

Welcome to CSC438!

Introduction, Countability

Harry Sha

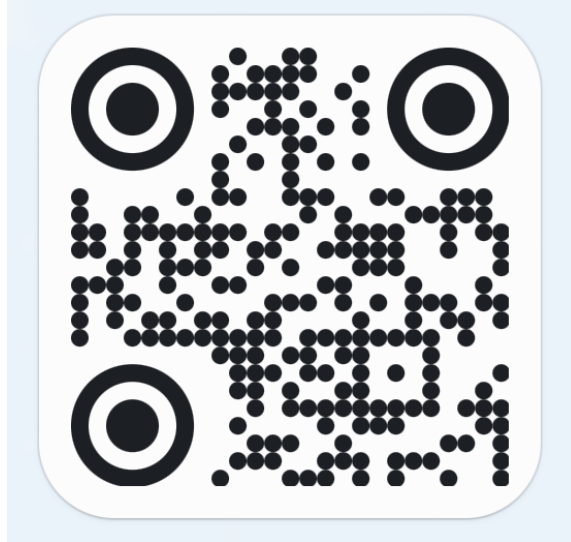
Fall 2026

Section 1

What is this course about?

Poll

<https://pe.app/harrysha>



Question

What do we want from a formal proof system?

- ① "Easy to follow". It should be easy to determine if a ~~a~~ string is a valid proof or not.
- ② For all sentences φ . We should be able to prove φ or $\neg\varphi$.
- ③ Soundness: We shouldn't be able to prove "false" things
- ④ Strong enough to reason about mathematical objects of interest

Highlight: Incompleteness

A highlight of this course is a proof that Hilbert's program is *unattainable*.

This is Gödel's Incompleteness Theorem.

Course overview

- Computability
- Propositional Logic
- First-Order Logic
- Incompleteness

Potential extra topics

- Kolmogorov complexity
- Set theory
- Efficient proofs

Logistics

- <https://www.cs.toronto.edu/~shaharry/csc438/>
- Syllabus.
- Piazza.
- Announcements made by email (via Quercus).

Culture

- Questions are extremely welcome.
- Feedback is also extremely welcome.
- Support each other and be nice to each other.

A favor

If you know someone that would be interested in the course, please let them know about it and encourage them to join the course :-)

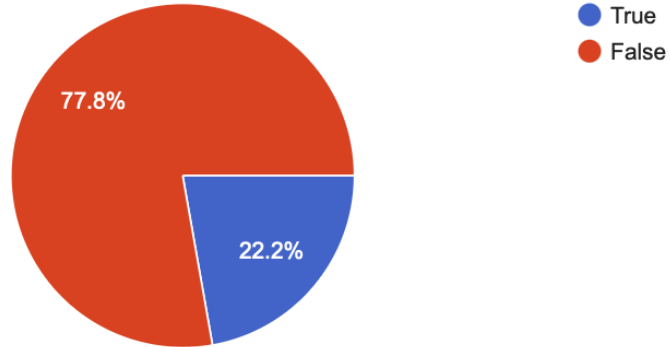
Intake Form

Most common prerequisites are MAT247 and CSC373

Intake Form

Is the following sentence true? "This sentence is false"

9 responses



Section 2

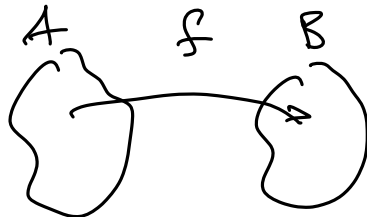
Background: Infinity

Pigeonhole Principle

If there are more pigeons than pigeonholes, then ... there has to be a pigeonhole w/ more than 1 pigeon.

if $|Pigeons| > |Pigeonholes|$ then there is no injective function $f: Pigeons \rightarrow Pigeonholes$.

Comparing the sizes of sets



there's no
• > •

Definition

Let A and B be sets (potentially infinite). $|A| \leq |B|$ if and only if there exists an injective function $f : A \rightarrow B$.

Notes:

- Equivalently¹, $|A| \leq |B|$ if and only if there exists a surjective function $f : B \rightarrow A$.
- The definition of $\leq$ induces definitions of $\geq, =, <, >$.

¹there are some caveats to this which we will explore in the problem set

Question

- Suppose $A \subseteq B$. Is it the case that $|A| \leq |B|$?
- Suppose $A \subsetneq B$. Is it the case that $|A| < |B|$?

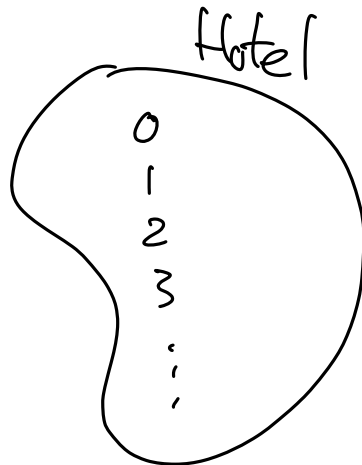
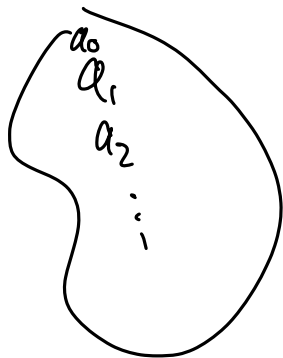
yes! $\forall a \exists a$.

no!

Example: Hilbert's Hotel

Hilbert's Hotel is an infinite hotel with rooms numbered $0, 1, 2, \dots$. Note that each room can only hold a **single** person.

Your job as the receptionist is to assign customers to hotel rooms.



Example

Two infinite buses

$$a_n \rightarrow z_n$$

$$b_n \rightarrow z_{n+1}$$

Example

An infinite number of infinite buses

Diagram illustrating a grid representing an infinite number of infinite buses. The vertical axis is labeled "person" and the horizontal axis is labeled "bus". The grid contains numbers 1 through 10, with red arrows indicating a diagonal traversal pattern.

	1	2	3	4
1	1	2	4	7
2	3	5	6	
3	6	9		
4	10			

$f(i,j)$ = order in which (i,j) is hit by the diagonals.

bus person

$$(i,j) \mapsto (i^{\text{th}}_{\text{prime}})^{j+1}$$

$$(i,j) \mapsto 2^i 3^j$$

Countable

Definition

A set A is countable if and only if it fits in Hilbert's Hotel.

I.e. $|A| \leq |\mathbb{N}|$.

I.e. There exists an injective function $A \rightarrow \mathbb{N}$, or a surjective function $\mathbb{N} \rightarrow A$.

Enumeration

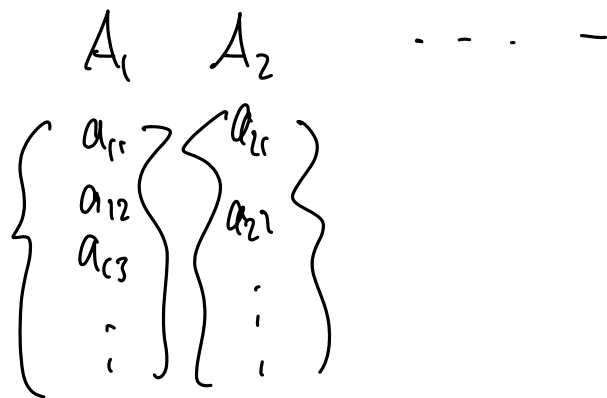
$$f: \mathbb{N} \rightarrow A .$$

$$f(i) = a_i$$

A surjective function $\mathbb{N} \rightarrow A$ is just an enumeration of the elements of A . I.e. $A = \{a_1, a_2, \dots\}$. Hence, a convenient definition of “ A is countable” is “ A has an enumeration”.

Summary

- The union of two countable sets is countable.
- $\mathbb{N} \times \mathbb{N}$ is countable
- The union of countably many countable sets is countable.



Exercise (later)

- How about $\mathbb{N}^3 = \mathbb{N} \times \mathbb{N} \times \mathbb{N}$?

Strings

$$\{0,1\}, \{a,b,c \dots\}$$

An alphabet is any finite set. We call the elements of the alphabet *characters* or *symbols*.

A *string* over Σ is a finite sequence of characters from Σ .

- Σ^n is the set of strings of length n over Σ
- Σ^* is the set of strings over Σ

Special case: $\Sigma^0 = \{\epsilon\}$, where ϵ is the empty string.

Examples

$$\Sigma = \{0, 1\}$$

$$\Sigma^2 = \{01, 00, 10, 11\}.$$

$$\Sigma^* = \{\epsilon, 0, 1, 00, 000, \dots\}$$

Question

Let Σ be a finite alphabet. Is Σ^* countable?

$$\Sigma^* = \bigcup_{n=0}^{\infty} \Sigma^n$$

countable!

by a few slides ago, Σ^* is countable.

Standard String Order

Here's an explicit enumeration of $\{0, 1\}^*$.

$\epsilon, 0, 1, 00, 01, 10, 11, 000, \dots$

The strings are ordered first by length and then by lexicographical order (dictionary order). This ordering is called *Standard String Order*.

Question

Are there sets that don't fit into Hilbert's Hotel? I.e. sets that are not countable?

Summary

- $|A| \leq |B|$ means there is an injective function $f : A \rightarrow B$
- A set is countable if and only if it fits into Hilbert's Hotel.
- $\mathbb{N} \times \mathbb{N}$ is countable. Σ^* is countable (where Σ is finite).

Section 3

Cantor's Theorem

Powerset

For any set A , the powerset of A , denoted $\mathcal{P}(A)$, is the set of all subsets of A .

Example:

$$A = \{0, 1\}.$$
$$\mathcal{P}(A) = \{ \{0\}, \{1\}, \underline{\{0, 1\}}, \emptyset \}$$

Theorem (Cantor, 1891)

Let A be a set. Then $|A| < |\mathcal{P}(A)|$. Equivalently, there does not exist a surjection from A to $\mathcal{P}(A)$.

Theorem (Cantor, 1891)

Let A be a set. Then $|A| < |\mathcal{P}(A)|$. Equivalently, there does not exist a surjection from A to $\mathcal{P}(A)$.

In particular, $\mathcal{P}(\mathbb{N})$ does not fit into Hilbert's Hotel.

Proof (Assuming A is countable)

By contradiction, suppose $\exists$ surjection $f: A \rightarrow P(A)$.

	D	a_1	a_2	a_3	$\dots a_j$
$f(a_1)$		$\checkmark$	$\times$	$\times$	$\checkmark$
$f(a_2)$		$\checkmark$	$\times$	$\times$	$\times$
$f(a_3)$		$\times$	$\times$	$\times$	$\times$
$\vdots$					

Claim: $D \neq f(a_j)$ for any j

suppose $D = \underline{f(a_j)}$.

if $a_j \in D$, $a_j \notin \underline{f(a_j)} \Rightarrow a_j \notin D$
 Contr!

if $a_j \notin D$, $a_j \in \underline{f(a_j)} \Rightarrow a_j \in D$.

Proof (General case)

$$D = \{a \in A : a \notin f(a)\},$$

notice $D \subseteq A$, so... $D = f(a)$ for some $a \in A$

suppose $D = f(a)$. now, we ask: is $a \in D$?

if $a \in D : \Rightarrow a \notin f(a) \Rightarrow a \notin D \Rightarrow \Leftarrow$

$a \notin D : \Rightarrow a \in f(a) \Rightarrow a \in D \Rightarrow \Leftarrow$

(surjectivity
of f)

Section 4

An Application of Cantor's Theorem to Computability

Decision Problems

A decision problem is a problem of the following form.

Given a set of strings A , decide whether or not the input is in A .

$$\text{EndsIn1} = \{1, 01, 11, 001, \dots\}.$$

Question: What is the set of decision problems?

$$\mathcal{P}(\Sigma^*)$$

Programs

```
def f():  
    print(1)
```

A particular program solves a problem $A \subseteq \Sigma^*$ if

$a \in A \leftrightarrow$ the program prints 1 on input a .

```
def end_1(x):  
    if x[-1] == "1": print(1)
```

Programs

A particular program *solves* a problem $A \subseteq \Sigma^*$ if

$$a \in A \leftrightarrow \text{the program prints 1 on input } a.$$

Programs can be identified by their source code.

Question: What can we say about the set of programs?

$$\text{Programs} = \Sigma^*$$

Question

Can a single program solve more than one problem?

Let P be a program, and $A, B \subseteq \Sigma^*$

st. $A \neq B$.

 A and B disagree on some element x :

WLOG $x \in A, x \notin B$.

$P(x)$

Putting the previous slides together

$$\text{Problems} = \mathcal{P}(\Sigma^*)$$

$$\text{Programs} = \Sigma^*$$

$$|\text{Programs}| < |\text{Problems}|.$$

in particular, if $\text{Solves} : \text{Programs} \rightarrow \text{Problems}$

Solves is not surjective by Cantor's Theorem.

Exercise (later)

How many problems are unsolvable by this argument?

Summary

- Cantor's Theorem says $|A| < |\mathcal{P}(A)|$.
- Programs = Σ^* , and DecisionProblems = $\mathcal{P}(\Sigma^*)$. So

$$|\text{Programs}| < |\text{DecisionProblems}|.$$

There exist decision problems that cannot be solved computationally.

»

We only used the following assumptions about programs.

- Programs are deterministic (i.e. they can't "solve" multiple decision problems), and
- Programs have finite descriptions (so that we can identify them with strings).

Next:

- A formal model of computation.
- Two explicit and natural problems that are undecidable.