The Last Lecture

A Brief Encounter with the Uncomputable

Lecture 26
Question
Question

Which of the following are countably infinite?
1. Set of all prime numbers
2. Set of all bit strings of length 32
3. Set of all bit strings of finite length
4. Set of all infinitely long bit strings

A. 1, 2, 3 and 4
B. 1, 2 and 3
C. 1 and 4
D. 1 and 3
E. None of the above choices
The Uncountable
The Uncountable

S uncountable if no one-to-one f:S→ℕ (equiv’ly, no onto f:ℕ→S)
The Uncountable

- **S uncountable** if no one-to-one \( f:S \rightarrow \mathbb{N} \) (equiv’ly, no onto \( f: \mathbb{N} \rightarrow S \))
- **Claim:** \( \mathbb{R} \) is uncountable
The Uncountable

S uncountable if no one-to-one $f:S \rightarrow \mathbb{N}$ (equiv’ly, no onto $f: \mathbb{N} \rightarrow S$)

Claim: $\mathbb{R}$ is uncountable

Related claims:
The Uncountable

- S uncountable if no one-to-one \( f:S \rightarrow \mathbb{N} \) (equiv’ly, no onto \( f: \mathbb{N} \rightarrow S \))

- **Claim:** \( \mathbb{R} \) is uncountable

- Related claims:
  - **Set** \( T \) of all **infinitely long** binary strings is uncountable
The Uncountable

- **S** is **uncountable** if no one-to-one \( f:S \to \mathbb{N} \) (equiv’ly, no onto \( f:\mathbb{N} \to S \))

- **Claim**: \( \mathbb{R} \) is uncountable

- **Related claims**:
  - Set \( \mathbb{T} \) of all **infinitely long** binary strings is uncountable
    - Contrast with set of all finitely long binary strings, which is a countably infinite set
The Uncountable

- S **uncountable** if no one-to-one \( f: S \to \mathbb{N} \) (equiv’ly, no onto \( f: \mathbb{N} \to S \))

- **Claim**: \( \mathbb{R} \) is uncountable

- Related claims:
  - Set \( T \) of all **infinitely long** binary strings is uncountable
  - Contrast with set of all finitely long binary strings, which is a countably infinite set
  - The power-set of \( \mathbb{N} \), \( \mathcal{P}(\mathbb{N}) \) is uncountable
The Uncountable

- **S uncountable** if no one-to-one \( f: S \to \mathbb{N} \) (equiv’ly, no onto \( f: \mathbb{N} \to S \))

- **Claim**: \( \mathbb{R} \) is uncountable

- **Related claims**:
  - Set \( T \) of all **infinitely long** binary strings is uncountable
  - Contrast with set of all **finitely long** binary strings, which is a countably infinite set
  - The power-set of \( \mathbb{N} \), \( P(\mathbb{N}) \) is uncountable
  - There is a bijection \( f: T \to P(\mathbb{N}) \) defined as \( f(s) = \{ i \mid s_i = 1 \} \)
The Uncountable

- **S uncountable** if no one-to-one \( f:S \rightarrow \mathbb{N} \) (equiv’ly, no onto \( f:\mathbb{N} \rightarrow S \))

- **Claim**: \( \mathbb{R} \) is uncountable

- **Related claims**:
  - Set \( T \) of all *infinitely long* binary strings is uncountable
    - Contrast with set of all finitely long binary strings, which is a countably infinite set
  - The power-set of \( \mathbb{N} \), \( P(\mathbb{N}) \) is uncountable
    - There is a bijection \( f: T \rightarrow P(\mathbb{N}) \) defined as \( f(s) = \{ i \mid s_i = 1 \} \)
    - e.g., set of even numbers in \( \mathbb{N} \) corresponds to the string 101010...
The Uncountable

- S **uncountable** if no one-to-one \( f: S \to \mathbb{N} \) (equiv’ly, no onto \( f: \mathbb{N} \to S \))

- **Claim:** \( \mathbb{R} \) is uncountable

- Related claims:
  - Set \( T \) of all **infinitely long** binary strings is uncountable
    - Contrast with set of all finitely long binary strings, which is a countably infinite set
  - The power-set of \( \mathbb{N} \), \( \mathcal{P}(\mathbb{N}) \) is uncountable
    - There is a bijection \( f: T \to \mathcal{P}(\mathbb{N}) \) defined as \( f(s) = \{ i \mid s_i = 1 \} \)

- How do we show something is not countable?!
  - e.g., set of even numbers in \( \mathbb{N} \) corresponds to the string 101010...
The Uncountable

S uncountable if no one-to-one \( f: S \rightarrow \mathbb{N} \) (equiv’ly, no onto \( f: \mathbb{N} \rightarrow S \))

Claim: \( \mathbb{R} \) is uncountable

Related claims:

- Set \( T \) of all infinitely long binary strings is uncountable
  - Contrast with set of all finitely long binary strings, which is a countably infinite set
- The power-set of \( \mathbb{N} \), \( \mathcal{P}(\mathbb{N}) \) is uncountable
- There is a bijection \( f: T \rightarrow \mathcal{P}(\mathbb{N}) \) defined as \( f(s) = \{ i \mid s_i = 1 \} \)

How do we show something is not countable?!

Cantor’s “diagonal slash”

e.g., set of even numbers in \( \mathbb{N} \) corresponds to the string 101010...
Cantor's Diagonal Slash
Cantor’s Diagonal Slash

Take any function $f: \mathbb{N} \rightarrow \mathcal{P}(\mathbb{N})$
Cantor’s Diagonal Slash

- Take any function \( f: \mathbb{N} \to \mathcal{P}(\mathbb{N}) \)
- Make a binary table with \( T_{ij} = 1 \) iff \( j \in f(i) \)

<table>
<thead>
<tr>
<th></th>
<th>f(0)</th>
<th>f(1)</th>
<th>f(2)</th>
<th>f(3)</th>
<th>f(4)</th>
<th>f(5)</th>
<th>f(6)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>f(0)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>f(1)</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>f(2)</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>f(3)</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>f(4)</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>f(5)</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>f(6)</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>
Cantor's Diagonal Slash

- Take any function \( f: \mathbb{N} \rightarrow \mathcal{P}(\mathbb{N}) \)
- Make a binary table with \( T_{ij} = 1 \) iff \( j \in f(i) \)
- Consider the set \( X \subseteq \mathbb{N} \) corresponding to the "flipped diagonal"

<table>
<thead>
<tr>
<th>( f(0) = )</th>
<th>1</th>
<th>0</th>
<th>0</th>
<th>1</th>
<th>0</th>
<th>0</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>( f(1) = )</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>( f(2) = )</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>( f(3) = )</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>( f(4) = )</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>( f(5) = )</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>( f(6) = )</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>
Cantor’s Diagonal Slash

- Take any function \( f: \mathbb{N} \to \mathcal{P}(\mathbb{N}) \)
- Make a binary table with \( T_{ij} = 1 \) iff \( j \in f(i) \)
- Consider the set \( X \subseteq \mathbb{N} \) corresponding to the “flipped diagonal”

\[
\begin{align*}
\begin{array}{cccccccc}
  & f(0) & f(1) & f(2) & f(3) & f(4) & f(5) & f(6) \\
\hline
0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\
1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\
2 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
3 & 1 & 1 & 0 & 1 & 0 & 1 & 0 \\
4 & 1 & 1 & 0 & 0 & 0 & 0 & 1 \\
5 & 0 & 1 & 0 & 1 & 1 & 0 & 1 \\
6 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\
\end{array}
\end{align*}
\]
Cantor's Diagonal Slash

- Take any function \( f : \mathbb{N} \rightarrow \mathcal{P}(\mathbb{N}) \)
- Make a binary table with \( T_{ij} = 1 \) iff \( j \in f(i) \)
- Consider the set \( X \subseteq \mathbb{N} \)
- corresponding to the “flipped diagonal”

\[ X = \{ j \mid T_{jj} = 0 \} = \{ j \mid j \not\in f(j) \} \]

<table>
<thead>
<tr>
<th>( f(0) )</th>
<th>( f(1) )</th>
<th>( f(2) )</th>
<th>( f(3) )</th>
<th>( f(4) )</th>
<th>( f(5) )</th>
<th>( f(6) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 0 0 1 0 0 0</td>
<td>0 0 1 0 1 0 0</td>
<td>1 1 1 1 1 1 1</td>
<td>1 1 0 1 0 1 0</td>
<td>1 1 0 0 0 0 1</td>
<td>0 1 0 1 1 0 1</td>
<td>0 1 0 1 0 1 0</td>
</tr>
</tbody>
</table>
Cantor's Diagonal Slash

- Take any function \( f: \mathbb{N} \rightarrow \mathcal{P}(\mathbb{N}) \)
- Make a binary table with \( T_{ij} = 1 \) iff \( j \in f(i) \)
- Consider the set \( X \subseteq \mathbb{N} \) corresponding to the "flipped diagonal"
  \[ X = \{ j \mid T_{jj} = 0 \} = \{ j \mid j \notin f(j) \} \]
- \( X \) doesn't appear as a row in this table (why?)

| \( f(0) \) | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| \( f(1) \) | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 |
| \( f(2) \) | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| \( f(3) \) | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 0 |
| \( f(4) \) | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 |
| \( f(5) \) | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1 |
| \( f(6) \) | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 |
Cantor’s Diagonal Slash

Take any function \( f: \mathbb{N} \to \mathcal{P}(\mathbb{N}) \)

Make a binary table with \( T_{ij} = 1 \) iff \( j \in f(i) \)

Consider the set \( X \subseteq \mathbb{N} \)

corresponding to the “flipped diagonal”

\( X = \{ j \mid T_{jj} = 0 \} \)
\( = \{ j \mid j \notin f(j) \} \)

\( X \) doesn’t appear as a row in this table (why?)

So \( f \) not onto

<table>
<thead>
<tr>
<th>( f(0) )</th>
<th>( f(1) )</th>
<th>( f(2) )</th>
<th>( f(3) )</th>
<th>( f(4) )</th>
<th>( f(5) )</th>
<th>( f(6) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 0 0 1 0 0 0</td>
<td>0 0 1 0 1 0 0</td>
<td>1 1 1 1 1 1 1</td>
<td>1 1 0 1 0 1 0</td>
<td>1 1 0 0 0 0 1</td>
<td>0 1 0 1 1 0 1</td>
<td>0 1 0 1 0 1 0</td>
</tr>
</tbody>
</table>
Cantor’s Diagonal Slash

Take any function \( f: \mathbb{N} \rightarrow \mathcal{P}(\mathbb{N}) \)

Make a binary table with \( T_{ij} = 1 \)
iff \( j \in f(i) \)

Consider the set \( X \subseteq \mathbb{N} \)
corresponding to the “flipped diagonal”

\[ X = \{ j \mid T_{jj} = 0 \} = \{ j \mid j \not\in f(j) \} \]

\( X \) doesn’t appear as a row in this table (why?)

So \( f \) not onto
Cantor’s Diagonal Slash

- Take any function \( f: \mathbb{N} \to \mathcal{P}(\mathbb{N}) \)
- Make a binary table with \( T_{ij} = 1 \) iff \( j \in f(i) \)
- Consider the set \( X \subseteq \mathbb{N} \)
  - corresponding to the “flipped diagonal”
    - \( X = \{ j \mid T_{jj} = 0 \} \)
    - \( = \{ j \mid j \notin f(j) \} \)
- \( X \) doesn’t appear as a row in this table (why?)
- So \( f \) not onto

Generalizes:
No onto function \( f: A \to \mathcal{P}(A) \) for any set \( A \)
Cantor’s Diagonal Slash

Take any function \( f: \mathbb{N} \rightarrow \mathcal{P}(\mathbb{N}) \)

Make a binary table with \( T_{ij} = 1 \) iff \( j \in f(i) \)

Consider the set \( X \subseteq \mathbb{N} \)
corresponding to the “flipped diagonal”

\[
X = \{ j \mid T_{jj} = 0 \} \\
= \{ j \mid j \notin f(j) \}
\]

\( X \) doesn’t appear as a row in this table (why?)

So \( f \) not onto

Generalizes:
No onto function \( f:A \rightarrow \mathcal{P}(A) \)
for any set \( A \)

May not have a table enumerating \( f \)
Cantor's Diagonal Slash

- Take any function \( f: \mathbb{N} \rightarrow \mathcal{P}(\mathbb{N}) \)
- Make a binary table with \( T_{ij} = 1 \) iff \( j \in f(i) \)
- Consider the set \( X \subseteq \mathbb{N} \)
corresponding to the “flipped diagonal”
  - \( X = \{ j \mid T_{jj} = 0 \} \)
    - \( = \{ j \mid j \notin f(j) \} \)
- \( X \) doesn’t appear as a row in this table (why?)
- So \( f \) not onto

Generalizes:

No onto function \( f: A \rightarrow \mathcal{P}(A) \)
for any set \( A \)
May not have a table enumerating \( f \)

Let \( X = \{ j \mid j \notin f(j) \} \)
Cantor’s Diagonal Slash

- Take any function $f: \mathbb{N} \rightarrow \mathcal{P}(\mathbb{N})$
- Make a binary table with $T_{ij} = 1$ iff $j \in f(i)$
- Consider the set $X \subseteq \mathbb{N}$ corresponding to the “flipped diagonal”
  \[ X = \{ j \mid T_{jj} = 0 \} = \{ j \mid j \notin f(j) \} \]
- $X$ doesn’t appear as a row in this table (why?)
- So $f$ not onto

Generalizes: No onto function $f: A \rightarrow \mathcal{P}(A)$ for any set $A$

May not have a table enumerating $f$

Let $X = \{ j \mid j \notin f(j) \}$

Claim: $\nexists i \in A$ s.t. $X = f(i)$
Cantor’s Diagonal Slash

- Take any function \( f: \mathbb{N} \rightarrow \mathcal{P}(\mathbb{N}) \)
- Make a binary table with \( T_{ij} = 1 \) iff \( j \in f(i) \)
- Consider the set \( X \subseteq \mathbb{N} \) corresponding to the “flipped diagonal”
  \[ X = \{ j \mid T_{jj} = 0 \} = \{ j \mid j \notin f(j) \} \]
- \( X \) doesn’t appear as a row in this table (why?)
- So \( f \) not onto

Generalizes:
- No onto function \( f: A \rightarrow \mathcal{P}(A) \) for any set \( A \)
- May not have a table enumerating \( f \)

Let \( X = \{ j \mid j \notin f(j) \} \)

Claim: \( \nexists \ i \in A \ s.t. \ X = f(i) \)

Suppose not: i.e., \( \exists i, \ X = f(i) \).
Cantor’s Diagonal Slash

- Take any function f: \( \mathbb{N} \rightarrow \mathcal{P}(\mathbb{N}) \)
- Make a binary table with \( T_{ij} = 1 \) iff \( j \in f(i) \)
- Consider the set \( X \subseteq \mathbb{N} \) corresponding to the “flipped diagonal”
  \[ X = \{ j \mid T_{jj} = 0 \} = \{ j \mid j \notin f(j) \} \]
- \( X \) doesn’t appear as a row in this table (why?)
- So f not onto

Generalizes:
- No onto function f:A→\( \mathcal{P}(A) \) for any set A
  - May not have a table enumerating f

Let \( X = \{ j \mid j \notin f(j) \} \)

Claim: \( \nexists i \in A \text{ s.t. } X = f(i) \)

Suppose not: i.e., \( \exists i, X = f(i) \).

\( i \in X \leftrightarrow i \in f(i) \leftrightarrow i \notin X \)
Cantor’s Diagonal Slash

- Take any function $f: \mathbb{N} \rightarrow \mathcal{P}(\mathbb{N})$
- Make a binary table with $T_{ij} = 1$ iff $j \in f(i)$
- Consider the set $X \subseteq \mathbb{N}$ corresponding to the “flipped diagonal”
  - $X = \{ j \mid T_{jj} = 0 \} = \{ j \mid j \notin f(j) \}$
- $X$ doesn’t appear as a row in this table (why?)
- So $f$ not onto

Generalizes:
No onto function $f:A \rightarrow \mathcal{P}(A)$ for any set $A$

May not have a table enumerating $f$

Let $X = \{ j \mid j \notin f(j) \}$

Claim: $\not\exists i \in A$ s.t. $X = f(i)$

Suppose not: i.e., $\exists i$, $X = f(i)$.
- $i \in X \iff i \in f(i) \iff i \notin X$
  - Contradiction!
Question
Question

Pick the correct statement. A is a non-empty set.

A. There is no one-to-one function from A to $\mathcal{P}(A)$
B. There is no onto function from $\mathcal{P}(A)$ to A
C. There is no one-to-one function from $\mathcal{P}(A)$ to A
D. There is a bijection between A and $\mathcal{P}(A)$ iff A is finite
E. None of the above
Question

Pick the correct statement. A is a non-empty set.

A. There is no one-to-one function from A to \( P(A) \)
B. There is no onto function from \( P(A) \) to A
C. There is no one-to-one function from \( P(A) \) to A
D. There is a bijection between A and \( P(A) \) iff A is finite
E. None of the above
Paradoxes and Relatives
Paradoxes and Relatives

Russell’s Paradox: In the universe of all sets, let

\[ S = \{ s \mid s \notin s \}. \] Then \( S \in S \iff S \notin S \)!!
Russell’s Paradox: In the universe of all sets, let $S = \{ s \mid s \not\in s \}$. Then $S \in S \iff S \not\in S$!!

“Naïve Set Theory” is inconsistent. Consistent theories developed which do not let one define such sets.
Russell’s Paradox: In the universe of all sets, let
\( S = \{ s \mid s \notin s \} \). Then \( S \in S \Leftrightarrow S \notin S \)!!

“Naïve Set Theory” is inconsistent. Consistent theories developed which do not let one define such sets.

In a library of catalogs, can you have a catalog of all catalogs in the library that don’t list themselves? (answer: No!)
Paradoxes and Relatives

Russell’s Paradox: In the universe of all sets, let
\[ S = \{ s \mid s \notin s \}. \] Then \( S \in S \iff S \notin S \).

“Naïve Set Theory” is inconsistent. Consistent theories developed which do not let one define such sets.

In a library of catalogs, can you have a catalog of all catalogs in the library that don’t list themselves? (answer: No!)

Liar’s paradox: “This statement is false.” (The statement is true iff it is false! Requires logic with “unknown/undefined” as truth value.)
Paradoxes and Relatives

Russell’s Paradox: In the universe of all sets, let $S = \{ s \mid s \notin s \}$. Then $S \in S \iff S \notin S$!!

“Naïve Set Theory” is inconsistent. Consistent theories developed which do not let one define such sets.

In a library of catalogs, can you have a catalog of all catalogs in the library that don’t list themselves? (answer: No!)

Liar’s paradox: “This statement is false.” (The statement is true iff it is false! Requires logic with "unknown/undefined" as truth value.)

Gödel numbered statements in a theory and showed that in any “rich” theory there must be a statement with number $g$ which says “statement with Gödel number $g$ is not provable”
Paradoxes and Relatives

Russell’s Paradox: In the universe of all sets, let
\[ S = \{ s \mid s \notin s \}. \] Then \( S \in S \iff S \notin S \) !!

“Naïve Set Theory” is inconsistent. Consistent theories developed which do not let one define such sets.

In a library of catalogs, can you have a catalog of all catalogs in the library that don’t list themselves? (answer: No!)

Liar’s paradox: “This statement is false.” (The statement is true iff it is false! Requires logic with “unknown/undefined” as truth value.)

Gödel numbered statements in a theory and showed that in any “rich” theory there must be a statement with number \( g \) which says “statement with Gödel number \( g \) is not provable”

This statement must be true if theory consistent (else a false statement is provable). Then the theory would be incomplete.
Reals are Uncountable
Reals are Uncountable

Enough to show a one-to-one mapping from $\mathbb{T}$, the set of infinite binary strings to the set of real numbers (why?)
Reals are Uncountable

Enough to show a one-to-one mapping from $T$, the set of infinite binary strings to the set of real numbers (why?)
Idea: treat a binary string $s_1s_2s_3...$ as the real number $0.s_1s_2s_3...$ in decimal
Reals are Uncountable

- Enough to show a one-to-one mapping from \( \mathbb{T} \), the set of infinite binary strings to the set of real numbers (\textit{why?})
- Idea: treat a binary string \( s_1s_2s_3... \) as the real number \( 0.s_1s_2s_3... \) in decimal
- This is a one-to-one mapping: a finite difference between the real numbers that two different strings map to
Reals are Uncountable

- Enough to show a one-to-one mapping from $\mathbb{T}$, the set of infinite binary strings to the set of real numbers (why?)
- Idea: treat a binary string $s_1s_2s_3...$ as the real number $0.s_1s_2s_3...$ in decimal
  - This is a one-to-one mapping: a finite difference between the real numbers that two different strings map to
- Note: if used binary representation instead of decimal representation, we’ll have strings $011111...$ and $10000...$ map to the same real number (though that can be handled)
Reals are Uncountable

 Enough to show a one-to-one mapping from $T$, the set of infinite binary strings to the set of real numbers (why?)

 Idea: treat a binary string $s_1s_2s_3...$ as the real number $0.s_1s_2s_3...$ in decimal

 This is a one-to-one mapping: a finite difference between the real numbers that two different strings map to

 Note: if used binary representation instead of decimal representation, we’ll have strings 011111... and 10000... map to the same real number (though that can be handled)

 On the other hand $|\mathbb{R}^2| = |\mathbb{R}|$. 
Reals are Uncountable

- Enough to show a one-to-one mapping from $T$, the set of infinite binary strings to the set of real numbers (why?)
- Idea: treat a binary string $s_1s_2s_3...$ as the real number $0.s_1s_2s_3...$ in decimal
- This is a one-to-one mapping: a finite difference between the real numbers that two different strings map to
- Note: if used binary representation instead of decimal representation, we’ll have strings 011111... and 10000... map to the same real number (though that can be handled)
- On the other hand $|\mathbb{R}^2| = |\mathbb{R}|$
- Because $|T^2|=|T|$ (bijection by interleaving), and we saw $|\mathbb{R}|=|T|$ (and hence $|\mathbb{R}^2|=|T^2|$ too)
The Uncomputable
The Uncomputable

We saw:
We saw:

The set of functions from $\mathbb{N}$ to $\mathbb{N}$ (or even from $\mathbb{N}$ to $\{0,1\}$) is uncountable
We saw:

- The set of functions from \(\mathbb{N}\) to \(\mathbb{N}\) (or even from \(\mathbb{N}\) to \(\{0,1\}\)) is uncountable.
- The set of finite length strings is countably infinite.
The Uncomputable

We saw:

- The set of functions from $\mathbb{N}$ to $\mathbb{N}$ (or even from $\mathbb{N}$ to $\{0,1\}$) is uncountable.
- The set of finite length strings is countably infinite.
- So we cannot assign a different string for each function.
The Uncomputable

We saw:
- The set of functions from \( \mathbb{N} \) to \( \mathbb{N} \) (or even from \( \mathbb{N} \) to \{0,1\}) is uncountable.
- The set of finite length strings is countably infinite.
- So we cannot assign a different string for each function.
- We cannot have a different program to compute each function.
We saw:

- The set of functions from $\mathbb{N}$ to $\mathbb{N}$ (or even from $\mathbb{N}$ to $\{0,1\}$) is uncountable.
- The set of finite length strings is countably infinite.

So we cannot assign a different string for each function.

We cannot have a different program to compute each function.

“Almost all” functions are uncomputable!
The Uncomputable
The Uncomputable

“Almost all” functions are uncomputable!
The Uncomputable

"Almost all" functions are uncomputable!

Later (CS 373): distinction between computing a predicate ("deciding") and "recognizing" when the predicate is true
The Uncomputable

“Almost all” functions are **uncomputable**!

Later (CS 373): distinction between computing a predicate ("deciding") and "recognizing" when the predicate is true

Almost all predicates “unrecognizable” too
The Uncomputable

“Almost all” functions are uncomputable!

Later (CS 373): distinction between computing a predicate ("deciding") and "recognizing" when the predicate is true

Almost all predicates “unrecognizable” too

But do we care about all these functions? Often yes!
The Uncomputable

“Almost all” functions are uncomputable!

Later (CS 373): distinction between computing a predicate ("deciding") and "recognizing" when the predicate is true

Almost all predicates “unrecognizable” too

But do we care about all these functions? Often yes!

Hilbert’s 10th problem: find an algorithm to check if a “Diophantine equation” has a solution (i.e., check if there is an integer solution to all the variables in a polynomial. e.g., the ones in Fermat’s last theorem, $x^3+y^3=z^3$, $x^4+y^4=z^4$, …)
The Uncomputable

“Almost all” functions are uncomputable!

Later (CS 373): distinction between computing a predicate ("deciding") and "recognizing" when the predicate is true

Almost all predicates “unrecognizable” too

But do we care about all these functions? Often yes!

Hilbert’s 10th problem: find an algorithm to check if a “Diophantine equation” has a solution (i.e., check if there is an integer solution to all the variables in a polynomial. e.g., the ones in Fermat’s last theorem, $x^3+y^3=z^3$, $x^4+y^4=z^4$, ...)

shown uncomputable in 1970
“Almost all” functions are uncomputable!

Later (CS 373): distinction between computing a predicate ("deciding") and "recognizing" when the predicate is true.

Almost all predicates “unrecognizable” too.

But do we care about all these functions? Often yes!

**Hilbert’s 10th problem:** find an algorithm to check if a “Diophantine equation” has a solution (i.e., check if there is an integer solution to all the variables in a polynomial. e.g., the ones in Fermat’s last theorem, \(x^3+y^3=z^3\), \(x^4+y^4=z^4\), ...)

**Hilbert’s Entscheidungsproblem:** given a statement in first order logic, check if it is true/provable.
**The Uncomputable**

“Almost all” functions are **uncomputable**!

Later (CS 373): distinction between computing a predicate ("deciding") and "recognizing" when the predicate is true

Almost all predicates “unrecognizable” too

But do we care about all these functions? Often yes!

**Hilbert’s 10th problem:** find an algorithm to check if a “Diophantine equation” has a solution (i.e., check if there is an integer solution to all the variables in a polynomial. e.g., the ones in Fermat’s last theorem, $x^3+y^3=z^3$, $x^4+y^4=z^4$, ...)

**Hilbert’s Entscheidungsproblem:** given a statement in first order logic, check if it is true/provable

shown uncomputable in 1970

shown uncomputable by Church and Turing [1936]
The Uncomputable

“Almost all” functions are uncomputable!

Later (CS 373): distinction between computing a predicate (“deciding”) and “recognizing” when the predicate is true.

Almost all predicates “unrecognizable” too.

But do we care about all these functions? Often yes!

Hilbert’s 10th problem: find an algorithm to check if a “Diophantine equation” has a solution (i.e., check if there is an integer solution to all the variables in a polynomial. E.g., the ones in Fermat’s last theorem, $x^3+y^3=z^3$, $x^4+y^4=z^4$, ...)

Hilbert’s Entscheidungsproblem: given a statement in first order logic, check if it is true/provable.

The Halting Problem: Given a (program,input) pair decide if the program halts or not.
The Uncomputable

“Almost all” functions are uncomputable!

Later (CS 373): distinction between computing a predicate ("deciding") and "recognizing" when the predicate is true.

Almost all predicates “unrecognizable” too.

But do we care about all these functions? Often yes!

Hilbert’s 10th problem: find an algorithm to check if a “Diophantine equation” has a solution (i.e., check if there is an integer solution to all the variables in a polynomial. E.g., the ones in Fermat’s last theorem, $x^3+y^3=z^3$, $x^4+y^4=z^4$, ...)

Hilbert’s Entscheidungsproblem: given a statement in first order logic, check if it is true/provable.

The Halting Problem: Given a (program,input) pair decide if the program halts or not.

Shown uncomputable in 1970.

shown uncomputable by Church and Turing [1936]
Discrete Structures

Wrap Up!
Discrete Structures (Intro to Mind Bending)

Wrap Up!
We Learned a Lot! :-)
We Learned a Lot! :-)  

Basic tools for expressing ideas  
Logic, Proofs,  
Sets, Relations, Functions
We Learned a Lot! :-)
We Learned a Lot! :-)

Basic tools for expressing ideas
Logic, Proofs,
Sets, Relations, Functions
We Learned a Lot! :-)

Basic tools for expressing ideas
Logic, Proofs,
Sets, Relations, Functions
We Learned a Lot! :-) 

Basic tools for expressing ideas
Logic, Proofs,
Sets, Relations, Functions
We Learned a Lot! :-)
We Learned a Lot! :-) 

Basic tools for expressing ideas
Logic, Proofs,
Sets, Relations, Functions
We Learned a Lot! :-) 

Basic tools for expressing ideas
Logic, Proofs,
Sets, Relations, Functions
We Learned a Lot! :-)
We Learned a Lot! :-)

Basic tools for expressing ideas
Logic, Proofs,
Sets, Relations, Functions
We Learned a Lot! :-)

Basic tools for expressing ideas
Logic, Proofs,
Sets, Relations, Functions

Numbers

Modular Arithmetic
- $[a]_m$ : the set of all elements $x$, s.t. $a = x \ (mod \ m)$
- Modular addition: $[a]_m + [b]_m = [a+b]_m$
- Modular multiplication: $[a]_m \times [b]_m = [a \cdot b]_m$
We Learned a Lot! :-)

Basic tools for expressing ideas
Logic, Proofs,
Sets, Relations, Functions

Numbers
Graphs

Bridges of Königsberg
Cross each bridge exactly once
If there is a walk that takes each edge exactly once, then only the end nodes of the walk can have odd degree (why?)
We Learned a Lot! :-)
We Learned a Lot! :-)

Basic tools for expressing ideas
Logic, Proofs,
Sets, Relations, Functions

Numbers

Graphs

Examples so far
- Complete graph $K_n$
- Complete bi-partite graph $K_{m,n}$
- Cycle graph $C_n$
- Path graph $P_n$
- Hypercube graph $Q_n$
We Learned a Lot! :-)

Basic tools for expressing ideas
Logic, Proofs,
Sets, Relations, Functions
We Learned a Lot! :-)
We Learned a Lot! :-)

Basic tools for expressing ideas
Logic, Proofs,
Sets, Relations, Functions
We Learned a Lot! :-)
We Learned a Lot! :-)
We Learned a Lot! :-)

Basic tools for expressing ideas
Logic, Proofs, Sets, Relations, Functions

Recursion
Induction
Numbers
Trees
Graphs
Bounding
big-O
Counting

Computation
Algorithms, Circuits, Grammars, Finite State Machines

Binary Search

Example: Finding (up to required precision) the square root of a number \( n \) (using only comparison and multiplication)

- Initial range: \([0, n]\) (say)
- How to compare desired object (here \( n \)) with middle element \( n/2 \)
- \( n \) and \( m \)
We Learned a Lot! :-)
We Learned a Lot! :-)

Basic tools for expressing ideas
Logic, Proofs, Sets, Relations, Functions

Recursion
Induction
Numbers
Trees
Graphs
Bounding big-O
Counting

Computation
Algorithms, Circuits, Grammars, Finite State Machines

Context-Free Grammar

Example: a (simplistic) syntax for arithmetic expressions
- Expr → Expr + Expr
- Expr → Expr × Expr
- Expr → Var
- Var → a
- Var → b
- Var → c
- Start: Expr
- Terminals +, ×, (, ), a, b, c
- (This grammar is "ambiguous" since there is another parse tree for the same string)
We Learned a Lot! :-)
We Learned a Lot! :-)

Basic tools for expressing ideas
Logic, Proofs, Sets, Relations, Functions

Recursion
Induction
Numbers

Trees
Graphs

Bounding big-O
Counting

Computation
Algorithms, Circuits, Grammars, Finite State Machines

Boolean Circuits

- A directed acyclic graph; Boolean valued wires, AND, OR, NOT gates, inputs, output
- Circuit evaluation OKT-VAL: given circuit C and inputs x, find C(x) (i.e., C boolean output value, on input x)
- Can be done very efficiently: if done in the right order, evaluating each wire takes O(1) time. OXT-VAL is in P
- OXT-SAT: given circuit C, is there a "satisfying" input for C (i.e., output)\(\neg\)? i.e., \(\exists x C(x)\neq 0\) in NP
We Learned a Lot! :-)

Basic tools for expressing ideas
Logic, Proofs, Sets, Relations, Functions

Recursion
Induction
Numbers
Trees
Graphs
Bounding big-O
Counting

P vs. NP
Computation
Algorithms, Circuits, Grammars, Finite State Machines

Boolean Circuits
- A directed acyclic graph: Boolean valued wires, AND, OR, NOT gates, inputs, output
- Circuit evaluation OKT-VAL: given circuit C and inputs x, find C(x) (i.e., C's boolean output value, on input x)
- Can be done very efficiently: if done in the right order, evaluating each wire takes O(1) time. OKT-VAL is in P
- OKT-SAT: given circuit C, is there a “satisfying” input for C (i.e., output)? (i.e., C(x)=1) in NP
- OKT-SAT: given C, is it that there is no satisfying input? (i.e., C(x)=0) in co-NP
We Learned a Lot! :-)

Basic tools for expressing ideas
Logic, Proofs, Sets, Relations, Functions

Recursion

Induction

Numbers

Trees

Graphs

Bounding big-O

Counting

P vs. NP

Computation
Algorithms, Circuits, Grammars, Finite State Machines

How do you count infinity?

We defined: $A$ is countably infinite if $|A| = |N|$, i.e., if there is a bijection between $A$ and $N$.

$\mathbb{N}$ is countable. Bijection by ordering points in $\mathbb{N}^2$ on a "curve":

$$(0,0), (1,0), (0,1), (2,0), (1,1), (0,2), \ldots$$

Note: $(0,0), (1,0), (2,0), \ldots$ will not give a bijection.

$\mathbb{R}$ is countable. $\mathbb{R}^1 \rightarrow \mathbb{N}$ defined as $f(x) = h(g(x),g(x))$, where $g: \mathbb{R} \rightarrow [0,1]$ and $h: [0,1] \rightarrow [0,1]$ are bijections.

More generally, if $A$ and $B$ are countable, the $A \times B$ is countable (extended to any finite number of sets by induction).
We Learned a Lot! :-)

Basic tools for expressing ideas:
- Logic, Proofs,
- Sets, Relations, Functions

Recursion

Induction

Numbers

Trees

Bounding big-O

Counting

Graphs

Uncomputability

P vs. NP

Computation
- Algorithms, Circuits, Grammars, Finite State Machines

How do you count infinity?
- We defined: A is countably infinite if |A| = |N|, i.e., if there is a bijection between A and N.
- \( \mathbb{N} \) is countable. Bijection by ordering points in \( \mathbb{N} \) on a “curve”
- \( (0,0), (1,0), (0,1), (2,0), (0,2), (1,1), (0,3), (2,1), (1,2), ... \)
- Note: \( (0,0), (1,0), (0,1) \) will not give a bijection
- \( \mathbb{R} \) is countable if \( \mathbb{R} \) is defined as \( f(x) = \frac{1}{x} \), where \( g(x) = x \cdot 2 \) and \( h(x) = x \cdot 3 \) are bijections, is a bijection.
- More generally, if A and B are countable, the \( \text{AxB} \) is countable (extended to any finite number of sets by induction)
We Learned a Lot! :-)

Basic tools for expressing ideas
Logic, Proofs,
Sets, Relations, Functions

Recursion
Induction
Numbers
Trees
Graphs
Bounding big-O
Counting
Uncomputability
P vs. NP
Computation
Algorithms, Circuits, Grammars,
Finite State Machines

Propositional Calculus
Basic tools for expressing Ideas
Logic, Proofs,
Sets, Relations, Functions
Logistics Wrap-Up
Logistics Wrap-Up

Final Exam on 18th 8:00 AM
Logistics Wrap-Up

- Final Exam on 18th 8:00 AM
- Locations: Main Library 66 (here) & Psychology 23
Logistics Wrap-Up

- Final Exam on 18th 8:00 AM
- Locations: Main Library 66 (here) & Psychology 23
- Seating details TBA
Logistics Wrap-Up

물을 18th 8:00 AM

Locations: Main Library 66 (here) & Psychology 23

Seating details TBA

Office hours: changes posted on Piazza
Logistics Wrap-Up

Final Exam on 18th 8:00 AM
Locations: Main Library 66 (here) & Psychology 23
Seating details TBA
Office hours: changes posted on Piazza
Tentative grades post midterm 2 to be posted today
Logistics Wrap-Up

- Final Exam on 18th 8:00 AM
- Locations: Main Library 66 (here) & Psychology 23
- Seating details TBA
- Office hours: changes posted on Piazza
- Tentative grades post midterm 2 to be posted today
- HW solutions to be posted tomorrow
Logistics Wrap-Up

Final Exam on 18th 8:00 AM
  Locations: Main Library 66 (here) & Psychology 23
  Seating details TBA
Office hours: changes posted on Piazza
Tentative grades post midterm 2 to be posted today
HW solutions to be posted tomorrow
A short PHQ9 to be released tomorrow, due Monday morning.
HW11 due Friday.
Logistics Wrap-Up

- Final Exam on 18th 8:00 AM
  - Locations: Main Library 66 (here) & Psychology 23
  - Seating details TBA
- Office hours: changes posted on Piazza
- Tentative grades post midterm 2 to be posted today
- HW solutions to be posted tomorrow
- A short PHQ9 to be released tomorrow, due Monday morning. HW11 due Friday.
- Now: ICES forms