CPSC 421/501 Nov 16, 2021

- Universal Turing Machines

- Oracle TM's ← Ch 6, briefly [Sip], Then Ch 9

Main point(s)

① $\mathrm{HALT} \subsetneq \mathrm{HALT}^{\mathrm{HALT}} \subsetneq \mathrm{HALT}^{\mathrm{HALT}^{\mathrm{HALT}}} \ldots$

strict inclusions!

② Later: $P^A \neq NP^A$ for some A

$P^B = NP^B$ for any PSPACE-complete B

most interesting half, for most people

Idea: Say you want to show

$x^n + y^n = z^n$ has no solutions

for $n \geq 3$, $x, y, z \in \mathbb{N} = \{1, 2, \ldots\}$

=

Say: I have a proof!

And -- your dad notices:

it also works for

$n \geq 3$

$n \in \mathbb{N}$

and x, y, z positive reals

proof doesn't distinguish

If so, your proof ☹

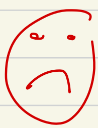
From last time:

We want to "standardize" all Turing machines

TM abstractly:

$$(Q, \Sigma, \Gamma, q_0, \delta, q_{acc}, q_{rej})$$

sets



"too many"

You get the "same algorithm" and you decide/recognize the same language, if you...

1 Assume

$$Q = \{1, \ldots, \underbrace{|Q|}_{\text{confusing}}\}$$

$$= \{1, \ldots, q\}$$

q is some integer, $q \in \mathbb{N}$

$q \geq 2$ since $q_{acc} \neq q_{rej}$

$q \geq 3$ since no harm

in making q_0, q_{acc}, q_{rej}

all different say $q_0 \to 1$, $q_{acc} \to 2$, $q_{rej} \to 3$

$q \geq 3$

If so:

instead $Q, \{q_0, q_{acc}, q_{rej}$

you just give $q \in \mathbb{N}$

think of q in base 1 ← sometimes avoid base 1

base 2

base 10

base 16

Now $Q, \{q_0, q_{acc}, q_{rej}$

just some base 2 integer

word over $\{0, 1\}$

We need to describe

$$\Sigma = \{1, 2, \ldots, s\}$$

$$\Gamma = \{1, 2, \ldots, r\}$$

"standardization"

We have $Q, q_0, q_{acc}, q_{rej}, \Sigma, \Gamma$

as

$$\underbrace{1011}_{q} \# \underbrace{11}_{s} \# \underbrace{101}_{r} \#$$ now describe δ

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

state q $\{1, \ldots, q\}$

$\{1, \ldots, r\}$

L or R

So ...

We need to specify

$q \cdot \gamma$ values

$\delta(1,1)$, $\delta(1,2)$, ...

So let's say we fix an alphabet:

$\{0, 1, L, R, \#\}$

to describe all TM algorithms

" " " standardized TM's

$\underbrace{1011}_{q} \# \underbrace{11}_{s} \# \underbrace{101}_{\gamma} \#$ now describe δ

11 states

input alphabet size 3

tape alphabet size 5

(can't be size 1, 2, 3)

11 states 5 symbols

$$Q \times \Gamma \to Q \times \Gamma \times \{L, R\}$$

Assume blank symbol is $s+1$, here 4

we give, here

55 values,

each value $\in Q \times \Gamma \times \{L, R\}$

in $\mathbb{N}$ in $\mathbb{N}$ in $\{L, R\}$

$\delta(1,1) = (7, 4, L)$

111 # 100 # L #

$\delta(1,2)$ – # " # " #

— — ..

input to this machine:

string over

$\Sigma = \{1, \ldots, s\}$

here $s = 3$, just

string over $\{1, 2, 3\}$

input 1 2 3 2 1 1 2 3 3

1 # 10 # 11 # 10 ...

---

break at 10:20 – 10:25 am

We have alphabet

$$\{0, 1, \#, L, R\}$$

or just $\{0, 1, \#\}$

convention $L \leftrightarrow 0$

$R \leftrightarrow 1$

$$111 \ \# \ 100 \ \# \ L \ \#$$

- - 1

$\downarrow$

0

A universal TM :

input alphabet

$$\boxed{\Sigma_{\text{to describe TM, inputs}} = \{0, 1, \#, L, R\}}$$

task : given

$\langle M, I \rangle$ ← a description of a TM and an input to the TM

M: TM

I: input

$\text{EncodeProg}(p) = \langle p \rangle =$ description of a program

My notes

[Sip]

A universal TM is, by definition, any "algorithm" (eventually a standardized TM) that takes

$\langle M, i \rangle$

M is a standardizes TM

i is an input to M

Example

1011 # 11 # 101 #

111 # 100 # L #

⋮

1001 # 11 # R ## 1 # 10 # 11 # 10 # 11 # 1

end of TM

end of input

Let's build a universal TM!

q_0

Q

$\shortparallel$

$\{1, \ldots, q\}$

tape 1

1 0 1 1 # ... # 1 1 # 1 ⊔ ⊔ ⊔

number for 1 to q: what state

tape 2

⊔ ⊔

which tape symbol are we reading?

tape 3

tape 4

copy the TM description

delta funct desc

tape 5

copy the input

original input

which entry of δ are we currently reading

tape 6

tape 7

Univ TM, U, only needs to:

- Do exactly what M would do on input i step by step

- Halt in accept state of U if M halts on i and accepts

- " " reject " " U

  " " " " " rejects

- Keeps going if M does not halt on i

Class over