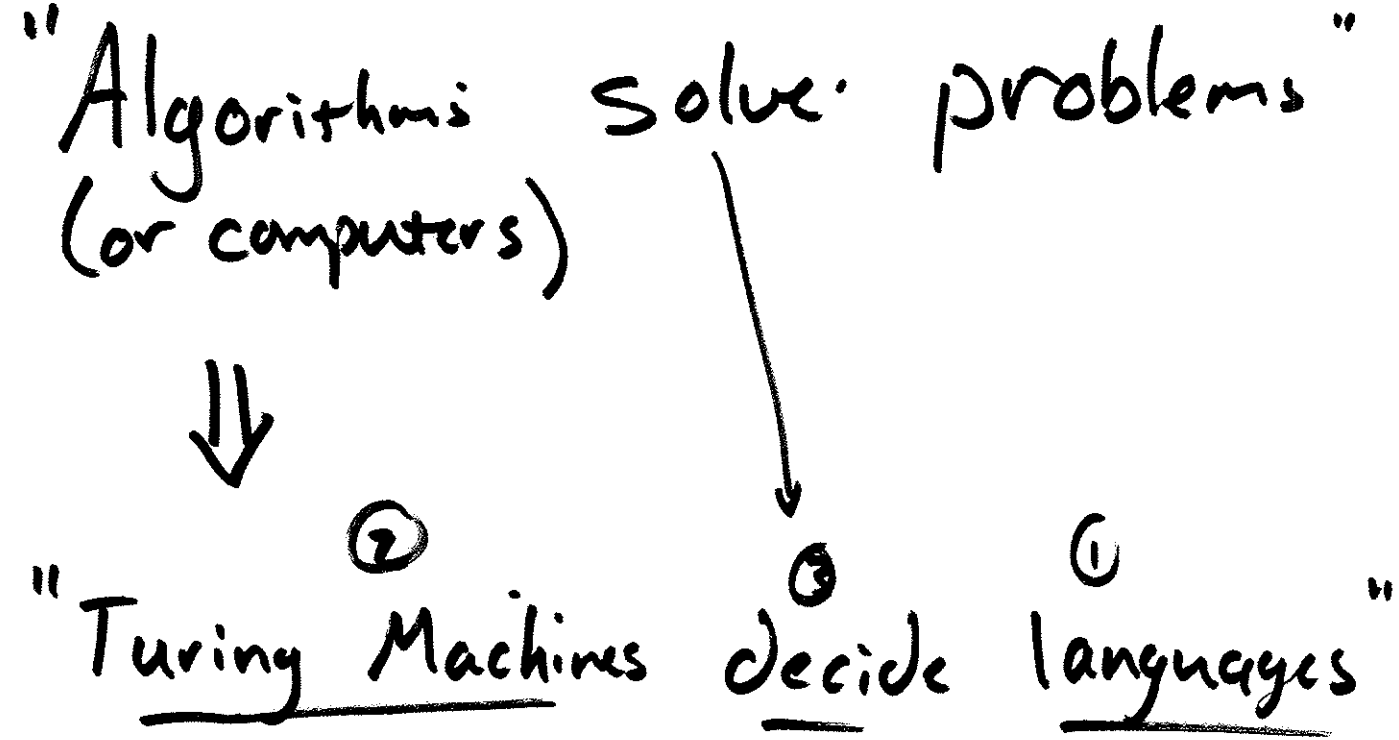
Computation

- Computational Models
- Limits of Computation
- Complexity
- Reductions

Motivation: What is computation?

What is computable



Church - Turing Thesis

Defn

- An alphabet, Σ is a finite set of symbols
- A string is a finite sequence/combination of symbols from Σ
- It suffices to consider $\Sigma = \{0, 1\}$

Justification: any alphabet can be encoded with $\{0,1\}$

→ Huffman Coding

→ ASCII, Unicode

String: $\lambda, 0, 1, 00, 01, 10, 11, 000, 001, \ldots$
(word)

↑ empty string: string of length zero

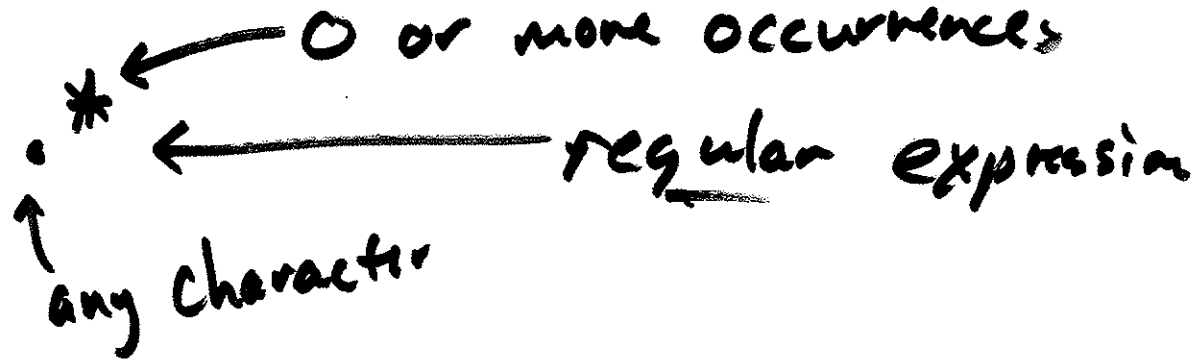
Alt: ϵ

~~"", NULL, None, etc~~

The set of all strings over $\Sigma = \{0, 1\}$ is denoted

$$\Sigma^* = \{\lambda, 0, 1, 00, 01, \ldots\}$$

- infinite set
- contains every possible finite string
- Star operator (Kleene Star): 0 or more combinations of ~~*~~ symbols from Σ



\b* \b+ ← 1 or more occurrences

\b*

any white space character.

(\b\b)* any number of even white space characters

grep

pcre

Defn

A language is a set

$$L \subseteq \Sigma^*$$

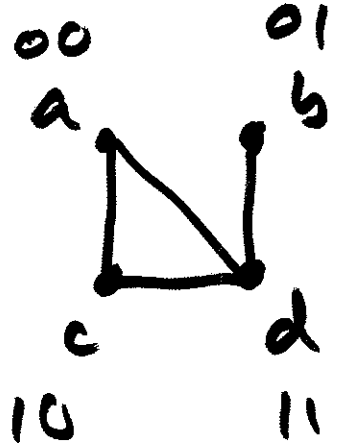
- it is simply a set (finite or infinite)

of strings over $\Sigma = \{0, 1\}$

Fact: any problem can be "encoded" as a language.

Let G be a graph, x, y be vertices in G,

Q: (problem): does there exist a path $x \rightsquigarrow y$?



$$\begin{bmatrix} 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{bmatrix}$$

G = encoding of n, adj matrix

$\langle G \rangle$ is an encoding of some graph G

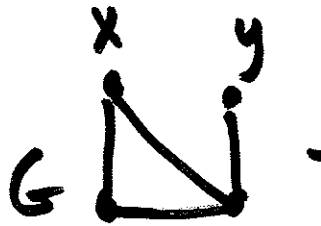
00:01; 10:11; 00110001 ... | 00:01 (x, y)

$\langle G \rangle$ is a string over Σ

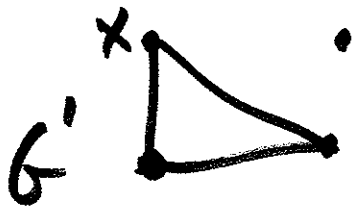
$\langle G, x, y \rangle$ is simply a binary string

Language:

$$L = \{\langle G, x, y \rangle \mid G \text{ is a graph and There exist a path } x \rightsquigarrow y\}$$



$\rightarrow \langle G, x, y \rangle \in L$ because a path $x \rightsquigarrow y$ exists



$\rightarrow \langle G', x, y \rangle \notin L$

← set complement

$\overline{L} = \{ \langle G, x, y \rangle \mid G$ is a graph, there is no path $x \rightsquigarrow y \}$

Answering the question: does there exist a path $x \rightsquigarrow y$ in G

$\Rightarrow$ is $\langle G, x, y \rangle$ in the language L?

string

set of strings (or not)

Defn

A computational model decides a language if for any input $x \in \Sigma^*$, it eventually halts (end) its computation an answers yes or no.

Yes/No: decision problems

$\langle G, x, y \rangle$: Does there exist a path $x \rightsquigarrow y$

Decision ↗ Yes or no?

Given G, x, y: what is the length of the shortest path $x \rightsquigarrow y$

Optimization version ↗

$\langle G, x, y, k \rangle$ = encody of a graph G, vertices x, y, integer k

is there a path of length $\leq k$ $x \rightsquigarrow y$

Functional: Given G, x, y, output the shortest path p : $x \rightsquigarrow y$

A: a sequence of vertices $x, v_1, v_2 \cdots y$

$\langle x, v_1, v_2 \cdots, y \rangle \in \Sigma^*$

Decision Version

$\langle G, x, y, z, i \rangle$ G, x, y, z vertices; i integer

Q: in the shortest path $x \rightsquigarrow y$ is the ith vertex z?

Finite State Automata (FSA)

- very simple, limited computational model
- Def: A FSA is a 5 tuple:
  - Q a set of states
  - Σ, our alphabet $\Sigma = \{0, 1\}$
  - $\delta : Q \times \Sigma \rightarrow Q$

    given the current state and the current input bit, transition to a new state

- q_0 is an initial state
- $F \subseteq Q$ are accept states
  (all others are reject states

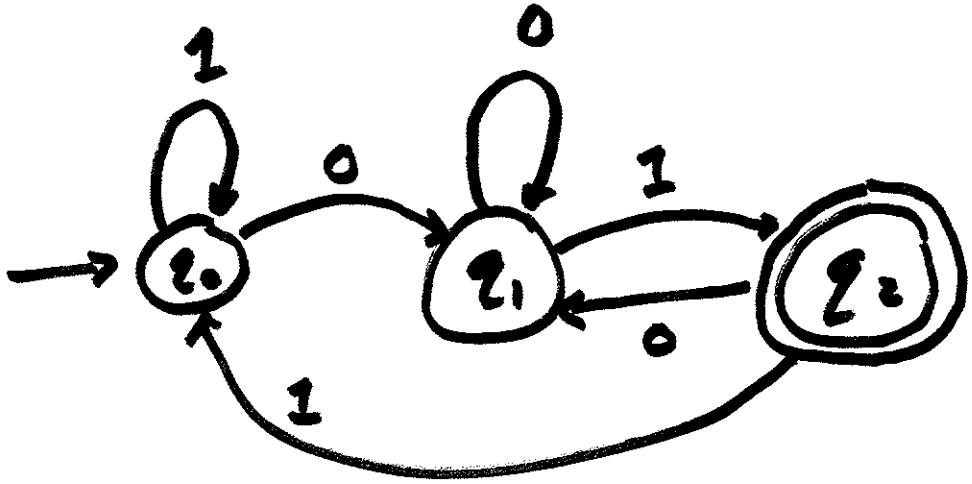
Intuition: FSA will:

- read input bit-by-bit
  - no output (read only)
  - cannot reexamine input (one way)
- at ~~the~~ the end of the input it stops in some state:

- accept state: input $x \in L$
  The input is in the language L defined by FSA

or

- reject state: input $x \notin L$



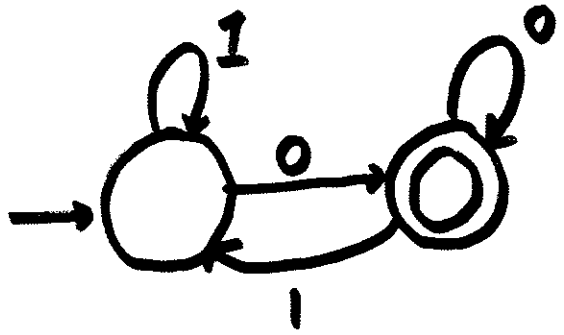
Strings	in or not?
0101	Y
1000	N
111	N
1111	N
01	Y
101	Y
1101	Y

q_0 initial state

(0) = accept state

L = { all strings ending in 01 }

FSA: accept any stry That ended with a zero



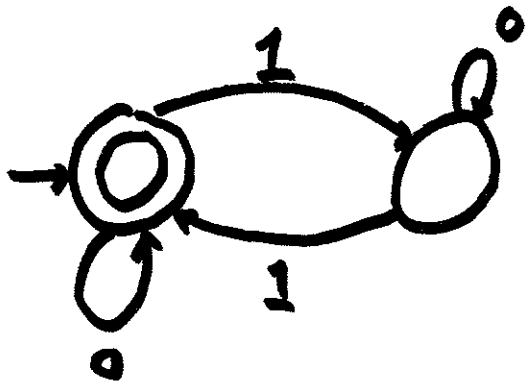
FSA decides even/odd

1110 = 14
110 = 6
100 = 4
10 = 2
0 = 0

Let x be a bitstring, parity of x is even(0) if it has an even number of 1s, odd(1) if it has an odd number of 1s

Given x: accept it it has an even # of 1s.

λ



words:

String	in?
0101	Y
1110	N

Given: an integer n, compute $\sum_{i=1}^{n} i = \frac{n(n+1)}{2} \stackrel{?}{=} k$

Gauss's Formula

$\langle n, k \rangle$?

Can a FSA count?

Given: an integer n, is it prime or not?

Given: an integer n, what is its prime factorization?

$O(n^n)$

$21 \rightarrow 3 \cdot 7$

$$L = \{ 0^n 1^n \mid n \in \mathbb{Z}^+ \}$$

$$= \{ 01, 0011, \underline{000111}, 00001111, \ldots \}$$

Fact: no FSA can decide L

it requires more computational resources

00011

While you read zeros in x:
- push 0 into S

while you see 1s:
- pop 0 from S

accept or reject?

if you run out of zero or it S still has zeros at the end → reject

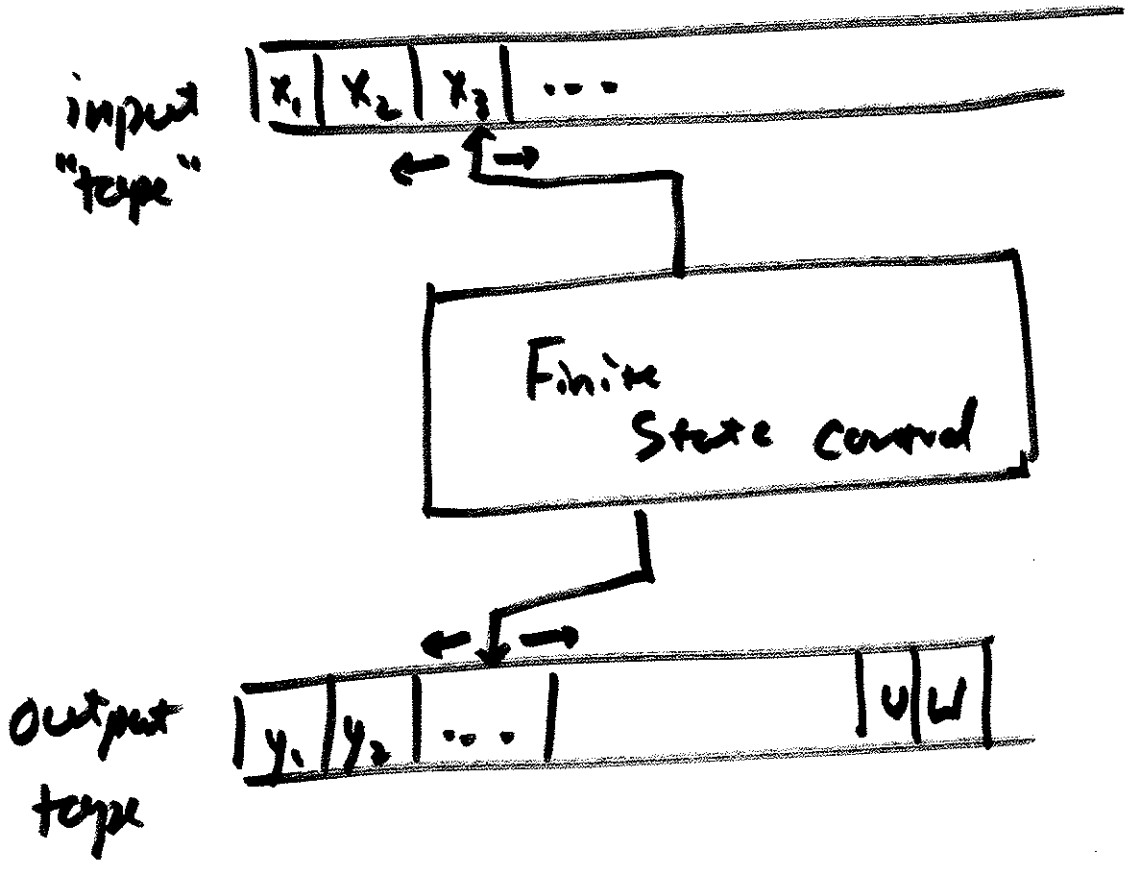
Regular Language → regular expression = FSA

ContextFree Language = Push down Automaton (Stack)

↑ language grammars

Decidable Language: Decided by a T.M.

Undecidable Language: NO TM exists for These

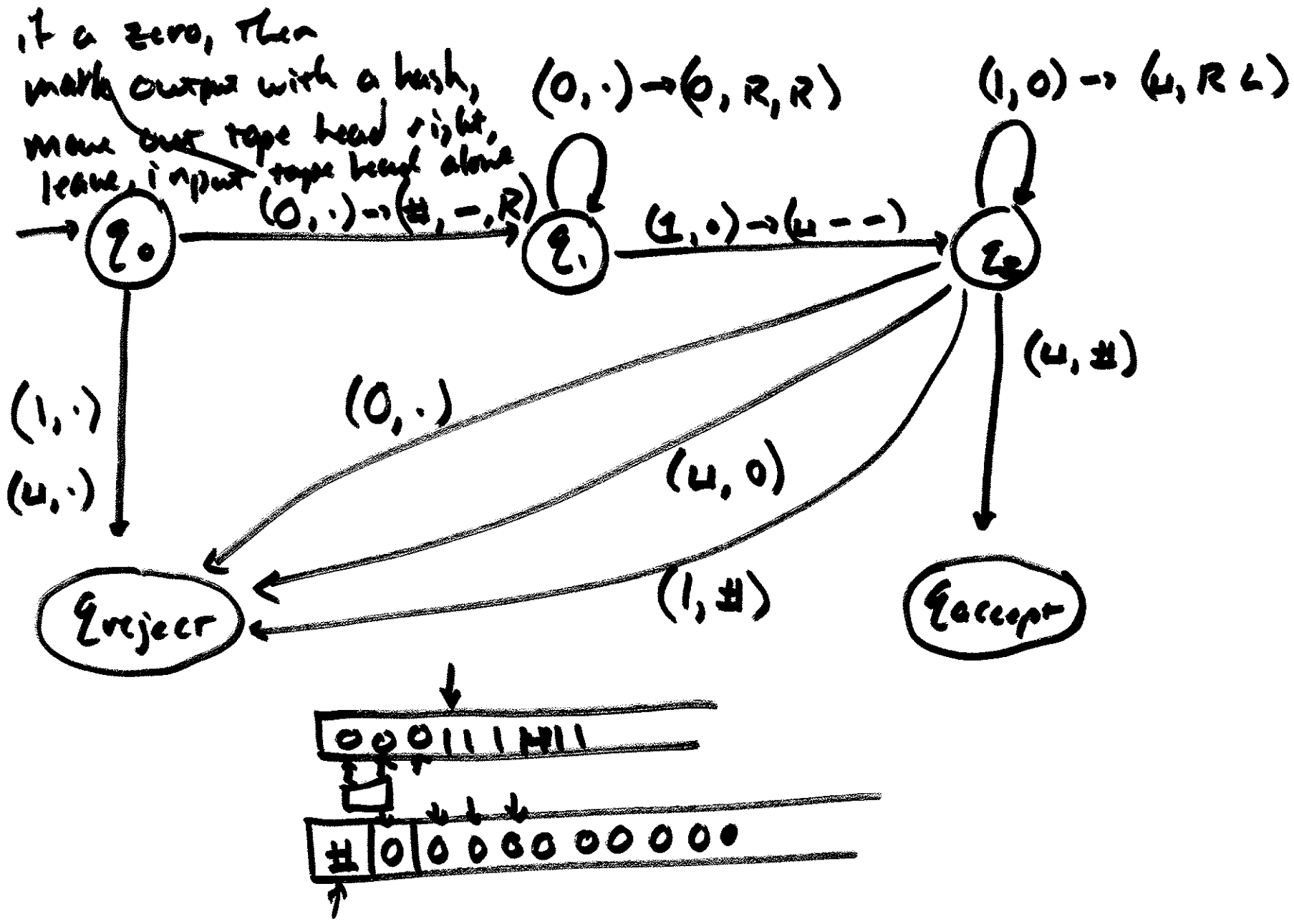

input "tape"
x_1 x_2 x_3 ...
Finite State control
output tape
y_1 y_2 ...
u u

␣ ␣␣␣

␣

$L = \{0^n 1^n \mid n > 0, n \in \mathbb{Z}\}$

(input, output)



000111

Algorithm = Turing Machine

Def: A Turing Machine (TM) is a 7-tuple:

- Q = set of states
- Σ = input tape alphabet → read only
- Γ = output tape alphabet → read/write
- $\Sigma = \Gamma = \{0, 1\}$
- $\delta : Q \times \Sigma \times \Gamma \to Q \times \Gamma \times \{L, R, -\}^2$
  - Q: current state
  - Σ: read input
  - Γ: current output bit
  - Q: new state
  - Γ: rewrite output bit
  - $\{L, R, -\}^2$: move left, right or none on both input output.
- q_0 initial state
- q_{accept} accept state
- q_{reject} reject state

Defn A Turing Machine M decides a language L iff for all inputs $x \in \Sigma^*$, $M(x)$ (Machine M run on input x) halts and for any $x \in L$, it halts in TM accept state; for any $x \notin L$, it halts in TM reject state

Can computers / algorithms solve any problem

$\Rightarrow$

For an language L, is there a TM that decides L?

No

Proof 1:

Fact 1 Turing Machines are finite

$\Rightarrow$ you can encode a machine as a string!

$\Rightarrow$ $\langle M \rangle$

Fact 2: Strings are countably infinite

$\Rightarrow$ ~~All TM~~ The set of all TMs is countably infinite

i.e. you can order them

$M_1\ M_2\ M_3 \ldots$

Σ 0 1 00 01 10 11 ...

Fact 3: The set of all languages is uncountably infinite.

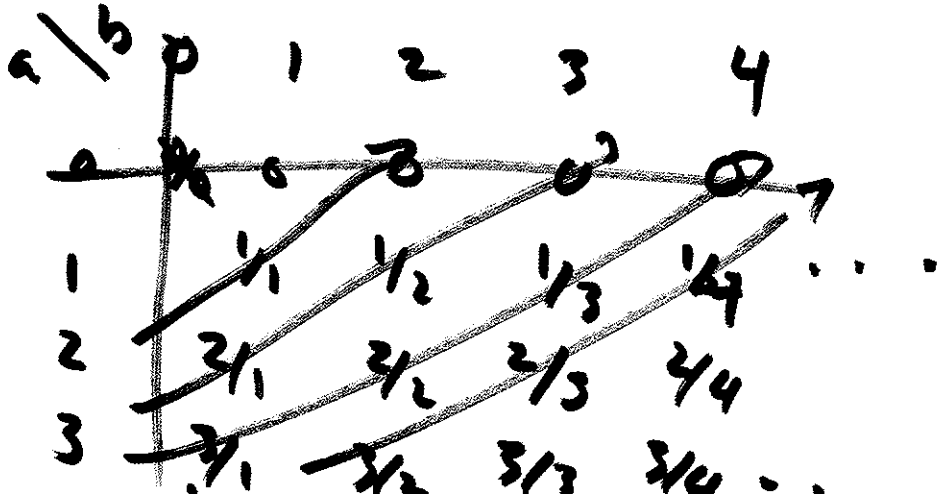
$\mathbb{N}$: 0 1 2 3 4 ...

$\mathbb{Z}$: 0 1 -1 2 -2, 3, -3

$\mathbb{Q}$: $\frac{a}{b}$ $b \neq 0$

$\mathbb{R}$?

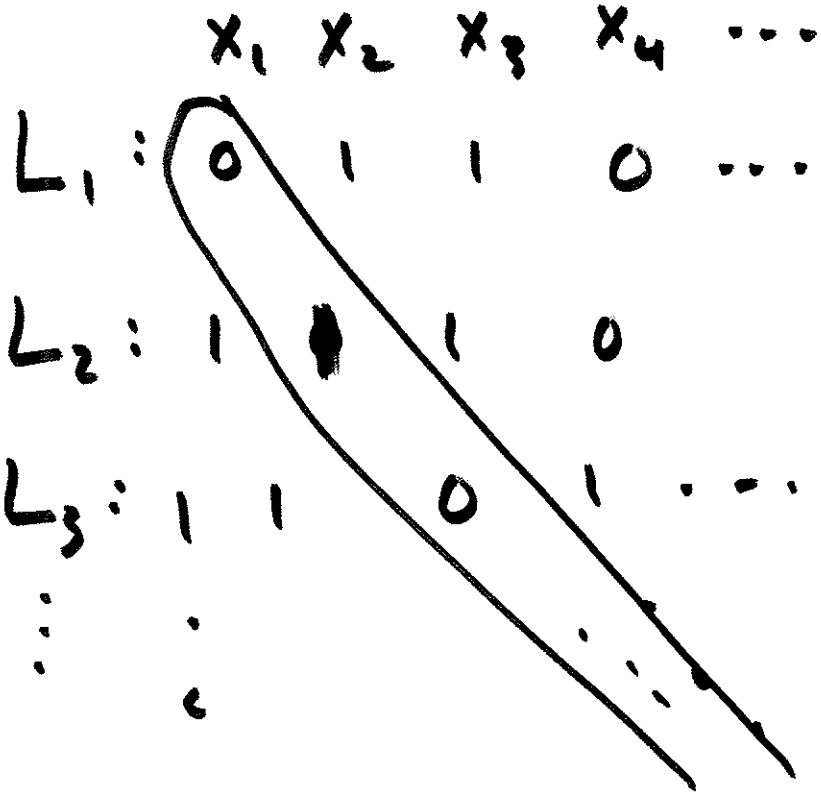
0 ... 1 2



Proof

Languages are uncountable:

By way of contradiction, suppose Languages are countable $\Rightarrow$ order $L_1, L_2, L_3, \ldots$



$0 \rightarrow x_i$ is in L_j

$1 \rightarrow x_i$ is not in L_j

New language:

$L' = \{ x_i \mid L_{x_i}$ does not rejects include $\}$

There are "more" languages than Turing Machines $\Rightarrow$ some languages do not have any TMs that decide them.

Fact: No TM exists That decides the Halting Problem

↳ HP: Given a TM M, and an input x,

Q: Does $M(x)$ halt or not (infinite loop)

proof:

By way of contradiction suppose such a TM exists.

$$H(\langle M, x \rangle) = \begin{cases} \text{halts, outputs 1} & \text{if } M(x) \text{ halts} \\ \text{halts, outputs 0} & \text{if } M(x) \text{ does not halt.} \end{cases}$$

Consider running a machine M on itself:

$M(\langle M \rangle)$

Consider: $\langle M, x \rangle \rightarrow \langle M, M \rangle$

emulators
operating systems
virtual machine
docker

$$Q(\langle M \rangle) = \begin{cases} \text{halt if} & H(\langle M, M \rangle) = 0 \\ \text{go into an infinite loop if} & H(\langle M, M \rangle) = 1 \end{cases}$$

while(1)
while(true)

$$Q(\langle Q \rangle) = \begin{cases} \text{halts if } H(\langle Q, Q \rangle) = 0 \\ \text{does not halt (infite loop)} \quad H(\langle Q, Q \rangle) = 1 \end{cases}$$