BITS Pilani
Pilani Campus

innovate achieve lead

PROOF TECHNIQUES - DIAGONALIZATION

- Counting and Countability
- R is not countable

Counting: Sizes of sets

- Size of a set A is less than or equal to the size of a set B if there is a 1-to-1 function from A to B:
  - i.e. **A <=^{c} B if there exists f: A -->$^{1-1}$ B**

- If A <=^{c} B and B <=^{c} A then **A =^{c} B** i.e. **A is equinumerous with B.**
  - Note that **if there exists f: A -->$^{1-1\ onto}$ B then A =^{c} B**

Countable and Uncountable Sets

- A set B ***is countable*** if it is equinumerous with ***N*** or a subset of ***N***.
  - Examples of ***countably infinite*** sets:
    - ***Q***$^+$, the set of positive rational numbers
- [Proof Outline (***Q***$^+$ is countably infinite):
  - ***Q***$^+$ **= { m/n | m**$\in$***N*** and **n**$\in$***N*** such that ***gcd*****(m,n)=1 }**
  - Then ***Q***$^+$ can be seen as an infinite matrix **T** where **T[i,j]** denotes **i/j**
  - Construct an enumeration (i.e. 1-1 onto mapping of ***N*** to T[i,j]) :
    - Count by walking T along left-to-right, bottom-up, diagonals one after the other.
    - Note that each diagonal can be characterized by a fixed **c=i+j**

]

Cantor's second diagonal method (a.k.a. Diagonalization)

- Theorem: The set of infinite binary sequences

  $$B = \{ (b^0, b^1, b^2, \ldots) \mid b^i = 0 \text{ or } b^i = 1 \text{ for all } i \}$$

  is *uncountable*.
- Proof (by contradiction):
  - Suppose that B is countable: then there is an enumeration

    $B = \{ A^0, A^1, \ldots \}$ where for each n, A^n is a binary sequence.
  - Construct a table where each row is A^n for some n and each column is a bit position (*see below*).

A^0	**$a^{0,0}$**	**$a^{0,1}$**	**$a^{0,2}$**	**…**	
A^1	$a^{1,0}$	$a^{1,1}$	$a^{1,2}$	…	
A^2	$a^{2,0}$	$a^{2,1}$	$a^{2,2}$	…	
…	…				

Cantor's second diagonal method (a.k.a. Diagonalization)

- Theorem: **B = { (b^0, b^1, b^2, …) | b^i =0 or b^i =1 for all i }** is *uncountable*.
- Proof (by contradiction):
  - Suppose that B is countable:
    - enumerate elements of **B** as a table: *each row is* $\mathbf{A^n}$ *for some* **n** *and each column is a bit position* (*see below*).
  - Define sequence C: ***by flipping each bit (i.e. 0 to 1 or 1 to 0) along the (left-to-right, top-down) principal diagonal***
  - **C** $\neq$ $\mathbf{A^n}$ for any **n**.
    - Why?
  - i.e.

    C is not enumerated!
  - **B** is uncountable!

$a^{0,0}$	$a^{0,1}$	$a^{0,2}$	…	
$a^{1,0}$	$a^{1,1}$	$a^{1,2}$	…	
$a^{2,0}$	$a^{2,1}$	$a^{2,2}$	…	
…				

C'

C is the bit-wise complement of **C'**

R is not countable

- Theorem: ***The real interval (0,1) is not countable.***
  - Use the result from the previous slide:
    - interpret
      - each infinite binary sequence $(b^0, b^1, b^2, \ldots)$ as
      - the real number $0.b^0b^1b^2\ldots$ represented in binary notation.
    - Then **(0,1) $=^c$ B**
- Theorem:
  - **R $=^c$ B**
- Proof:
  - Find a bijunction from **R** to **(0,1)**

BITS Pilani
Pilani Campus

innovate achieve lead

PROOFS BY DIAGONALIZATION

Example:

• ***There are more non-computable problems than computable problems***.

Aside: Infinity of Infinities

- Exercise:
  - Consider the ***diagonalization*** technique used to prove that the set of all binary strings is uncountable.
  - Generalize it to prove:
    - **Cantor's Power-Set Theorem**:
      - For any set **S**, **S $<^c$ P(S)**
        - i.e. *the power-set of a set is strictly larger*.
- Question:
  - How many infinities do you have?

Class of Problems that are Not Computable

- **Theorem NonC:**
  - *There are more problems that are not computable than problems that are computable*.
- If "programs" in a general purpose programming language , say C, solve "computable problems",
  - then the size of the "class of computable problems" is at most the size of the "class of programs".
- **Theorem Problems-Programs** :
  - *There are more problems than there are C programs*.
- [Note: *Typically, Turing Machines are used as the standard, instead of C programs.* End of Note.]

Proof of ***Theorem Problems-Programs*** [by *Cardinality comparison*]:

- Number of programs (written in, say, C) is equal to $|\mathbb{N}|$

  By Lemma 1

- Let **S** be **{ f | f is a function from** $\mathbb{N}$ **to {0,1}}**.

  Then $|S| = 2^{|\mathbb{N}|}$ i.e. *the size of the power-set of* $\mathbb{N}$

  By Lemma 2.

- $|\mathbb{N}| < |\mathbb{P}(\mathbb{N})| = 2^{|\mathbb{N}|}$
  - By Cantor's Power-Set Theorem.

Proof of ***Theorem NonC***

- Set **Pb** of problems is of size > $2^{|\mathbb{N}|}$
- Set **Pr** of computable problems (i.e. programs) is of size $|\mathbb{N}|$
- Set of non-computable problems is of size $(2^{|\mathbb{N}|} - |\mathbb{N}|) > |\mathbb{N}|$

Lemma 1:

The number of programs that can be written using a given programming language, say C, is equal to $|\mathbb{N}|$, where $\mathbb{N}$ is the set of all natural numbers.

- Proof:

Define a bijection from the *set of all strings using a finite alphabet* to $\mathbb{N}$:

for j = 1, 2, …

for each string of length j, in lexicographic order (i.e. dictionary order):

assign a unique natural number (in increasing order)

[Note: *Let the size of the alphabet be K. Then each string can be coded as a unique number in base K+1.* End of Note.]

Lemma 2:

Consider the set **S = { f | f is a function from $\mathbb{N}$ to {0,1}}**

|S| = $2^{|\mathbb{N}|}$ i.e. size of the power-set of $\mathbb{N}$

Proof:

- <u>*Map each function* **f** *in* **S** *to*</u> a unique subset **T_f** of $\mathbb{N}$:

  f(x)=1 iff x is in the subset T_f

  This is a <u>*one-to-one, onto*</u> mapping [Why?]

Exercise:

- Prove that the given mapping is one-to-one:

  i.e. if $f \neq g$ then $T_f \neq T_g$
- Prove that given mapping is onto:

  i.e. for any subset T of $\mathbb{N}$ there is a corresponding f in S (that is mapped to T).