CPSC 421/501 Nov 18, 2021

Today:

- Universal TM's
- Delightful TM's
- ACCEPTANCE_{TM}, HALTING_{TM}, ... undecidable
- "Abstractify"
  - Program-Input Systems

    Results: $\mathcal{P} \times \mathcal{I} \to \{\text{yes}, \text{no}, \text{loops}\}$

  - Enhanced P-I Systems
- Oracle TM's ← Defined Ch 6 ← Ch 9

$\Sigma_{Wow!} = \{0, 1, \#, L, R\}$

here for convenience

There's a map:

$$\underbrace{\text{Standardized TM}}_{\text{"Reasonable description of a TM"}} \longrightarrow \left(\Sigma_{Wow!}\right)^*$$

Standardized TM:

blank

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

subject to:

(1) $Q = \{1, \dots, q\}$ some $q \in \mathbb{N}$,

$\boxed{(2)\ \Sigma = \{1, \dots, s\} \text{ some } s \in \mathbb{N},}$

(3) $\Gamma = \{1, \dots, s, s+1, \dots, \gamma\}$

blank

$\gamma \in \mathbb{N}$

(4)

(5) $q_{acc} = 2$, $q_{rej} = 3$, $q_0 = 1$

and $q_0 \neq q_{acc}, q_{rej}$

$[\gamma \geq s+1, \quad q \geq 3, \dots]$

Def A *standardized TM* is one that satisfies the above,

a *standardized alphabet* is

a $\Sigma = \{1, \ldots, s\}$ for some $s \in \mathbb{N}$.

=

So $\Sigma = \{a, b\}$ is not standardized

can "we solve"

$$\text{PALINDROME}_{\Sigma = \{a, b\}} \; ?$$

Take bijection

$$\{a, b\} \longrightarrow \{1, 2\}$$

ask same question of $\text{PALINDROME}_{\{1, 2\}}$

Remark: Any TM over $\Sigma = \{1, \ldots, q\}$

has an "equivalent TM"

that is standardized:

i.e.

Result(some TM, input)

should be {yes, no, loops}

$\mathcal{P}$ = set of programs some subset of $\left(\Sigma_{\text{wow!}}\right)^*$

$\mathcal{I}$ = set of inputs " "

$$\Sigma_{\text{wow!}} = \{0, 1, \#, L, R\}$$

Abstractly, an algorithm is

Result : $P \times \mathcal{I} \longrightarrow \{$ yes, no, loops $\}$

compare

Prof $\times$ IceCream $\rightarrow \left\{ \begin{array}{l} \text{yes} = \text{like}, \\ \text{no} = \text{doesn't like} \end{array} \right\}$

e.g.

TM last class!

$\langle$ q in binary $\rangle$ # $\langle$ s in binary $\rangle$ #

$\langle$ r in binary $\rangle$ # $\underbrace{\delta(1,1)}$

$\delta(1,2)$

$\vdots$

TM & input:

⟨q in binary⟩ # ⟨s in binary⟩ #

⟨r in binary⟩ # $\underbrace{\delta(1,1)}$

$\delta(1,2)$

$\vdots$

$\delta(q,r)$

1st symbol/letter of input

2nd " "

$\vdots$

last symbol/letter of input

So this gives

$$\text{EncodProg}: \mathcal{P} \longrightarrow \Sigma_{\text{wow!}}^{*}$$

TM

EncodeBoth Prog And Input

$$\mathcal{P} \times \mathcal{I} \longrightarrow \Sigma_{\text{wow!}}^{*}$$

$$\mathcal{I} \text{ subset of } \{0, 1, \#\}^{*}$$

For simplicity

$$\mathcal{I} = \Sigma_{\text{wow!}}^{*}$$

if input is not what we expect, say not a valid input

=

Back to TM's:

$$\text{ACCEPTANCE}_{TM} = A_{TM} = \stackrel{\text{def}}{=} \{ \langle p, i \rangle \mid \text{Result}(p, i) = \text{accept} \}$$

$$[S_{ip}] = \{ \langle p, i \rangle \mid p(\langle i \rangle) = \text{accept} \}$$

$$= \{ \text{EncodeBoth}(p, i) \mid \text{Result}(p, i) = \text{~~accept~~ yes} \}$$

Helps me to think about what is going on

Thm: $A_{TM} = \text{ACCEPTANCE}_{TM}$

is undecidable (in the context of TM's).

Thm: HALT_{TM}

$$= \{ \langle p, i \rangle \mid p(\langle i \rangle) = \overset{\text{accept}}{\text{yes}} \text{ or } \overset{\text{reject}}{\text{no}} \}$$

$$= \{ \text{Encoding}(p, i) \mid \text{Result}(p, i) = \text{yes or no} \}$$

$=$

Proof:

Break 10:15 – 10:20

Define: For each TM, M, over Σ

(1) $LANGUAGE_{\Sigma}(M)$

$= \{ i \in \Sigma^* \mid M \text{ accepts } i \}$

we say M recognizes

(2) We say M "halts on i"

if – – –

M accepts i, or

M rejects i, but

M doesn't loop on i

(3) We say M is a <u>decider</u>

if on any input, i, M

either rejects or accepts, i.e. M

halts.

I.e.

$$\text{Result}\left(\underset{\text{TM}}{\text{a}}, \underset{\text{input}}{\text{an}}\right) \in \{\text{yes}, \text{no}, \text{loops}\}$$

"to decide", M is decider, $\{\text{accept}, \text{reject}, \text{loops}\}$

means

$$\text{Result} : \underset{(\text{TM})}{\text{Program}} \times \text{Input} \to \{\text{yes}, \text{no}, \text{loops}\}$$

also a map $\to \{\text{yes}, \text{no}\}$

(as a subset of)

i.e. reach accept/yes

reject/no

after finitely many steps

Example:

Define U to be a universal TM

if

$\text{LANGUAGE}_{\text{standardized TM}}(U)$

$= \text{ACCEPTANCE}_{\text{standard TM}}$

Really a universal TM is an enhanced debugging tool

very mild simplification!

Universal TM as simulates,

yes → yes
no → no
loops → loops

build a U as above

$\langle p, i \rangle \xrightarrow{\text{universal TM}}$ if

accept $\to$ accept

reject $\to$ reject

loops $\to$ loop

put in either reject or loops

as in [Sip] §4.2

=

<u>**Def**</u>: A TM (program), D, is <u>delightful</u> if

$$\boxed{\forall M \text{ TM} \quad \text{Result}(D, \langle M \rangle) = \neg \text{Result}(M, \langle M \rangle)}$$

[Sip] $\quad D(\langle M \rangle) = \neg M(\langle M \rangle)$

OR

$\text{Result}(D, \text{EncodeProg}(M))$

$= \neg\,\text{Result}(M, \text{EncodeProg}(M))$

$\boxed{\begin{array}{l}\textbf{Thm: } \text{If } D \text{ is \underline{delightful},} \\ \text{then} \\ (1)\ \text{Result}(D, \text{EncodeProg}(D)) \\ \quad = \neg\,\text{Result}(D, \text{EncodeProg}(D)) \\ (2)\ \text{Hence } \text{Result}(D, \text{EncodeProg}(D)) = \text{loops}\end{array}}$

Cor: Hence there is no U TM that decides ACCEPTANCE, ie.

ACCEPTANCE is <u>undecidable</u>.