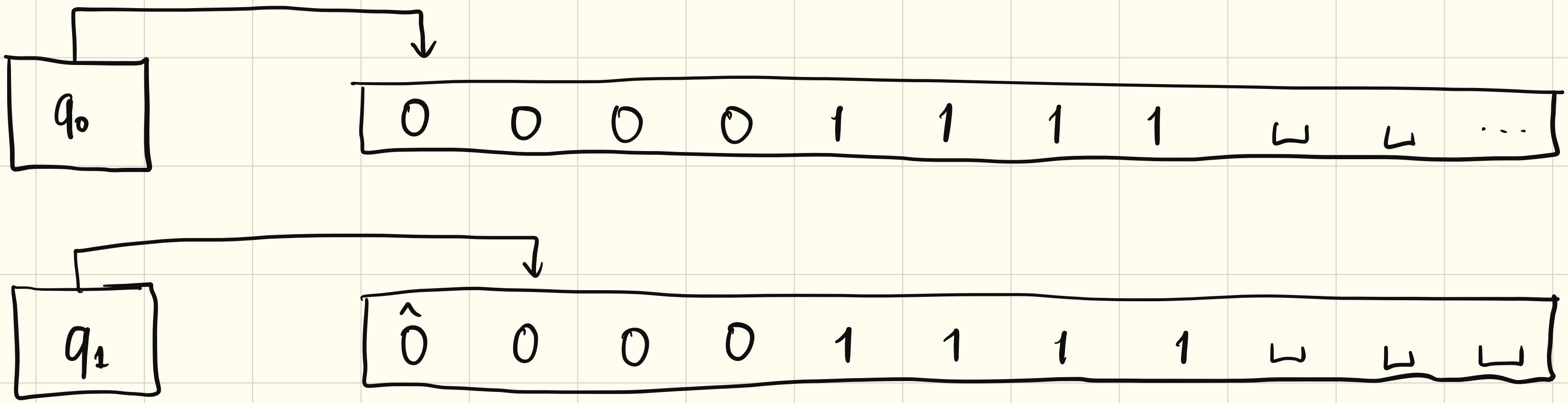
$$L(M) := \{x \in \Sigma^* \mid M \text{ accepts } x\}$$

if zero, add another state (see table below)

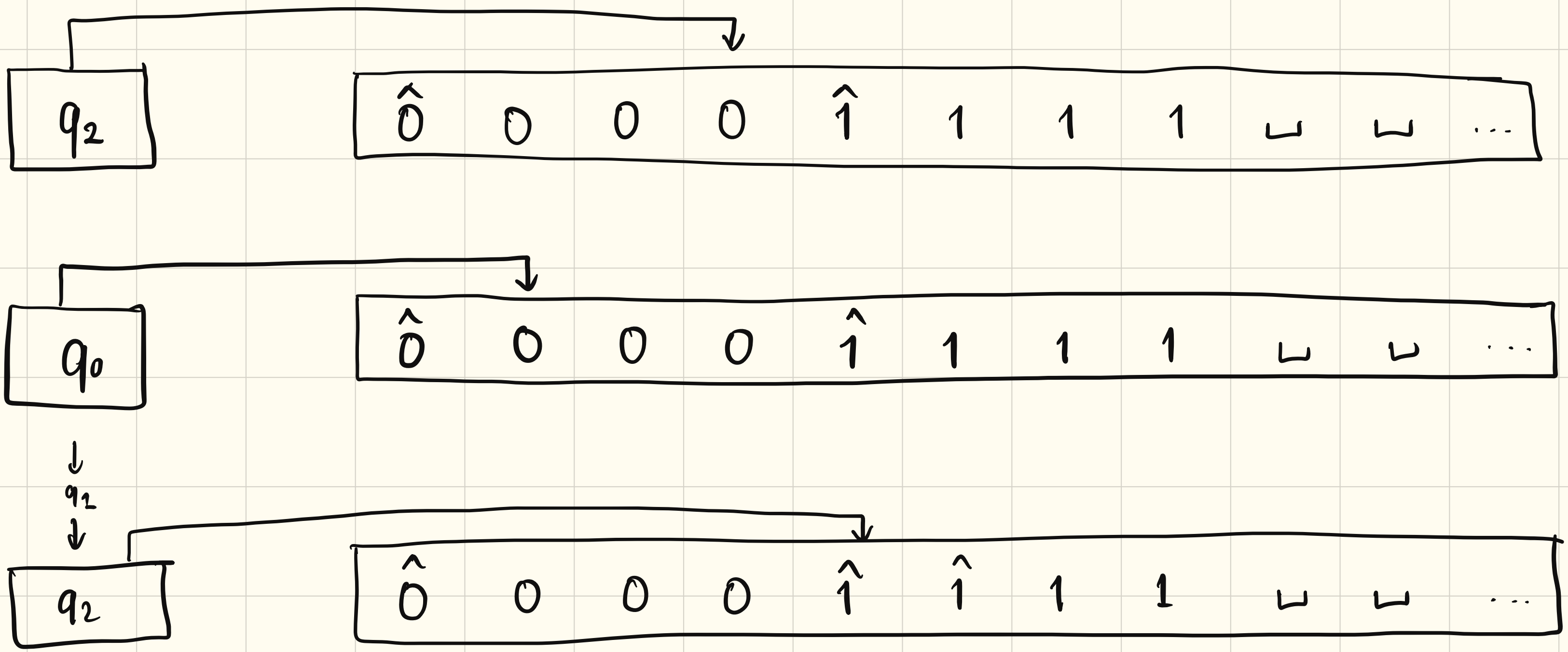
$$A = \{0^n 1^n \mid n \geq 1\}$$

Construct M_1 as follows

$$\Gamma = \{0, 1, \sqcup, \hat{0}, \hat{1}\}$$



q_1 : looking for 1



$Q \backslash \Gamma$	0	1	$\sqcup$	$\hat{0}$	$\hat{1}$
q_0	$(q_1, \hat{0}, R)$	$(q_r, \ldots, \ldots)$	$(q_r, \ldots, \ldots)$	$(q_r, \ldots, \ldots)$	$(q_4, \hat{1}, R)$
q_1	$(q_1, 0, R)$	$(q_2, \hat{1}, L)$	$(q_r, \underbrace{\sqcup}_{\text{write something}}, L)$	$(q_r, \ldots, \ldots)$	$(q_1, \hat{1}, R)$
q_2	$(q_2, 0, L)$	$(q_r, \ldots, \ldots)$	$(q_r, \ldots, \ldots)$	$(q_0, \hat{0}, R)$	$(q_2, \hat{1}, L)$
q_4	q_r	q_r	$(q_a, \ldots)$	$(q_r, \ldots)$	$(q_4, \hat{1}, R)$

$\hat{0} \quad \hat{0} \quad \hat{0} \quad \hat{0} \quad \hat{1} \quad \hat{1} \quad \hat{1} \quad \hat{1} \quad \underbrace{\sqcup}$

if anything else, reject

q_4: all zeroes at the start have been matched

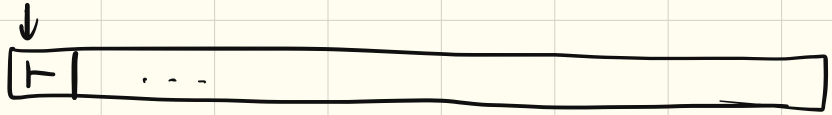
What happens when the machine moves left on the leftmost cell?

Two options

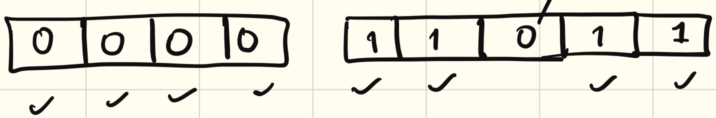
→ Keep it at the leftmost cell.

→ Add another cell $\boxed{\vdash}$

denotes the start



q_4 will reject this



For proof: modify the machine to first check $0^* 1^*$ → makes proof easier

We don't care about the time costs for now.

- Computability theory
- Complexity theory

Hilbert : Is math decidable?

Given $P(x_1, \dots, x_n)$

$$= x_1^2 + 2x_1 x_2 + 9x_3 x_2 + \cdots$$

$$+ \text{------} = 0$$

Find integer roots.

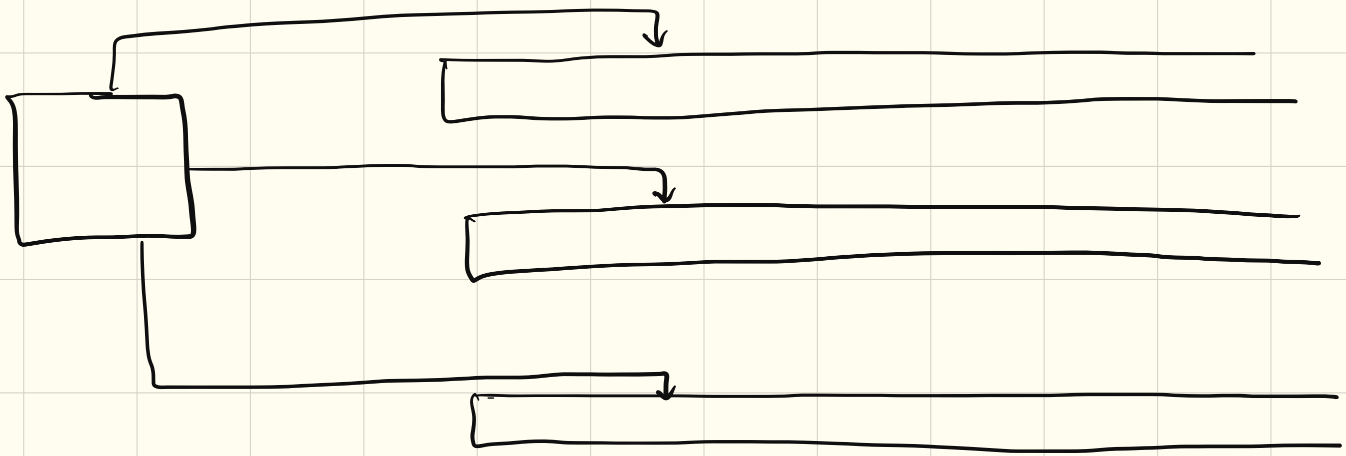
'Give me a procedure at the end of which I can decide whether P has integer sol^n'

$B = \{ P \mid P \text{ has integer roots} \} \longrightarrow$ is B recursive?

Multiple tapes (finite, same for all inputs)

$\cong$

Single tape

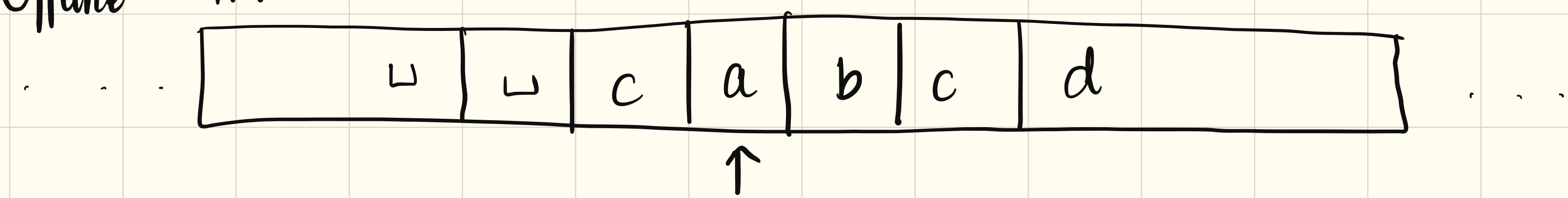


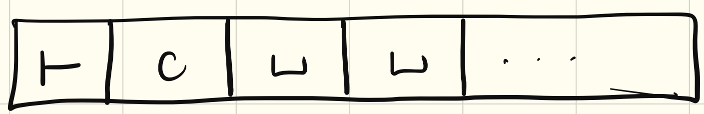
26 Mar 2025

- Last class notes

Variants of TM

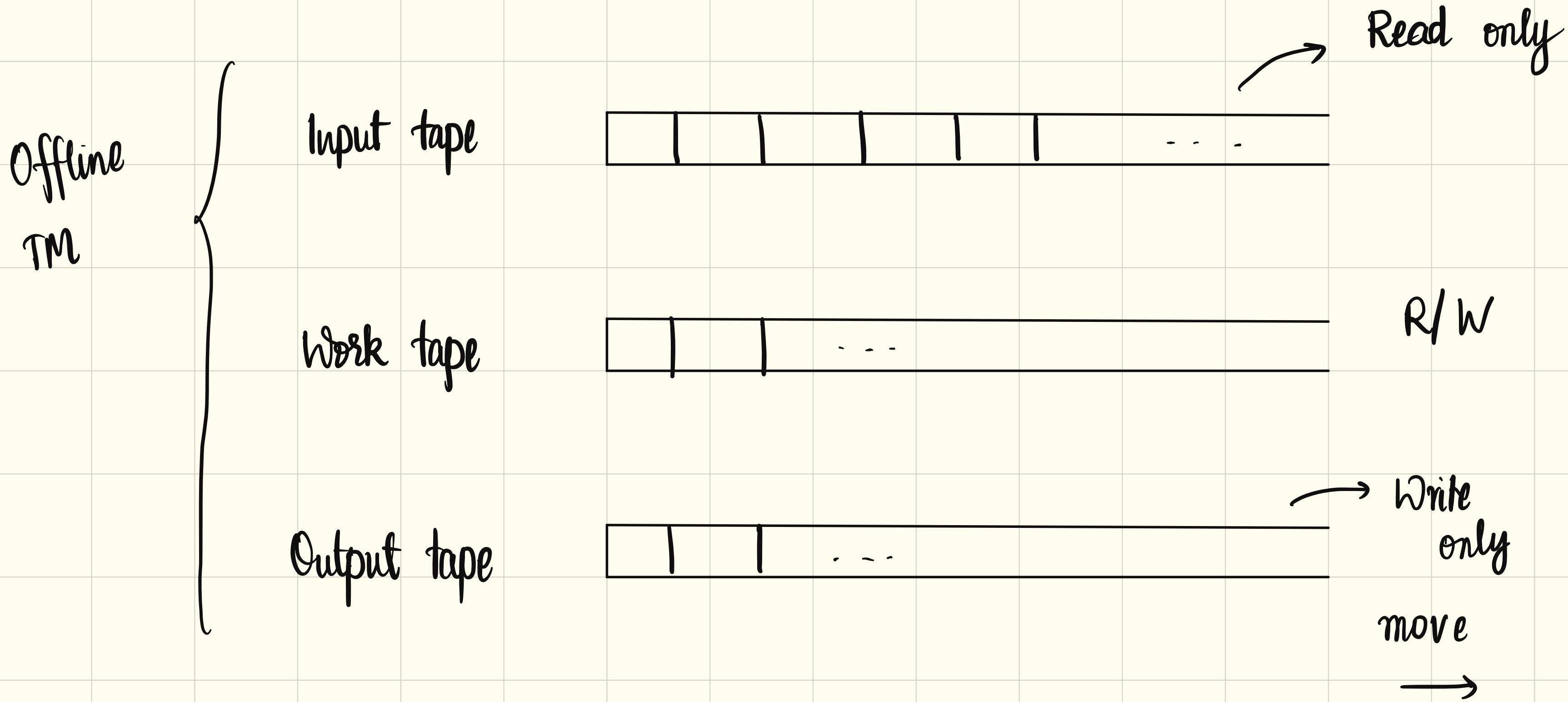
- Multitape TM $\longrightarrow$ equivalent
- Non-deterministic TM
- Two-way infinite tape TM $\longrightarrow$ equivalent? $\longrightarrow$ prove equivalent to multitape
- Offline - TM





→ time and space → resources

→ we want to optimize both.



→ Separates the computation

→ Output could use n bits while work tape would need only $\log n$ bits

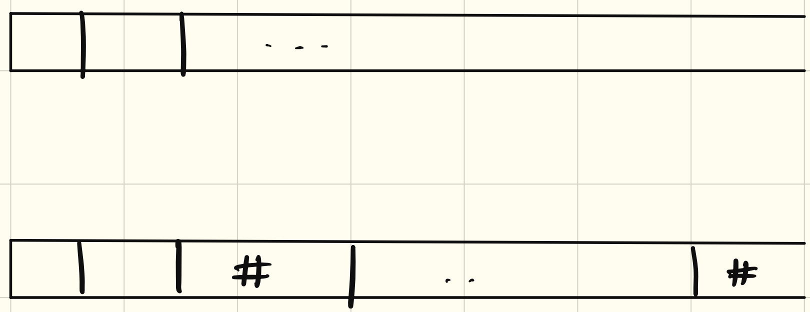
→ Another version ⟶ each cell has registers (?) that can do basic add subtract operation

→ RAM model

→ Enumerator / Enumeration machines ⟶ does not have input tape

Work tape

Output tape



at certain point it keeps printing out a string $x \in \Sigma^*$

e.g: print all the primes

is 1 a prime $\xrightarrow{\text{comp.}}$ print NO

2

3

⋮

Language of such a machine = all strings generated throughout its lifetime

→ may or may not halt

→ halt ⇒ finite language ⇒ regular

recognized by a single tape TM

Theorem: L is recursively enumerable iff L is generated by an enumerator

→ Name

→ In the beginning they were only generating, not accepting or rejecting

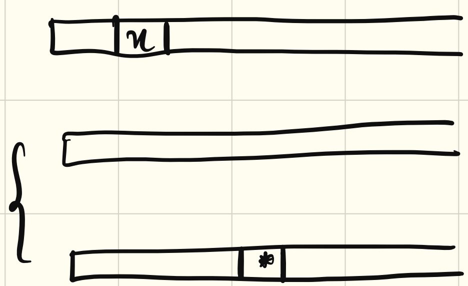
($\Longleftarrow$) Let M be an enumerator generating L

→ Single tape TM

S: on input x

behaviour: accepts $x \iff x \in L \iff x \in G(M)$

→ keep waiting



M

($\Longrightarrow$) Let S be a TM recognizing L

M:
generate all inputs $\in \Sigma^*$
Run S on x
print x if S accepts

$\rightarrow$ Problem:
For some input S may not halt

Keep an upper bound on number of operations $\rightarrow$ What bound?

Solution: For $i = 1, 2, \ldots$

Run S for i steps on $x_1, \ldots, x_i$

Print those x_i's where computation halts in accept.

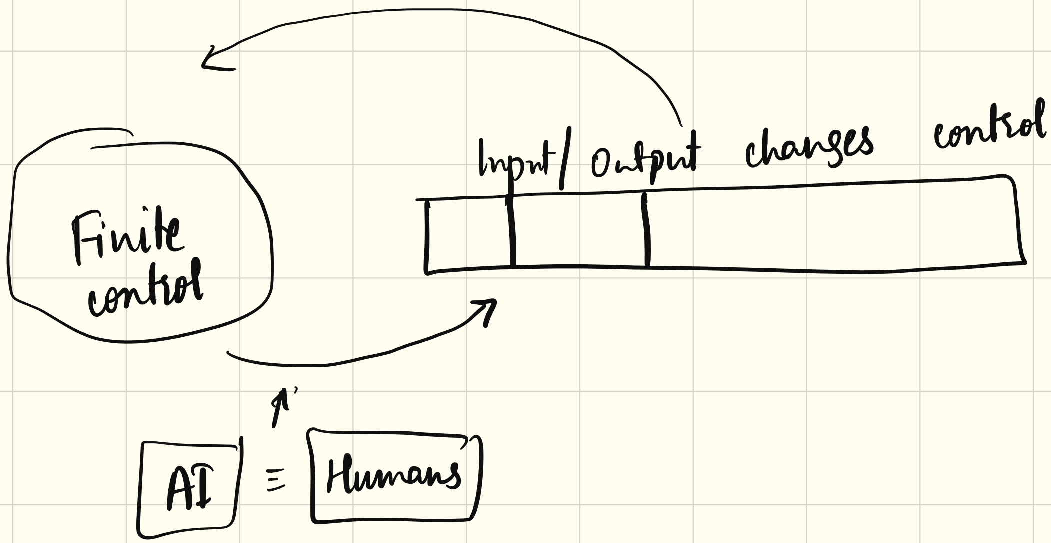
End for.

Makes sure that every string is considered for some finite no. of steps.

all

$i = 1, 2, \ldots$

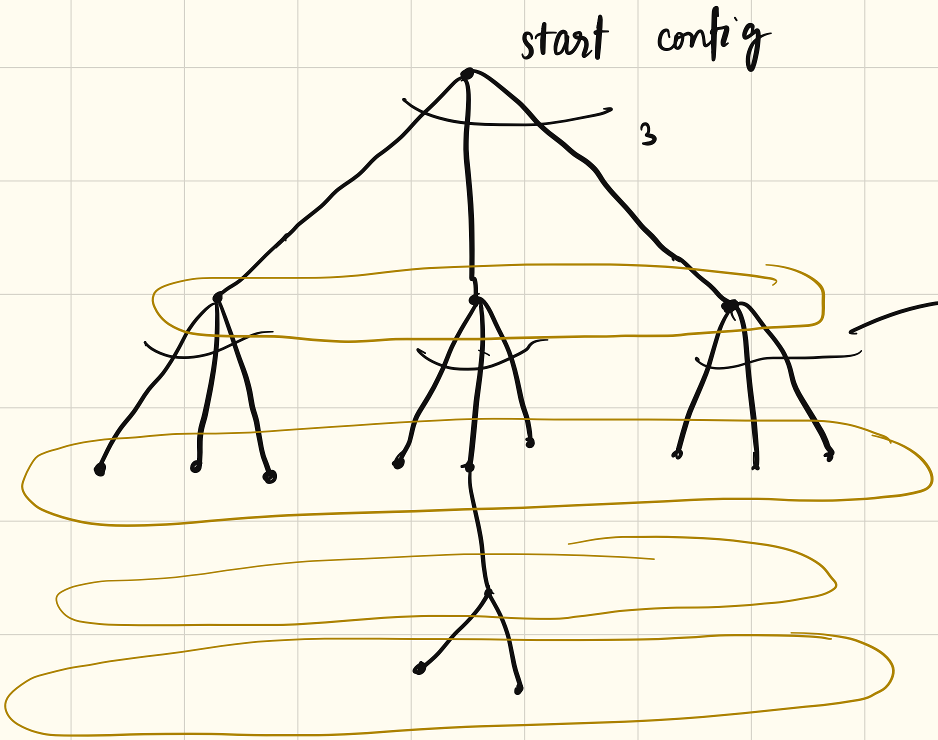
Can a turing machine modify itself ?



→ Non-deterministic Turing Machine

$$\delta : Q \times \Gamma \longrightarrow \mathcal{P}(Q \times \Gamma \times \{L, R\})$$

??

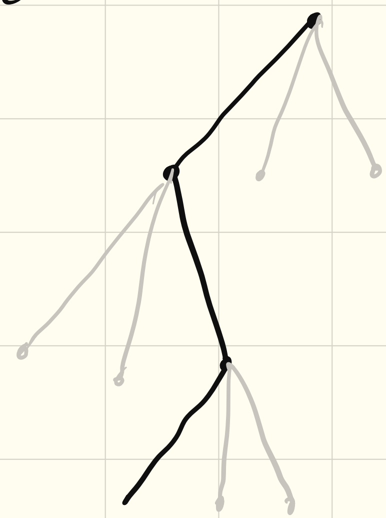


finite no. of possible configs

Make space to store all the information

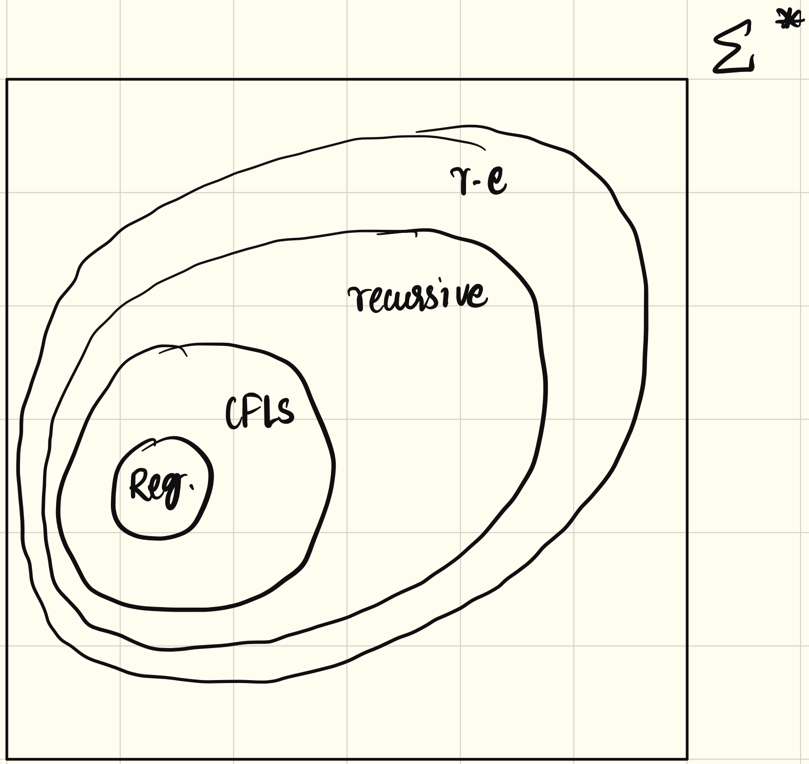
Alternative: Make guesses

Try all possible choices and check



27 Mar 2025

→ No Quiz 4



Q1 If L is a CFL, then, is L recursive?

if $\exists$ a TM $\xrightarrow{}$ decider M s.t

$L(M) = L$

Q2: If L is regular, then is L recursive?

↓

DFA ⇝ never get stuck in ε-transition loops

because not in DFA

∴ Yes

Q1: → CFL → CFG in CNF form

implement CKY algorithm → taught earlier, probably

??

Q3 If L is recursive, then is $\bar{L}$ recursive?

Yes, swap qaccept and qreject for M that accepts L.

Q4 If L is r.e. then is $\bar{L}$ r.e. too?

Swapping won't work

→ there can be many cases of looping → won't be accepted in either

Q5 If L is r.e.,
and $\bar{L}$ is r.e.

then is L recursive?

$$\left.\begin{array}{llll} M_L & \text{accepts } x & \text{iff} & x \in L \\ M_{\bar{L}} & \text{accepts } x & \text{iff} & x \in \bar{L} \end{array}\right\} \longrightarrow$$

run simultaneously

↓

1 step in each

use our variants of TM

↓

one of these will accept

M_L accept $\Rightarrow$ accept

$M_{\bar{L}}$ accept $\Rightarrow$ reject

Q4 If $\boxed{L \in r.e \text{ but not recursive}}$??

$\bar{L} \in re$ then L is recursive $\Rightarrow\Leftarrow$

$\therefore$ $L \in r.e$ does not always mean $\bar{L} \in r.e.$

→ Note: For now we are interested in solving yes/no questions, and we want a recursive TM

recursive $\Leftrightarrow$ decidable

Given an integer x; decide if x is prime?

$$L_{prime} = \{ p \mid \text{prime } p \}$$

$x \in L_{prime}$?

Undecidable $\iff$ not recursive

Detour: Cardinality of Sets

We say that A and B have same size if $\exists$ a $f : A \to B$ s.t f is one-one and onto.

bijective.

$\longrightarrow$ works for both finite and infinite sets.

Ex: $\mathbb{N} = \{1, 2, \ldots\}$

$\mathbb{Z} = \{\ldots, -2, -1, 0, 1, \ldots\}$

$\mathbb{N} \subset \mathbb{Z}$ but $|\mathbb{N}| = |\mathbb{Z}|$

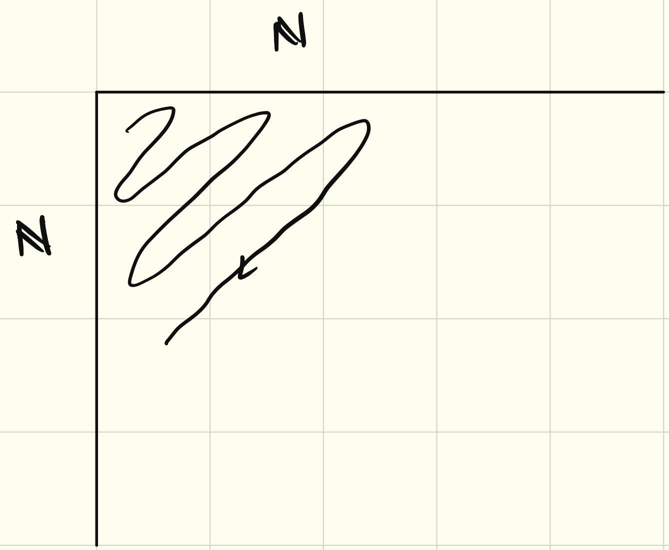
1	0
2	-1
3	1
4	-2
⋮	⋮
	x

i

A is said to be countable if $|A| = |\mathbb{N}|$

bijection from A to $\mathbb{N}$

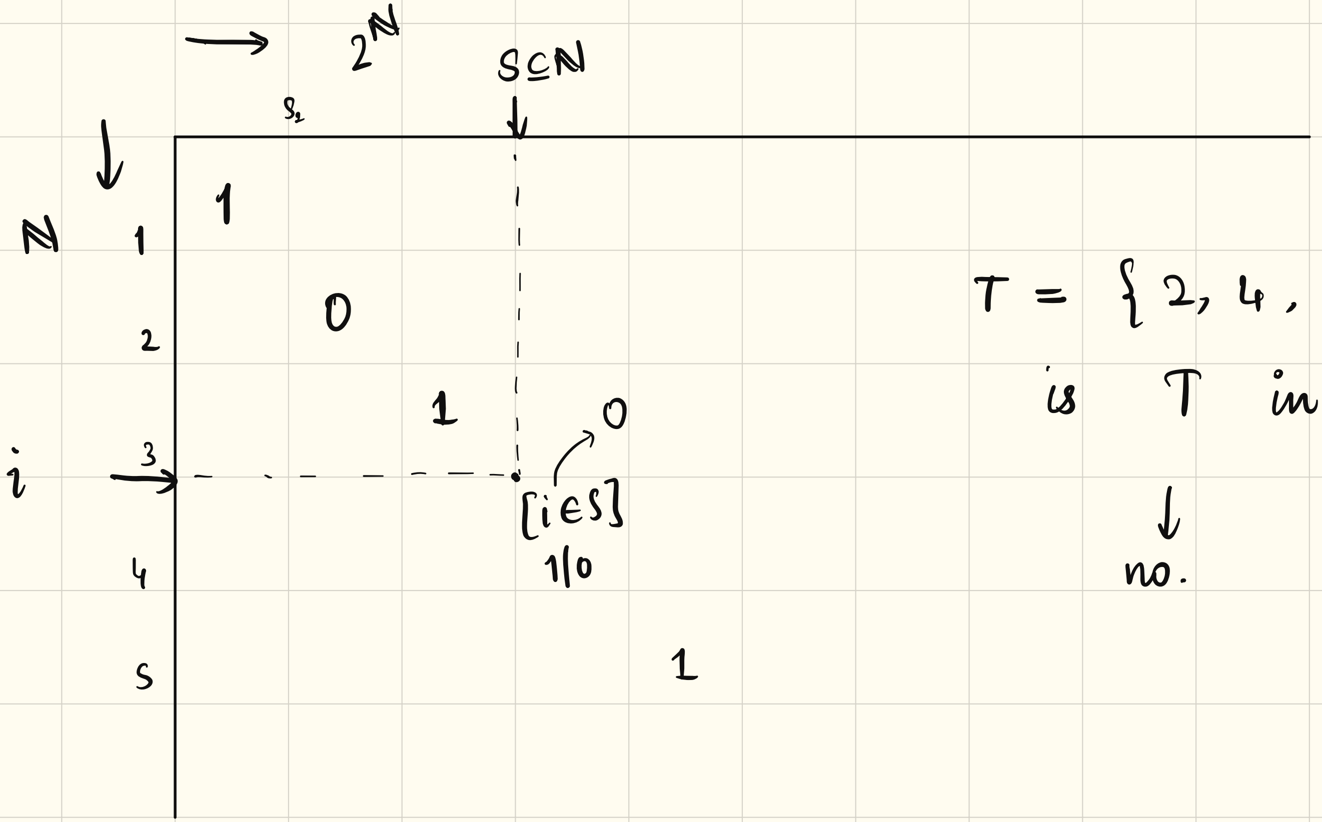
Q: Is $|\mathbb{N} \times \mathbb{N}| = |\mathbb{N}|$?



Q : $\mathbb{N}$ vs. $2^{\mathbb{N}}$ or $\mathcal{P}(\mathbb{N})$

Proof. by Contradiction: Suppose $\exists$ a bijection

$$f : \mathbb{N} \to 2^{\mathbb{N}}$$



$T = \{2, 4, \ldots\}$

is T in $\mathcal{P}(\mathbb{N})$?

$\downarrow$

no.

$T \neq S_j$ for any j because if $j \in S_j$ then $j \notin T$ o/w $j \in T$.

Doubt: Construction of T takes ∞ time

$\downarrow$

It's not about time, it's about existence.