We will show that this problem is not decidable.

There quite a few undecidable problems:

For example, program verification are in general not solvable.
\[ A_{TM} = \{ \langle M, w \rangle \mid M : \text{a TM that accepts } w \} \]

- We will prove that \( A_{TM} \) is undecidable
- However, \( A_{TM} \) is Turing recognizable
- We can simply simulate \( \langle M, w \rangle \)
- To be decidable we hope to avoid an infinite loop
  - if at one point, know it cannot halt
  \( \Rightarrow \) reject
- Thus this problem is called the halting problem
We need a technique called “diagonalization method” for the proof.

It was developed by Cantor in 1873 to check if two infinite sets are equal.

Example: consider

set of even integers

versus

set of \(\{0, 1\}^*\)

Which one is larger?

Definition: two sets are equal if elements can be paired.
Definition 4.12 I

- \( f \) is a one-to-one function if:

\[ f(a) \neq f(b) \text{ if } a \neq b \]

- Left: a one-to-one function; right: not
Definition 4.12 II

- \( f : A \rightarrow B \) onto if

\[ \forall b \in B, \exists a \text{ such that } f(a) = b \]

- Example:

\[ f(a) = a^2, \text{ where } A = (-\infty, \infty) \text{ and } B = (-\infty, \infty) \]

This is not an onto function because for \( b = -1 \), there is no \( a \) such that \( f(a) = b \)
However, if we change it to

\[ f(a) = a^2, \text{ where } A = (-\infty, \infty) \text{ and } B = [0, \infty) \]

it becomes an onto function

Definition: a function is called a correspondence if it is one-to-one and onto

Example:

\[ f(a) = a^3, \text{ where } A = (-\infty, \infty) \text{ and } B = (-\infty, \infty) \]
Example 4.13

- \( N = \{1, 2, \ldots\} \)
- \( E = \{2, 4, \ldots\} \)

The two sets can be paired

\[
\begin{array}{c|c}
  n & f(n) = 2n \\
  \hline
  1 & 2 \\
  2 & 4 \\
  \vdots & \vdots \\
\end{array}
\]

We consider \( N \) and \( E \) have the same size

Definition: a set is countable if it is finite or same size as \( N \)
Rational Numbers Countable

\[ Q = \{ \frac{m}{n} \mid m, n \in \mathbb{N} \} \text{ countable} \]
Note that we skip counting elements with common factors (e.g., 2/2)
We will use the diagonalization method.

The proof is by contradiction.

Assume $R$ is countable. Then there is a table as follows:

<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>55.55555</td>
</tr>
<tr>
<td>3</td>
<td>0.12345</td>
</tr>
<tr>
<td>4</td>
<td>0.50000</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
Consider

\[ x = 0.4641 \ldots \]
\[ 4 \neq 1, \; 6 \neq 5 \]

We have

\[ x \neq f(n), \forall n \]

But \( x \in R \), so a contradiction

To avoid the problem

\[ 1 = 0.9999 \ldots \]

for every digit of \( x \) we should not choose 0 or 9
Some languages not Turing-recognizable

- $\Sigma^*$ is countable
  Simply count $|w| = 1, 2, 3, \ldots$
  For example, if $\Sigma = \{0, 1\}$, then
  \[ \{\epsilon, 0, 1, 00, 01, 10, 11, \ldots \} \]
- The sets of TMs is countable
- Each machine can be represented as a finite string (think about the formal definition)
- Thus the set of TMs is a subset of $\{0, 1\}^*$
- Let
Some languages not Turing-recognizable II

$L$: all languages over $\Sigma$
$B$: all infinite binary sequences

- For any $A \in L$
  - there is a corresponding element in $B$
- Example:

  $A : 0\{0, 1\}^*$
  $\Sigma^* = \{\epsilon, 0, 1, 00, 01, 10, 11, 000, 001, \ldots\}$
  $A = \{0, 00, 01, 000, 001, \ldots\}$
  $\chi_A = 010110011\ldots$
Some languages not Turing-recognizable

- One-to-one correspondence between $B$ and $L$
- $B$ is uncountable (like real numbers)
  Therefore, $L$ is uncountable
- Each TM $\Rightarrow$ handles one language in $L$
  Set of TM is countable, but $L$ is not
- Thus some languages cannot be handled by TM