COMS W3261 — Lecture 10, Part 2

Idea: Some problems can't be solved by computers (!)

Consider: $A_{TM} = \{\underbrace{\langle M, w \rangle}_{\text{encoded TM}} \mid M \text{ is a TM and } M \text{ accepts } w\}$

Easy to recognize: just simulate M on w, accept if M accepts.

(But what if M loops infinitely on w? How can we stop our simulation?)

Recall: Long ago, we proved that $\mathbb{R}$ can't be listed out as a sequence. We did this by diagonalization:

Suppose for contradiction some sequence exists with all $\mathbb{R}$:

$a_1 = \cancel{0}.00012$

$a_2 = 0.\cancel{1}3962$

$a_3 = 1.3\cancel{9}821$

$a_4 = 0.11\cancel{1}22$

$\vdots$

Given such a sequence, we can build a number not in it:

$n = 1.202\ldots$

By definition, $n \neq a_j$ for any j!

Similar idea here.

Definition. A set is countable if it is finite or if there exists a one-to-one mapping to the natural numbers $\mathbb{N} = 1, 2, 3, \ldots$.

Example: $\mathbb{Z}$ is countable.

n	$f(n)$
1	0
2	-1
3	1
4	-2
5	2

$2\mathbb{N}$ is countable.

n	$f(n)$
1	2
2	4
3	6
4	8

Formally, we proved that $\mathbb{R}$ is <u>uncountable.</u>

(Exercise: is $\mathbb{Q}$ countable? $\mathbb{Q} := \frac{m}{n}, \; m, n \in \mathbb{Z}$).

<u>Observation.</u> The set of all TMs is countable.

<u>Proof.</u> TMs are finite objects by definition. Consider some encoding $\langle \cdot \rangle$ that maps TMs to strings over the alphabet Σ.

Σ^* is countable (to see this, write down a mapping that counts all strings of length 0, then length 1, then length 2, ...).

For every TM M, let $f(M) = k$ if M is the k^{th} Turing Machine written down in our enumeration of Σ^*. ∎

(Example. $\Sigma = \{0, 1\}$.

Enumeration of Σ^*:

$$\varepsilon, \; 0, \; 1, \; 00, \; 01, \; 10, \; 11, \; \ldots, \; s_4, \; \underset{=\langle M \rangle}{s_5}, \; s_{1000}, \ldots$$

If $\langle M \rangle$ is an encoding of M into $\{0,1\}$, it appears in this sequence.
All TMs appear somewhere in the sequence, which creates an ordering.)

<u>Observation.</u> The set of infinite binary strings is uncountable.

<u>Proof.</u> By diagonalization.

$a_1 = \cancel{0}1100110\ldots$

$a_2 = 1\cancel{1}00101\ldots$

$a_3 = 00\cancel{1}0100\ldots$

$\vdots$

$b = 100\ldots$

$b \neq a_j$ for any j.

<u>Theorem.</u> <u>Some languages are not Turing recognizable.</u>

<u>Proof.</u> We can encode any language as an infinite binary string indicating which elements of Σ^* are members.

$$\Sigma^* = \{ \varepsilon, \; 0, \; 1, \; 00, \; 01, \; 10, \; 11, \; \ldots$$

$$L \subseteq \Sigma^* = \; 0 \;\; 1 \;\; 1 \;\; 1 \;\; 0 \;\; 0 \;\; 0 \;\; \ldots$$

(this means $L = \{0, 1, 00, \cdots\}$)

We have a 1-to-1 correspondence between languages and infinite binary strings. By our second observation, the set of languages is uncountable.

By our first observation, the set of TMs is countable, so there can be no one-to-one mapping between TMs and languages. In particular, this rules out the possibility that every language is recognized by some TM.

(Alternative proof: could imagine writing down all the TMs and creating a language of strings such that the n^{th} TM doesn't correctly behave on the n^{th} string.)

3. Undecidable & Unrecognizable Languages.

Paradoxes.

Liar's paradox: "This statement is false." → true? no. → false? no.

Russell's paradox:

"Consider the set S of all sets that don't contain themselves. Is S a member of itself?"

$$S \in S \Rightarrow \times$$
$$S \notin S \Rightarrow \times$$

Can conclude that any foundation for set theory that lets you create a set "like S" is "problematic."

(= Barber's paradox: B shaves everybody who doesn't shave themself. Who shaves B?)

We'll show that $A_{TM} = \{\langle M, w\rangle \mid M \text{ is a TM and } M \text{ accepts } w\}$ is undecidable by showing if it were, it would create a paradox.

Proof. Assume $A_{TM} = \{\langle M, w\rangle \mid M \text{ is a TM}, M \text{ accepts } w\}$ is decidable, and let H be a decider for A_{TM}.

$$H(\langle M, w\rangle) = \begin{cases} \text{accept if } M(w) \text{ accepts} \\ \text{reject if } M \text{ doesn't accept } w \text{ (rejects or loops)} \end{cases}$$

Given H, we can build a new TM, D:

$D =$ " On input $\langle M\rangle$, where M is a TM:

1. Run H on $\langle M, \langle M\rangle\rangle$.
2. Output the opposite of what H outputs."

$\langle M\rangle$ is a string. H finds out whether M accepts on its own description.

Thus
$$D(\langle M\rangle) = \begin{cases} \text{accept if } M \text{ does not accept } \langle M\rangle \\ \text{reject if } M \text{ accepts } \langle M\rangle. \end{cases}$$

So — what does $D(\langle D\rangle)$?

$$D(\langle D\rangle) = \begin{cases} \text{accept if } D(\langle D\rangle) \text{ rejects} \\ \text{reject if } D(\langle D\rangle) \text{ accepts.} \end{cases}$$

contradiction
paradox.

Thus H cannot exist, so A_{TM} is undecidable.

So: A_{TM} is not decidable. It is recognizable. Now we can also show a specific unrecognizable language.

Def. (co-Turing-recognizable.) A language is co-Turing-recognizable if its complement is Turing-recognizable.

(Ex. A_{TM} is recognizable, $\overline{A_{TM}}$ is co-Turing-recognizable.)

Thm. A language is decidable if and only if it is Turing-recognizable and co-Turing recognizable.

Proof. Let A be a decidable language. Then it has a decider, and $\overline{A_{TM}}$ can be recognized by a TM that does the opposite of the decider.

Now: if a language is recognizable and co-Turing recognizable, decide by running both recognizers in parallel. Accept if the recognizer for A accepts, reject if the recognizer for $\overline{A}$ accepts.

go back and forth between two simulations, run each for a few steps.

<u>Thm.</u> $\overline{A_{TM}}$ is not Turing-recognizable.

Why? Well, A_{TM} is recognizable. If $\overline{A_{TM}}$ was also recognizable, then A_{TM} would be decidable by the previous theorem. A_{TM} is not decidable, so $\overline{A_{TM}}$ must not be Turing-recognizable.

<u>Next time:</u> (Donuts / bagels!)

Time complexity.

Reductions.