Poll: How big is infinity?

Mark what's true.
(A) There are more real numbers than natural numbers.
(B) There are more rational numbers than natural numbers.
(C) There are more integers than natural numbers.
(D) pairs of natural numbers

Same Size. Poll.

Two sets are the same size?
(A) Bijection between the sets.
(B) Count the objects and get the same number. same size.
(C) Counting to infinity is hard.
(A), (B),

Countable.

How to count?
0, 1, 2, 3,…
The Counting numbers.
The natural numbers! N

Definition: S is countable if there is a bijection between S and some subset of N. If the subset of N is finite, S has finite cardinality. If the subset of N is infinite, S is countably infinite.

Countably infinite subsets.

Enumerating a set implies countable.
Corollary: Any subset T of a countable set S is countable.
Enumerate T as follows:
Get next element, x, of S,
output only if x ∈ T.
Implications:
Z⁺ is countable.
It is infinite since the list goes on.
There is a bijection with the natural numbers.
So it is countably infinite.
All countably infinite sets have the same cardinality.

Enumeration example.

All binary strings.
B = \{0,1\}⁺.
B = { ϕ, 0.0, 0.01, 0.1, 0.10, 0.100, 0.1000, 0.10000, 0.100000, …}. ϕ is empty string.
For any string, it appears at some position in the list.
If n bits, it will appear before position 2ⁿ⁺¹.
Should be careful here.
B = { ϕ, 0.0, 0.00, 0.000, 0.0000, 0.00000, 0.000000, …}
Never get to 1.

More fractions?

Enumerate the rational numbers in order...
0,..., 1/2...
Where is 1/2 in list?
After 1/3, which is after 1/4, which is after 1/5...
A thing about fractions:
any two fractions has another fraction between it.
Can’t even get to "next" fraction!
Can’t list in "order".
Consider pairs of natural numbers: \( N \times N \)

E.g.: \((1, 2), (100, 30), \) etc.

For finite sets \( S_1 \) and \( S_2 \),
then \( S_1 \times S_2 \) has size \( |S_1| \times |S_2| \).

So, \( N \times N \) is countably infinite squared ???

The pair \((a, b)\), is in first \( (a + b + 1)(a + b)/2 \) elements of list!

(i.e., "triangle").

Enumerate in list:

\[(0,0), (1,0), (0,1), (2,0), (1,1), (0,2), \ldots \]

Countably infinite.

The rationals are countably infinite.

Are the set of reals countable?

Let's consider the reals \([0,1]\).

Each real has a decimal representation.

\[
\begin{align*}
500000000\ldots \ (1/2) \\
785398162\ldots \ \pi/4 \\
367879441\ldots \ 1/e \\
632120558\ldots \ 1 - 1/e \\
345212312\ldots \ Some \ real \ number
\end{align*}
\]
Diagonalization.
If countable, there a listing, \( L \) contains all reals. For example
0: 0.500000000...
1: 0.785398162...
2: 0.367879441...
3: 0.632120558...
4: 0.345212312...
...
Construct “diagonal” number: 0.77677...
Diagonal Number: Digit \( i \) is 7 if number \( i \)'s \( i \)th digit is not 7 and 6 otherwise.
Diagonal number for a list differs from every number in list!
Diagonal number not in list.
Diagonal number is real.
Contradiction!

Subset \([0,1]\) is not countable!!

All reals?
What about all reals?
No.
Any subset of a countable set is countable.
If reals are countable then so is \([0,1]\).

Poll: diagonalization Proof.
Mark parts of proof.
(A) Integers are larger than naturals cuz obviously.
(B) Integers are countable cuz, interleaving bijection.
(C) Reals are uncountable cuz obviously!
(D) Reals can’t be in a list: diagonal number not on list.
(E) Powerset in list: diagonal set not in list.
(B), (C)?, (D), (E)

The Continuum hypothesis.
There is no set with cardinality between the naturals and the reals.
First of Hilbert’s problems!
## Cardinalities of uncountable sets?

Cardinality of $[0, 1]$ smaller than all the reals?

Let $f : R^+ \rightarrow [0, 1]$.

$$f(x) = \begin{cases} 
  x + \frac{1}{2} & 0 \leq x \leq \frac{1}{2} \\
  \frac{1}{2} & x > \frac{1}{2}
\end{cases}$$

### One to one.

If $x \neq y$:
- If both in $[0, 1/2]$, a shift $\Rightarrow f(x) \neq f(y)$.
- If neither in $[0, 1/2]$ a division $\Rightarrow f(x) \neq f(y)$.
- If one is in $[0, 1/2]$ and one isn't, different ranges $\Rightarrow f(x) \neq f(y)$.

### Bijection!

$[0, 1]$ is the same cardinality as nonnegative reals!

## Generalized Continuum hypothesis.

There is no infinite set whose cardinality is between the cardinality of an infinite set and its power set.

The powerset of a set is the set of all subsets.

## Resolution of hypothesis?

Gödel. 1940.

Can't use math!

If math doesn't contain a contradiction.

This statement is a lie.

Is the statement above true?

The barber shaves every person who does not shave themselves.

Who shaves the barber?

Self reference.

Can a program refer to a program?

Can a program refer to itself?

Uh oh....

## The Barber!

The barber shaves every person who does not shave themselves.

(A) Barber not Mark. Barber shaves Mark.
(B) Mark shaves the Barber.
(C) Barber doesn’t shave themself.
(D) Barber shaves themself.

It's all true. It's all a problem.
Changing Axioms?

Goedel:
Any set of axioms is either inconsistent (can prove false statements) or incomplete (true statements cannot be proven.)

Concrete example:
Continuum hypothesis: “no cardinality between reals and naturals.”
Continuum hypothesis not disprovable in ZFC
(Goedel 1940.)

BTW:
Cantor..bipolar disorder...
Goedel..starved himself out of fear of being poisoned...
Russell..was fine.....but for...two schizophrenic children..
Dangerous work?

See Logicomix by Doxiaidis, Papadimitriou (was professor here), Papadatos, Di Donna.

Is it actually useful?

Write me a program checker!
Check that the compiler works!
How about.. Check that the compiler terminates on a certain input.

HALT(P, I)
P - program
I - input.
Determines if P(I) (P run on I) halts or loops forever.
Run P on I and check!
How long do you wait?
Something about infinity here, maybe?

Implementing HALT.

HALT(P, I)
P - program
I - input.
Determines if P(I) (P run on I) halts or loops forever.
Theorem: There is no program HALT.
Proof: Assume there is a program HALT(·,·).
Turing(P)
1. If HALT(P, P) = “halts”, then go into an infinite loop.
2. Otherwise, halt immediately.
Assumption: there is a program HALT.
There is text that "is" the program HALT.
Can run Turing on Turing!
Does Turing(Turing) halt?
Turing(Turing) halts
then HALTS(Turing, Turing) = halts
then Turing(Turing) loops forever.

Halt and Turing.

Proof: Assume there is a program HALT(·,·).
Turing(P)
1. If HALT(P, P) = “halts”, then go into an infinite loop.
2. Otherwise, halt immediately.
Assumption: there is a program HALT.
There is text that "is" the program HALT.
Can run Turing on Turing!

Does Turing(Turing) halt?
Turing(Turing) halts
then HALTS(Turing, Turing) = halts
then Turing(Turing) loops forever.
Contradiction. Program HALT does not exist!
Questions?
Another view of proof: diagonalization.

Programs?

What are programs?

(A) Instructions.
(B) Text.
(C) Binary String.
(D) They run on computers.

All are correct.

Proof play by play.

Assumed HALT(P,I) existed.
What is P? Text.
What is I? Text.
What does it mean to have a program HALT(P,I)?

You have Text that is the program HALT(P,I).
Have Text that is the program TURING.
Here it is!!

Turing(P)
1. If HALT(P,P) \( \Rightarrow \) "halts", then go into an infinite loop.
2. Otherwise, halt immediately.

Turing "diagonalizes" on list of program.
It is not a program!!!!

\[ \equiv \] HALT is not a program.

Questions?

No computers for Turing!

We are so smart!

Wow, that was easy!
We should be famous!

Turing machine.

A Turing machine.
– an (infinite) tape with characters
– be in a state, and read a character
– move left, right, and/or write a character.

Universal Turing machine
– an interpreter program for a Turing machine
– where the tape could be a description of a ... Turing machine!

Now that's a computer!

Turing: AI, self modifying code, learning...
Turing and computing.

Just a mathematician?
“Wrote” a chess program.
Simulated the program by hand to play chess.
It won! Once anyway.
Involved with computing labs through the 40s.
Helped Break the enigma code.
The polish machine...the bomba.

Computing on top of computing...

Computer, assembly code, programming language, browser, html, javascript...
We can’t get enough of building more Turing machines.

More about Alan Turing.

▶ Brilliant codebreaker during WWII, helped break German Enigma Code (which probably shortened war by 1 year).
▶ Seminal paper in numerical analysis: Condition number. Math 54 doesn’t really work. Almost dependent matrices.
▶ Imitation Game.

Turing: personal.

Tragic ending...
▶ Arrested as a homosexual, (not particularly closeted)
▶ given choice of prison or (quackish) injections to eliminate sex drive;
▶ took injections.
▶ lost security clearance...
▶ suffered from depression;
▶ (possibly) suicided with cyanide at age 42 in 1954. (A bite from the apple...) accident?

Undecidable problems.

Does a program, $P$, print “Hello World”?
How? What is $P$? Text!!!!!!
Find exit points and add statement: $\text{Print }$ “Hello World.”

Can a set of notched tiles tile the infinite plane?
Proof: simulate a computer. Halts if finite.

Does a set of integer equations have a solution?
Example: $x^n + y^n = 1$?
Problem is undecidable.
Be careful!
Is there an integer solution to $x^n + y^n = 1$?
(Diophantine equation.)
The answer is yes or no. This “problem” is not undecidable.
Undecidability for Diophantine set of equations $\implies$ no program can take any set of integer equations and always correctly output whether it has an integer solution.

Back to technical..

This statement is a lie. Neither true nor false!
Every person who doesn’t shave themselves is shaved by the barber.
Who shaves the barber?

def Turing($P$):
    if Halts($P$):
        while(true): pass
    else:
        return

...Text of Halt...

Halts Program $\iff$ Turing Program. ($P \iff Q$)
Turing(“Turing”)? Neither halts nor loops! $\implies$ No Turing program.
No Turing Program $\iff$ No halt program. ($\neg Q \iff \neg P$)
Program is text, so we can pass it to itself, or refer to self.
**Summary: decidability.**

- Computer Programs are an interesting thing.
  - Like Math.
  - Formal Systems.
- Computer Programs cannot completely “understand” computer programs.
- Computation is a lens for other action in the world.

**Kolmogorov Complexity, Google, and CS70**

- Of strings, \( s \).
  - Minimum sized program that prints string \( s \).
  - What Kolmogorov complexity of a string of 1,000,000, one’s?
  - What is Kolmogorov complexity of a string of \( n \) one’s?
    - for \( i = 1 \) to \( n \): print ‘1’.

**Kolmogorov Complexity, Google, and CS70**

- What is the minimum I need to know (remember) to know stuff.
  - Radius of the earth?
  - Distance to the sun?
  - Population of the US?
  - Acceleration due to gravity on earth?
  - Syntax of pandas? Google + Stackoverflow.
  - Plus “how to program” and remembering a bit.
- Calculus: what is minimum you need to know?
  - Depends on your skills!
  - Reason and understand an argument and you can generate a lot.

**Calculus**

- What is the first half of calculus about?
  - The slope of a tangent line to a function at a point.
  - Slope is rise/run. Oh, yes: \( \lim_{h \to 0} \frac{f(x + h) - f(x)}{h} \).
  - Chain rule? Derivative of a function composition.
    - Intuition: composition of two linear functions?
      - \( f(x) = ax \), \( g(x) = bx \). \( f(g(x)) = ab x \). Slope is ab.
      - Multiply slopes!
        - \( f'(g(x))' = f'(g(x)) \cdot g'(x) \)
        - But...but...
      - For function slopes of tangent differ at different places.
        - So, where? \( f'(g(x)) \) slope of \( f \) at \( g(x) \) times slope of \( g \) at \( x \).
        - \( f'(g(x))' = f'(g(x)) \cdot g'(x) \).

**Product Rule.**

- Idea: use rise in function value!
  - \( d(\text{uv}) = (u + du)(v + dv) - uv = udv + vdu + dudv \to udv + vdu \).
- Any concept:
  - A quick argument from basic concept of slope of a tangent line.
  - Perhaps.

**Derivative of sine?**

- \( \sin(x) \).
- What is \( x \)? An angle in radians.
- Let’s call it \( \theta \) and do derivative of \( \sin \theta \).
- \( \theta \) - Length of arc of unit circle
  - \( \sin \theta \rightarrow \text{Rise. Similar triangle!!!} \)
Arguments, reasoning.

What you know: slope, limit.
Plus: definition.
yields calculus.
Minimization, optimization, ....
Knowing how to program plus some syntax (google) gives the ability to program.
Knowing how to reason plus some definition gives calculus.
Discrete Math: basics are counting, how many, when are two sets the same size?
Probability: division.
...plus reasoning.

CS 70 : ideas.

Induction = every integer has a next one. Graph theory.
Number of edges is sum of degrees.
\( \Delta + 1 \) coloring. Neighbors only take up \( \Delta \).
Connectivity plus connected components.
Eulerian paths: if you enter you can leave.
Euler's formula: tree has \( v - 1 \) edges and 1 face plus each extra edge makes additional face.
\[ v - 1 = (f - 1) + e \]

Number theory.
A divisor of \( x \) and \( y \) divides \( x - y \).
The remainder is always smaller than the divisor.
\[ \Rightarrow \] Euclid's GCD algorithm.
Multiplicative Inverse.
Fermat's theorem from function with inverse is a bijection.
Gives RSA.
Error Correction.
(Any) Two points determine a line.
(well, and \( d \) points determine a degree \( d + 1 \)-polynomials.
Cuz, factoring.
Find line by linear equations.
If a couple are wrong, then multiply them by zero, i.e., Error polynomial.

CS70 and your future?

What's going on?
Define. Understand properties. And build from there.
Tools: reasoning, proofs, care.
Gives power to your creativity and in your pursuits.
....and you will pursue probability in this course.