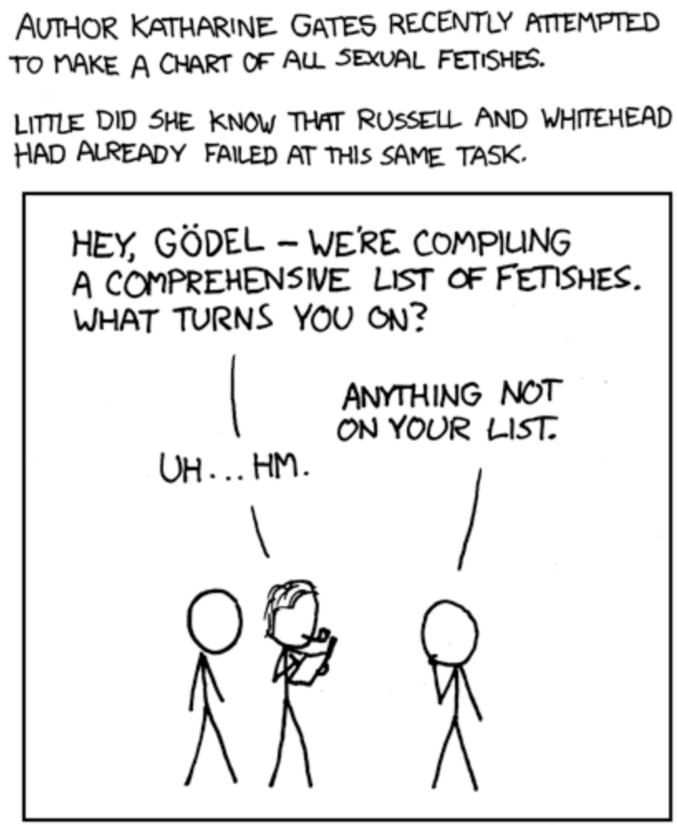
CSE 311: Foundations of Computing

Lecture 26: Cardinality



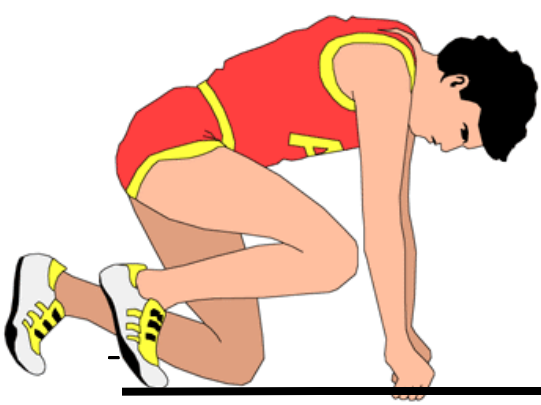
Cardinality and Computability

Computers as we know them grew out of a desire to avoid bugs in mathematical reasoning

A brief history of reasoning

Ancient Greece

- Deductive logic
  - Euclid's Elements
- Infinite things are a problem
  - Zeno's paradox



Starting with Cantor

- How big is a set?
  - If S is finite, we already defined |S| to be the number of elements in S.
  - What if S is infinite? Are all of these sets the same size?

    Natural numbers $\mathbb{N}$

    Even natural numbers

    Integers $\mathbb{Z}$

    Rational numbers $\mathbb{Q}$

    Real numbers $\mathbb{R}$

Size!

Two sets A and B have the same when...

Injectivity, Surjectivity, and Bijectivity

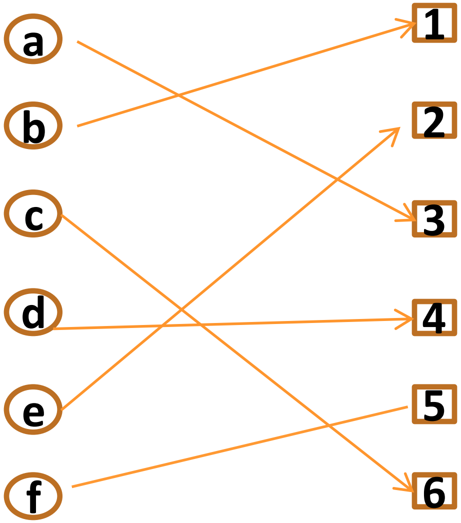
A function $f : A \to B$ is **injective** when every element is mapped to by *at most* one input.

A function, $f : A \to B$, is **surjective** when every element is mapped to by *at least* one input.

A function, $f : A \to B$, is **bijective** when every element is mapped to by *exactly* one input.

Cardinality

Two sets A and B have the same size (same cardinality) iff there is a bijection $f : A \rightarrow B$.



Cardinality

Consider the function $f : \mathbb{N} \to \mathbb{E}$ where $f(n) = 2n$.

$f(0)$	=	0
$f(1)$	=	2
$f(2)$	=	4
$f(3)$	=	6
$f(4)$	=	8
$f(5)$	=	10
$f(6)$	=	12
$f(7)$	=	14

Every Natural Number appears on the left

Every Even Natural Number appears on the right

Countability

A set S is *countable* iff there is an surjective function g: $\mathbb{N} \rightarrow$ S and S is infinite. Recall, this means that every number in S is mapped to.

A set S is *countable* iff we can list out the members of S without missing any.

Integers

Consider the function $f : \mathbb{N} \to \mathbb{Z}$ where $f(n) = \ldots$

$f(0)$	$=$	0
$f(1)$	$=$	-1
$f(2)$	$=$	1
$f(3)$	$=$	-2
$f(4)$	$=$	2
$f(5)$	$=$	-3
$f(6)$	$=$	3

Every Natural Number appears on the left

Every Integer appears on the right

Insight: Programs are Functions!

If we can write a program that prints out all the numbers in a set (each exactly once), then that set is enumerable!

```
public static void enumerateZ() {
   int positive = 0;
   int negative = -1;
   while (true) {
      System.out.println(positive);
      System.out.println(negative);
      positive++;
      negative--;
   }
}
```

The set of all integers is countable

```
public static void enumerateZ() {
   int positive = 0;
   int negative = -1;
   while (true) {
      System.out.println(positive);
      System.out.println(negative);
      positive++;
      negative--;
   }
}
```

We need to show that for any integer, x, enumerateZ prints x.
Suppose x is non-negative. The xth iteration through the loop will print x, because we always print `positive` and increment it each time.
Suppose x is negative. Then, x = -y for some non-negative y.
The (y-1)st iteration through the loop will print x, because we decrement `negative` each time.
Since all integers are negative or non-negative, we list all possible integers.

Is the set of positive rational numbers countable?

Between any two rational numbers there are an infinite number of others...

The set of positive rational numbers is countable

1/1	1/2	1/3	1/4	1/5	1/6	1/7	1/8	...
2/1	2/2	2/3	2/4	2/5	2/6	2/7	2/8	...
3/1	3/2	3/3	3/4	3/5	3/6	3/7	3/8	...
4/1	4/2	4/3	4/4	4/5	4/6	4/7	4/8	...
5/1	5/2	5/3	5/4	5/5	5/6	5/7	...	
6/1	6/2	6/3	6/4	6/5	6/6	...		
7/1	7/2	7/3	7/4	7/5	....			
...	...	...	...	...				

The set of positive rational numbers is countable

$$\mathbb{Q}^+ = \{1/1,\ \ 2/1, 1/2,\ \ 3/1, 2/2, 1/3,\ \ 4/1, 2/3, 3/2, 1/4,\ \ 5/1, 4/2, 3/3, 2/4, 1/5,\ \ldots\}$$

List elements in order of

- numerator+denominator
- breaking ties according to denominator

Only k numbers have total of k

Technique is called “dovetailing”

The Positive Rationals are Countable: Another Way

```
public static void enumerateQ() {
    for (nat sum=2; ; sum++) {
        for (nat p=1; p < sum; p++) {
            nat q = sum – p;
            System.out.println(new Rational(p, q));
        }
    }
}
```

We have to show that this function lists all positive rational numbers.
First, note that any positive fraction has a sum that is at least two.
Then, we want to show that for any sum s, the program reaches s. Note that the inner for loop runs for exactly s – 1 iterations, which is always finite. So, the program will eventually reach any sum.

Consider r = p/q. Note that the sum for this fraction is p + q. By the above, the program reaches this sum. Furthermore, since 1 < p < p +q, the inner loop prints out p/q.

Claim: Σ^* is countable for every finite Σ

```
public static void enumerateSigmaStar() {
    for (nat len=0; len < 3; len++) {
        printStringsOfLength(len, "");
    }
}

public static void printStringsOfLength(nat len, String s) {
    if (len == 0) {
        System.out.println(s);
        return;
    }
    for (char c : Sigma) {
        printStringsOfLength(len - 1, s + c);
    }
}
```

We must show that every string is printed. First, note that every string has a length. So, if we print out strings of every length, we've printed out all strings. Next, we show that printStringsOfLength(n, s) prints all strings of length n prefixed by s. We go by induction.

BC (n=0): The empty string is the only string of length 0; note that when len is 0, the function prints s; so, it prints s.

IH: Suppose the claim is true for some k >= 0.

IS: We know printStringsofLength(k – 1, s + c) prints all strings of length k – 1 prefixed by s + c. Since we loop through all possible values of c, these are the same strings as those of length k, prefixed by s.

The set of all Java programs is countable

If Σ = <all valid characters in java programs>, then the set of Java programs is a subset of Σ^*. Then, the listing for Σ^* from the previous slide prints all Java programs. Thus, the set of all Java programs is countable.

Georg Cantor

- Set theory
- Cardinality
- Continuum hypothesis

Is the set of real numbers countable?

Between any two real numbers there are an infinite number of others...

What about the real numbers?

Q: Is *every* set is countable?

A: Theorem [Cantor] The set of real numbers (even just between 0 and 1) is NOT countable

Proof is by contradiction using a new method called diagonalization…

Proof by Contradiction

- Suppose that $\mathbb{R}^{[0,1)}$ is countable
- Then there is some listing of all elements
  $$\mathbb{R}^{[0,1)} = \{ r_1, r_2, r_3, r_4, \ldots \}$$
- We will prove that in such a listing there must be at least one missing element which contradicts statement "$\mathbb{R}^{[0,1)}$ is countable"
- The missing element will be found by looking at the decimal expansions of r_1, r_2, r_3, r_4, …

Real Numbers between 0 and 1: $\mathbb{R}^{[0,1)}$

- Every number between 0 and 1 has an infinite decimal expansion:

$1/2 =$ 0.50000000000000000000000...

$1/3 =$ 0.33333333333333333333333...

$1/7 =$ 0.14285714285714285714285...

$\pi - 3 =$ 0.14159265358979323846264...

$1/5 =$ 0.19999999999999999999999...

$=$ 0.20000000000000000000000...

Representations of real numbers as decimals

Representation is unique except for the cases that decimal ends in all 0's or all 9's.

$$x = 0.1999999999999999999999999\ldots$$

$$10x = 1.999999999999999999999999\ldots$$

$9x = 1.8$ so

$x=0.20000000000000000000\ldots$

Won't allow the representations ending in all 9's

All other representations give different elements of $\mathbb{R}^{[0,1)}$

Supposed listing of $\mathbb{R}^{[0,1)}$

		1	2	3	4	5	6	7	8	9	...
r_1	0.	5	0	0	0	0	0	0	0	...	...
r_2	0.	3	3	3	3	3	3	3	3	...	...
r_3	0.	1	4	2	8	5	7	1	4	...	...
r_4	0.	1	4	1	5	9	2	6	5	...	...
r_5	0.	1	2	1	2	2	1	2	2	...	...
r_6	0.	2	5	0	0	0	0	0	0	...	...
r_7	0.	7	1	8	2	8	1	8	2	...	...
r_8	0.	6	1	8	0	3	3	9	4	...	...
...	....	...	....	....	...	...	...	...	...	...	

Supposed listing of $\mathbb{R}^{[0,1)}$

		1	2	3	4	5	6	7	8	9	...
r_1	0.	5	0	0	0	0	0	0	0	...	...
r_2	0.	3	3	3	3	3	3	3	3	...	...
r_3	0.	1	4	2	8	5	7	1	4	...	...
r_4	0.	1	4	1	5	9	2	6	5	...	...
r_5	0.	1	2	1	2	2	1	2	2	...	...
r_6	0.	2	5	0	0	0	0	0	0	...	...
r_7	0.	7	1	8	2	8	1	8	2	...	...
r_8	0.	6	1	8	0	3	3	9	4	...	...
...	....	...	....	....	...	...	...	...	...	...	

Flipped Diagonal

		1	2	3	4							
r_1	0.	5^{1}	0	0	0							
r_2	0.	3	3^{5}	3	3							
r_3	0.	1	4	2^{5}	8	5	7	1	4	...	...
r_4	0.	1	4	1	5^{1}	9	2	6	5	...	...
r_5	0.	1	2	1	2	2^{5}	1	2	2	...	...
r_6	0.	2	5	0	0	0	0^{5}	0	0	...	...
r_7	0.	7	1	8	2	8	1	8^{5}	2	...	...
r_8	0.	6	1	8	0	3	3	9	4^{5}	...	...
...	....	...	....	....	...	...	...	...	...	...$^{...}$	

Flipping Rule:

If digit is 5, make it 1

If digit is not 5, make it 5

Flipped Diagonal Number D

		1	2	3	4	5	6	7	8	9	…
D =	0.	1	5	5	1	5	5	5	5	…	

D is in $\mathbb{R}^{[0,1)}$

But for all **n**, we have **D**≠$\mathbf{r_n}$ since they differ on $\mathbf{n}^{th}$ digit (which is not **9**)

⇒ list was incomplete

⇒ $\mathbb{R}^{[0,1)}$ is not countable

The set of all functions $f : \mathbb{N} \to \{0,1,\ldots,9\}$ is not countable

The set of all functions $f : \mathbb{N} \to \{0,1,...,9\}$ is not countable

Suppose for contradiction that the set $S = \{f : (f : \mathbb{N} \to \{0,1,...,9\})\}$ is countable. Then, there exists a function $g : \mathbb{N} \to S$ that is surjective.

Construct a function $h : \mathbb{N} \to \{0,1,...,9\}$ as follows:

$$h(n) = 9 - g(n)(n)$$

Note that $h \in S$, because it is a function from $\mathbb{N} \to \{0,1,...,9\}$. We claim h is not in our listing. Consider $g(n)$. Note that $g(n)(n)$ is a number between 0 and 9; however, $9 - x \neq x$. So, $h \neq g(n)$. So, h is not in our listing.

This is a contradiction; so, it follows that S is uncountable.

Non-computable Functions

The set of all functions $f : \mathbb{N} \to \{0, 1, \ldots, 9\}$ is uncountable.

The set of all Java programs is countable.

There are INFINITELY many functions that uncomputable.

Back to the Halting Problem

- Suppose that there is a program **H** that computes the answer to the Halting Problem
- We will build a table with a row for each program (just like we did for uncountability of reals)
- If the supposed program **H** exists then the **D** program we constructed as before will exist and so be in the table
- But **D** must have entries like the "flipped diagonal"
  - **D** can't possibly be in the table.
  - Only assumption was that **H** exists. That must be false.

Some possible inputs **x**

programs **P**

	<P_1>	<P_2>	<P_3>	<P_4>	<P_5>	<P_6>	....					
P_1	**0**	1	1	0	1	1	1	0	0	0	1	...
P_2	1	**1**	0	1	0	1	1	0	1	1	1	...
P_3	1	0	**1**	0	0	0	0	0	0	0	1	...
P_4	0	1	1	**0**	1	0	1	1	0	1	0	...
P_5	0	1	1	1	**1**	1	1	0	0	0	1	...
P_6	1	1	0	0	0	**1**	1	0	1	1	1	...
P_7	1	0	1	1	0	0	**0**	0	0	0	1	...
P_8	0	1	1	1	1	0	1	**1**	0	1	0	...
P_9	. . . .											
.	. . . .											
.												

(P,x) entry is **1** if program **P** halts on input **x**
and **0** if it runs forever

Some possible inputs **x**

D behaves like **flipped diagonal**

programs **P**

	<P_1>	<P_2>	<P_3>	<P_4>	<P_5>	<P_6>	....					
P_1	0 1	1	1	0	1	1	1	0	0	0	1	...
P_2	1	1 0	0	1	0	1	1	0	1	1	1	...
P_3	1	0	1 0	0	0	0	0	0	0	0	1	...
P_4	0	1	1	0 1	1	0	1	1	0	1	0	...
P_5	0	1	1	1	1 0	1	1	0	0	0	1	...
P_6	1	1	0	0	0	1 0	1	0	1	1	1	...
P_7	1	0	1	1	0	0	0 1	0	0	0	1	...
P_8	0	1	1	1	1	0	1	1 0	0	1	0	...
P_9	.	.	. .	.	.	.	.	.	.	.	.	
.	.	.	. .	.	.	.	.	.	.	.	.	
.												

(P,x) entry is **1** if program **P** halts on input **x**
and **0** if it runs forever

recall: code for D assuming subroutine H that solves the halting problem

- Function **D(x)**:
  - if **H(x,x)=1** then
    - **while** (true); /* loop forever */
  - else
    - **no-op**; /* do nothing and halt */
  - endif

- If **D** existed it would have a row different from every row of the table
  - **D** can't be a program so **H** cannot exist!