EECS 2001A : Introduction to the Theory of Computation

Suprakash Datta

Course page: http://www.eecs.yorku.ca/course/2001
Also on Moodle

Countably Infinite Languages

- Let $\Sigma = \{0\}$. Then Σ^* is countable
  $f : \mathbb{N} \to \Sigma^*$, $f(i) = a^{i-1}$

- Let Σ be a finite alphabet. Then Σ^* is countable
  Idea: We list Σ^* in increasing order of length and for strings of the same length we list them in lexicographic order
  E.g.: $\{0, 1\} = \{\epsilon, 0, 1, 00, 01, 10, 11, 000, \ldots\}$
  Then each finite length string gets a unique finite label

- IMPORTANT: Set of all Turing machines T is countable:
  Idea: Every TM can be encoded as a string over some Σ. There is a surjective map from Σ^* to T.

Countably Infinite Languages - 2

- We just argued that the set of all Turing machines T is countable
- What about the set of all languages (problems)?
  We have argued before that this set is $\mathcal{P}(\Sigma^*)$
- We will show next that $\mathcal{P}(\Sigma^*)$ and some other sets (e.g., $\mathbb{R}, \mathcal{P}(\mathbb{N})$) are not countable!

$\mathcal{P}(\Sigma^*)$ is not Countable

Claim: There is no surjection $f : \mathbb{N} \to \mathcal{P}(\Sigma^*)$
Proof by contradiction. Assume there is a surjection f.

- $f(1), f(2), \ldots$ are all infinite bit strings in $\{0, 1\}^{\mathbb{N}}$
- Define the infinite string $y = y_1 y_2 \ldots$ by
  $y_j = \mathrm{NOT}(\text{j-th bit of } f(j))$
- On the one hand $y \in \{0, 1\}^{\mathbb{N}}$, but on the other hand: for every $j \in \mathbb{N}$ we know that $f(j) \neq y$ because $f(j)$ and y differ in the j-th bit
- f cannot be a surjection: $\{0, 1\}^{\mathbb{N}}$ is uncountable.

Diagonalization

s_1	=	0	0	0	0	0	0	0	0	0	0	0	...
s_2	=	1	1	1	1	1	1	1	1	1	1	1	...
s_3	=	0	1	0	1	0	1	0	1	0	1	0	...
s_4	=	1	0	1	0	1	0	1	0	1	0	1	...
s_5	=	1	1	0	1	0	1	1	0	1	0	1	...
s_6	=	0	0	1	1	0	1	1	0	1	1	0	...
s_7	=	1	0	0	0	1	0	0	0	1	0	0	...
s_8	=	0	0	1	1	0	0	1	1	0	0	1	...
s_9	=	1	1	0	0	1	1	0	0	1	1	0	...
s_{10}	=	1	1	0	1	1	1	0	0	1	0	1	...
s_{11}	=	1	1	0	1	0	1	0	0	1	0	0	...
$\vdots$		$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$

s	=	1	0	1	1	1	0	1	0	0	1	1	...

- Look at the bit string on the diagonal of this table: $s_d = 0100...$
- The negation of s_d, given by $s = 1011\ldots$, does not appear in the table

Diagonalization: Recap

- We looked at a very innovative technique for proving that a set S is uncountable
- It is a proof by contradiction and starts off by assuming S is countable
- The argument does not (and should not) assume any specific ordering of the set S
- Rather it says: "Give me any enumeration/listing (or labeling with $\mathbb{N}$, or bijection with $\mathbb{N}$), and I will construct an element that is not listed/enumerated/labeled..., and that is a contradiction"

More Diagonalization: $\mathcal{P}(\mathbb{N})$ is not countable

- The set $\mathcal{P}(\mathbb{N})$ contains all the subsets of $\{1, 2, \ldots\}$
- Each subset $X \subseteq \mathbb{N}$ can be identified by an infinite string of bits $x_1 x_2 \ldots$ such that $x_j = 1$ iff $j \in X$
- There is a bijection between $\mathcal{P}(\mathbb{N})$ and $\{0, 1\}^{\mathbb{N}}$ - each bit string represents a unique subset of $\mathbb{N}$ and each subset of $\mathbb{N}$ corresponds to a unique bit string
- We could stop here and invoke the last slide, but let us rework the proof in the last slide
- Proof by contradiction: Assume $\mathcal{P}(\mathbb{N})$ countable. Hence there must exist a surjection f from $\mathbb{N}$ to the set of infinite bit strings $\{0, 1\}^{\mathbb{N}}$, or
  "There is a list of all infinite bit strings"
- Make the exact same diagonalization argument

More Diagonalization: $\mathbb{R}$ is not countable

- Will use diagonalization to prove $R' = [0, 1)$ is uncountable
- Let f be a function $\mathbb{N} \to R'$. So $f(1), f(2), \ldots$ are all infinite digit strings (padded with zeroes if required), and let $f(i)_j$ be the j-th bit of $f(i)$
- Define the infinite string of digits $y = y_1 y_2 \ldots$ by

$$\begin{aligned} y_j &= f(i)_i + 1 \text{ if } f(i)_i < 8 \\ &= 7 \text{ if } f(i)_i \geq 8 \end{aligned}$$

- Invoke diagonalization to get a contradiction
- So $R' \subset \mathbb{R}$ is not countable, and therefore $\mathbb{R}$ is not countable

Other Questions on Infinite Sets

- The set $\mathbb{N}$ is countable by definition. So a proof showing it is uncountable (using diagonalization) must fail. But where does it fail?
- We showed that $\mathcal{P}(\mathbb{N})$ (and $\mathbb{R}$) are uncountable. What about $\mathcal{P}(\mathbb{R})$?
- What about $\mathcal{P}(\mathcal{P}(\mathbb{R}))$?
- Can we build bigger and bigger infinities this way?
  Cantor's Continuum hypothesis: YES!

Back to TM's and Languages

- We showed that the set of languages is not countable
- We showed that the set of TM's is countable
- So there are many languages that are not Turing recognizable
- Are there interesting languages for which we can prove that there is no Turing machine that recognizes it?

Our First Undecidable Language

The acceptance problem for Turing Machines:
$A_{TM} = \{\langle M, w\rangle | M \text{ is a TM that accepts } w\}$
Theorem: A_{TM} is undecidable

- Proof by contradiction: Assume that TM G decides A_{TM}
- So G is as follows

$$\begin{aligned} G(\langle M, w\rangle) &= \text{“accept” if } M \text{ accepts } w \\ &= \text{“reject” if } M \text{ does not accept } w \end{aligned}$$

- From G we construct a new TM D that will get us into trouble...

Our First Undecidable Language - 2

Design a new TM D that takes as input a TM M as follows

- D runs TM G on input $\langle M, \langle M \rangle\rangle$
- Disagree on the answer of G
- Note that D always terminates because G always terminates
- So in short,

$$\begin{aligned} D(\langle M \rangle) &= \text{“accept” if } G \text{ rejects } \langle M, \langle M \rangle\rangle \\ &= \text{“reject” if if } G \text{ accepts } \langle M, \langle M \rangle\rangle \end{aligned}$$

- So,

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

Our First Undecidable Language - 3

- Recall,

$$\begin{aligned} D(\langle M \rangle) &= \text{“accept” if } M \text{ rejects } \langle M \rangle \\ &= \text{“reject” if if } M \text{ accepts } \langle M \rangle \end{aligned}$$

- Now run D on itself (i.e., $\langle D \rangle$)
- Result:,

$$\begin{aligned} D(\langle D \rangle) &= \text{“accept” if } D \text{ rejects } \langle D \rangle \\ &= \text{“reject” if if } D \text{ accepts } \langle D \rangle \end{aligned}$$

- This makes no sense: D only accepts if it rejects, and vice versa
- This is a contradiction, therefore A_{TM} is undecidable

Viewing the Last Proof as Diagonalization

	$\langle M_1 \rangle$	$\langle M_2 \rangle$	$\langle M_3 \rangle$	$\langle M_4 \rangle$	$\cdots$	$\langle D \rangle$	$\cdots$
M_1	accept	reject	accept	reject			
M_2	accept	accept	accept	accept			
M_3	reject	reject	reject	reject	$\ldots$		
M_4	accept	accept	reject	reject			
$\vdots$			$\vdots$		$\ddots$		
D	reject	reject	accept	accept	$\cdots$	?	
$\vdots$							

- This is an instance of self-referencing by a program
- This is sometimes natural - a character counting program can run on itself

Self-referencing Problems

- Some such problems are decidable
  - How big is $\langle M \rangle$?
  - Is $\langle M \rangle$ a proper TM?
- Others are not
  - Does $\langle M \rangle$ halt or not?
  - Is there a smaller program M' that is equivalent?

Turing Unrecognizability

- A_{TM} is not TM-decidable, but it is TM-recognizable. Wy?
- Is there a language that is not TM-recognizable?
- A useful result:
  Theorem: If a language A is TM-recognizable and its complement $\overline{A}$ is recognizable, then A is TM-decidable.
- Proof: Run the recognizing TMs for A and in parallel on input x. Wait for one of the TMs to accept. If the TM for A accepted: "accept x"; if the TM for $\overline{A}$ accepted: "reject x"

$\overline{A}_{TM}$ is not TM-recognizable

- By the previous theorem it follows that $\overline{A}_{TM}$ cannot be TM-recognizable, because this would imply that A_{TM} is TM decidable

- We call languages like $\overline{A}_{TM}$ co-TM recognizable

Other Languages that are not TM-recognizable

- $E_{TM} = \{\langle G \rangle | G \text{ is a TM with } L(G) = \emptyset\}$
  - This is co-TM recognizable
    Obvious strategy: if the language is non-empty, we can find the first string that is accepted ...
  - Is it TM-recognizable (and thus decidable)?
    Answer turns out to be NO
- $EQ_{TM} = \{\langle G, H \rangle | G, H \text{ are TM's with } L(G) = L(H)\}$
  - Is this co-TM recognizable?
  - Is it TM-recognizable?
  - Turns out both answers are NO

We need more tools to reason about these languages