Lecture # 9 Scribble

Topics to cover:
- Cantor's diagonalization
- Halting Theorem (2)
- TM reduceability

Acceptable vs. Decidable

- A language L is acceptable if some TM accepts L and unacceptable otherwise
  semi-computable, semi decidable
  Turing-recognizable, listable
  recursively enumerable

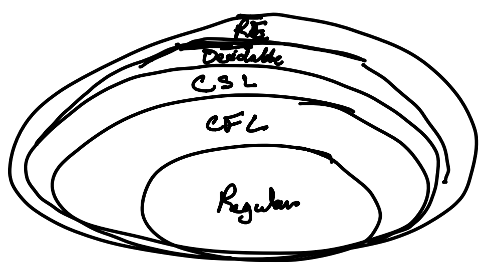
- L is decidable if some TM decides L
  recursive or computable
  always accepts/rejects
  never loops

Central Problem

Want to know if a arbitrary TM will halt on an arbitrary input

$A_{TM} = \{ \langle M, w \rangle \mid M \text{ is a TM and } M \text{ accepts } w \}$

Is A_{TM} decidable?



Infinite

- Set $\mathbb{N}$ of natural number $\{1, 2, 3, \ldots\}$
- Set $\mathbb{E}$ of even numbers $\{0, 2, 4, 6, \ldots\}$
- Set $\mathbb{R}$ of real numbers $[0, 1)$

Cantor's proposition: two sets have the same size if their elements can be paired in a 1-1 correspondence
bijection

$\mathbb{N} \to \mathbb{E} \quad e = 2n$ where $n \in \mathbb{N}$ & $e \in \mathbb{E}$

Cantor's diagnolization argument

if a bijection between $\mathbb{N}$ & $\mathbb{R}$ existed

$\mathbb{N}$ \ T	x_0	x_1	x_2	x_3	...	x_m	...
0	.5	0	0	0	...	0	...
1	.2	7	8	5	...	4	...
2	.3	1	2	0	...	8	...
⋮	⋮	⋮	⋮	⋮		⋮	
n	.1	1	2	9	...	7	...
⋮	⋮	⋮	⋮	⋮		⋮	

$z = .6\,8\,2 \ldots\ldots 8 \ldots\ldots$ $n+1$

z cannot exist in T

enumerability

a set is enumerable iff there exists a bijection between it and $\mathbb{N}$

S of all possible infinite binary strings

$\mathbb{N}$								
0	1	1	0	1	1	0	...	
1	0	1	1	1	1	1	...	
2	1	1	0	0	0	0	...	
⋮								
n	1	0	0	0	0	1	0	...
⋮								

$z = 0\,0\,1 \ldots 0 \ldots$

There are infinitely many languages not all languages are enumerable

The set of Turing Machines is RE/countable.

All TMs can be encoded in binary strings (finite)

$|\Sigma| = m$ encoding of the TM is k digits long

L(TM encodings) = finite # of TMs encoded in k digits

$\_\ \_\ \_\ \_\ \_ = |\Sigma|^k$

$|\Sigma| = 2$

$\mathbb{N} = 1, 2, 3 \ldots$

$k=1$ | 0 1

$k=2$ | 2 3 4 5

$k=3$ | 6 7 8 9 10 11 12 13

# of TMs < # of languages

$A_{TM} = \{ \langle M, w \rangle \mid M \text{ is a TM and } M \text{ accepts } w \}$

Is A_{TM} decidable?

$$H(\langle M, w \rangle) = \begin{cases} \text{accepts} & \text{if } M \text{ accepts } w \\ \text{rejects} & \text{if } M \text{ rejects } w \end{cases}$$

D where on input $\langle M \rangle$:

1. Run H on $\langle M, \langle M \rangle \rangle$ ← Does M accept itself
2. Output opposite of H ← Complement

$$D(\langle M \rangle) = \begin{cases} \text{accept} & \text{if } H(\langle M, \langle M \rangle \rangle) \text{ rejects} \\ \text{reject} & \text{if } H(\langle M, \langle M \rangle \rangle) \text{ accepts} \end{cases}$$

$$D(\langle D \rangle) = \begin{cases} \text{accept} & \text{if } H(\langle D, \langle D \rangle \rangle) \text{ rejects} & \text{if } D \text{ does not accept } D \\ \text{reject} & \text{if } H(\langle D, \langle D \rangle \rangle) \text{ accepts} & \text{if } D \text{ does accept } D \end{cases}$$

Diagonalization Arg

machines	$\langle M_1 \rangle$	$\langle M_2 \rangle$	. . .	$\langle M_k \rangle$	. . . ← string inputs
M_1	acc	rej		DH (doesn't halt)	
M_2	rej	acc		acc	
⋮					
M_k	DH	rej		DH	
⋮					

Create T_H which decides $T(A_{TM})$

machines ↓ / ← string inputs

H	$\langle M_1 \rangle$	$\langle M_2 \rangle$	...	$\langle M_k \rangle$	...
M_1	acc	rej		rej	
M_2	rej	acc		acc	
⋮					
M_k	rej	rej		rej	
⋮					

Create V_D which is what happens if we pass the diagonal of T_H into D

$$V_D = [\text{rej}, \text{rej}, \ldots \text{acc} \ldots]$$

machines ↓ / ← string inputs

	$\langle M_1 \rangle$	$\langle M_2 \rangle$	...	$\langle M_k \rangle$	...
M_1	acc	rej		DH (doesn't halt)	
M_2	rej	acc		acc	
⋮					
M_k	DH	rej		DH	
⋮					

Reducibility:

$E_{TM} = \{\langle M \rangle \mid M \text{ is a TM and } L(M) = \emptyset\}$

Is E_{TM} decidable:

R is decider of E_{TM}

R accepts if $L(M)$ is empty, rejects otherwise

Show that a TM S can be constructed with R that decides A_{TM}

M_1 = On input x : If $x \neq w$ reject
$x = w$ run M on w and if M accepts, accept

S = On input $\langle M, w \rangle$

1. Construct M_1 ← accepts if M accepts w

   if M accept w. then $L(M_1) \neq \emptyset$

2. Run R on input $\langle M_1 \rangle$

3. If R accepts, reject, else accept