Examples of basic operations

- Convince yourself that the following types of operations are easy to implement as part of TM "program" (but perhaps tedious to write out...)
  - copying
  - moving
  - incrementing/decrementing
  - arithmetic operations +, -, *, /

Universal TMs and encoding

- the input to a TM is always a string in $\Sigma^*$
- often we want to interpret the input as representing another object
- examples:
  - tuple of strings (x, y, z)
  - 0/1 matrix
  - graph in adjacency-list format
  - Context-Free Grammar

Universal TMs and encoding

- some strings not valid encodings and these are not in the language

- We can easily construct a Universal TM that recognizes the language:
  $A_{TM} = \{<M, w> : M \text{ is a TM and } M \text{ accepts } w\}$
  - how?
- this is a remarkable feature of TMs (not possessed by FA or NPDAs...)
- means there is a general purpose TM whose input can be a "program" to run
Church-Turing Thesis

- many other models of computation
  - we saw multitape TM, nondeterministic TM
  - others don’t resemble TM at all
- common features:
  - unrestricted access to unlimited memory
  - finite amount of work in a single step
- every single one can be simulated by TM
- many are equivalent to a TM
- problems that can be solved by computer does not depend on details of model!

Church-Turing Thesis

- the belief that TMs formalize our intuitive notion of an algorithm is:
  
  The Church-Turing Thesis
  
  everything we can compute on a physical computer can be computed on a Turing Machine

- Note: this is a belief, not a theorem.

Recursive Enumerability

- Why is “Turing-recognizable” called RE?
- Definition: a language $L \subseteq \Sigma^*$ is recursively enumerable if there is exists a TM (an “enumerator”) that writes on its output tape

  $\#x_1\#x_2\#x_3\#\ldots$

  and $L = \{x_1, x_2, x_3, \ldots\}$.

- The output may be infinite

Recursive Enumerability

**Theorem:** A language is Turing-recognizable iff some enumerator enumerates it.

Proof:

$(\Rightarrow)$ Let $E$ be the enumerator. On input $w$:
- Simulate $E$. Compare each string it outputs with $w$.
- If $w$ matches a string output by $E$, accept.

Recursive Enumerability

**Theorem:** A language is Turing-recognizable iff some enumerator enumerates it.

Proof:

$(\Leftarrow)$ Let $M$ recognize language $L \subseteq \Sigma^*$.
- let $s_1, s_2, s_3, \ldots$ be enumeration of $\Sigma^*$ in lexicographic order.
- for $i = 1,2,3,\ldots$
    - simulate $M$ for $i$ steps on $s_1, s_2, s_3, \ldots, s_i$
    - if any simulation accepts, print out that $s_j$

Undecidability

- decidable
- RE
- all languages
- decidable $\subseteq$ RE $\subseteq$ all languages
- our goal: prove these containments proper
Countable and Uncountable Sets

- the natural numbers \( \mathbb{N} = \{1,2,3,\ldots\} \) are countable.

- Definition: a set \( S \) is countable if it is finite, or it is infinite and there is a bijection \( f: \mathbb{N} \to S \).

Theorem: the positive rational numbers \( \mathbb{Q} = \{m/n : m, n \in \mathbb{N}\} \) are countable.

Proof:

\[
\begin{array}{cccccccccccc}
1/1 & 1/2 & 1/3 & 1/4 & 1/5 & 1/6 & \ldots \\
2/1 & 2/2 & 2/3 & 2/4 & 2/5 & 2/6 & \ldots \\
3/1 & 3/2 & 3/3 & 3/4 & 3/5 & 3/6 & \ldots \\
4/1 & 4/2 & 4/3 & 4/4 & 4/5 & 4/6 & \ldots \\
5/1 & \ldots
\end{array}
\]

Theorem: the real numbers \( \mathbb{R} \) are NOT countable (they are “uncountable”).

How do you prove such a statement?
- assume countable (so there exists bijection \( f \))
- derive contradiction (some element not mapped to by \( f \))
- technique is called diagonalization (Cantor)

Proof:
- suppose \( \mathbb{R} \) is countable
- list \( \mathbb{R} \) according to the bijection \( f \):

<table>
<thead>
<tr>
<th>( n )</th>
<th>( f(n) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3.14159...</td>
</tr>
<tr>
<td>2</td>
<td>5.55555...</td>
</tr>
<tr>
<td>3</td>
<td>0.12345...</td>
</tr>
<tr>
<td>4</td>
<td>0.50000...</td>
</tr>
</tbody>
</table>

\( x = 0.a_1a_2a_3a_4... \)

set \( x = 0.a_1a_2a_3a_4... \) where digit \( a_i \neq i^{th} \) digit after decimal point of \( f(i) \) (not 0, 9)

e.g. \( x = 0.2312... \)

\( x \) cannot be in the list!

non-RE languages

Theorem: there exist languages that are not Recursively Enumerable.

Proof outline:
- the set of all TMs is countable
- the set of all languages is uncountable
- the function \( L: \{\text{TMs}\} \to \{\text{languages}\} \) cannot be onto

\[ \ldots \]
non-RE languages

- Lemma: the set of all TMs is countable.
- Proof:
  - each TM $M$ can be described by a finite-length string $<M>$
  - can enumerate these strings, and give the natural bijection with $\mathbb{N}$

non-RE languages

- Lemma: the set of all languages is uncountable
- Proof:
  - fix an enumeration of all strings $s_1, s_2, s_3, \ldots$ (for example, lexicographic order)
  - a language $L$ is described by its characteristic vector $f_L$ whose $i^{th}$ element is 0 if $s_i$ is not in $L$ and 1 if $s_i$ is in $L$

So far...

- This language might be an esoteric, artificially constructed one. Do we care?
- We will show a natural undecidable $L$ next.

The Halting Problem

- Definition of the “Halting Problem”:
  $\text{HALT} = \{ <M, x> : \text{TM } M \text{ halts on input } x \}$
- $\text{HALT}$ is recursively enumerable.
  - proof?
- Is $\text{HALT}$ decidable?