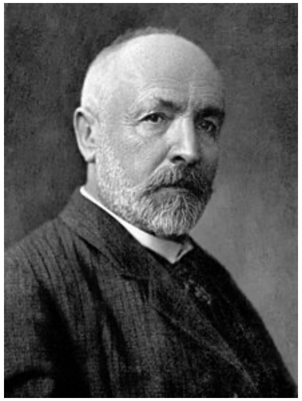
Undecidability



Warm-up question (4/12/22)

What is your favorite board or card game?

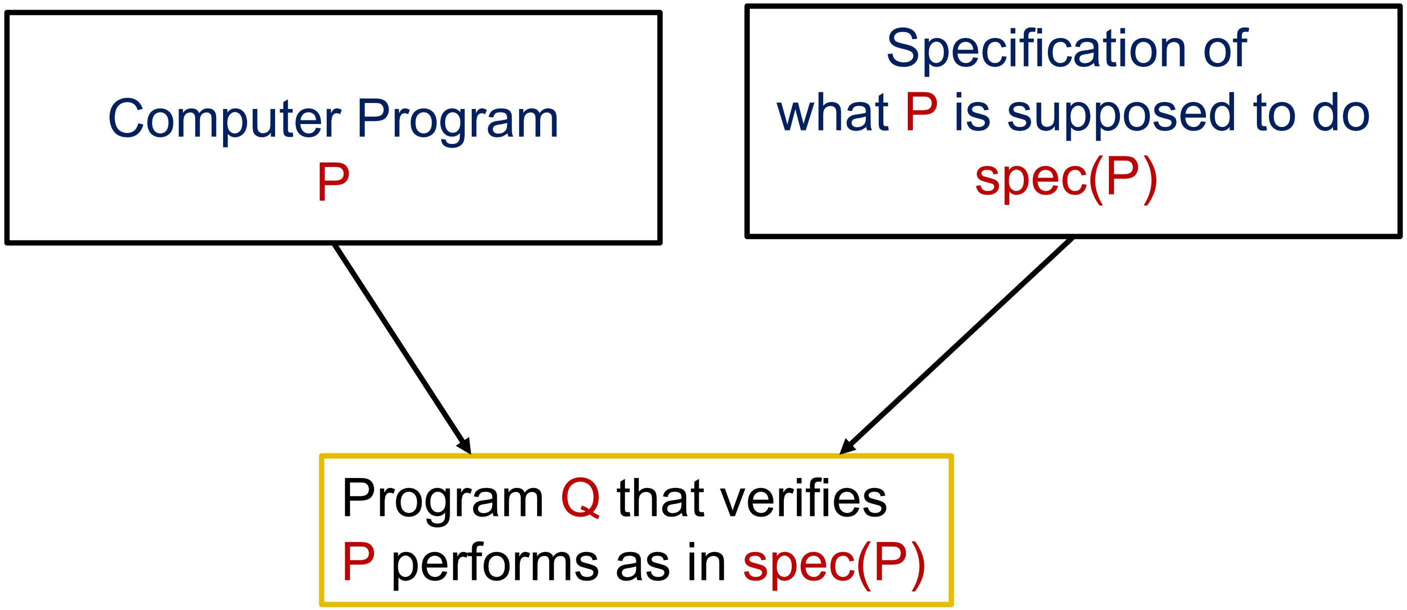
Send me your response by Canvas email.

Today (this week) we will prove

One of the most philosophically important theorems of the theory of computation:

There is a specific problem that is algorithmically unsolvable

Consider software verification



Computer Program
P

Specification of
what P is supposed to do
spec(P)

Program Q that verifies
P performs as in spec(P)

No Q exists!
That is, software verification is not solvable by computer

Turing Machine: Acceptance

Let:

A_{TM} = {<M,w> | M is a TM and M accepts w}

Theorem:

A_{TM} is undecidable

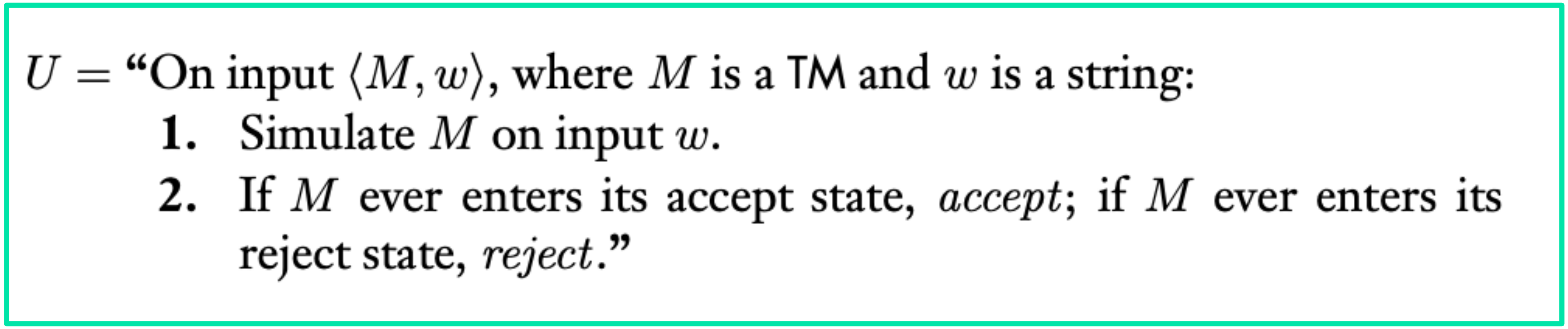
We will prove this theorem shortly, but first a “smaller” theorem:

A_{TM} is Turing-recognizable

Reminder:
A_{DFA} is decidable
A_{CFG} is decidable

Universal Turing Machine

The following Turing machine U recognizes A_{TM}



$U =$ "On input $\langle M, w\rangle$, where M is a TM and w is a string:

1. Simulate M on input w.
2. If M ever enters its accept state, *accept*; if M ever enters its reject state, *reject*."

- This machine loops on input <M,w> if M loops on w
- U is called ***universal TM*** because it is capable of simulating any other TM from the description of that machine

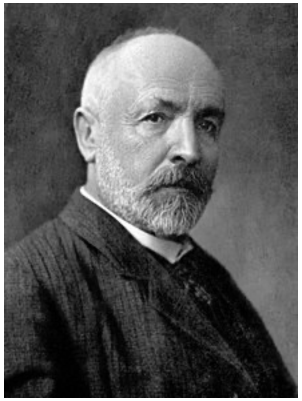
The diagonalization method

- The proof of the undecidability of A_{TM} uses a technique called *diagonalization*

- Diagonalization was discovered by mathematician Georg Cantor in 1873

- Cantor was concerned with the problem of measuring the sizes of infinite sets

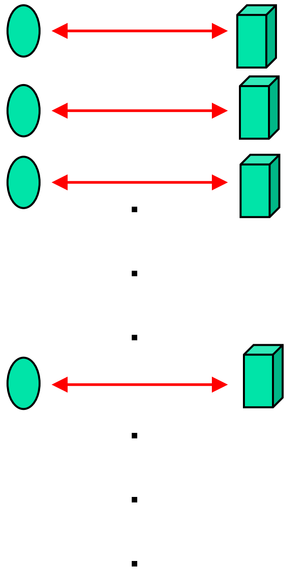
- If we have two infinite sets, how can we tell whether one is larger than the other or whether they are of the same size?



Georg Cantor
(1845—1918)

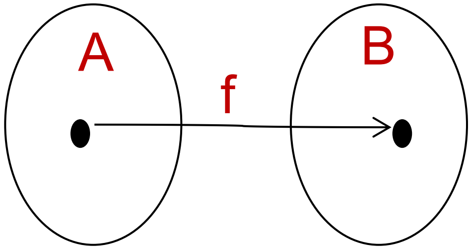
Counting elements of an infinite set…

- Cantor observed that two finite sets have the same size if the elements of one set can be paired with elements of the other set.
- The idea can be extended to infinite sets



Pairing: some technical definitions

Suppose we have sets A and B and
a function f from A to B.



f is **one-to-one**
if it never maps two different elements to the same place -- i.e., f(a) ≠ f(b) whenever a ≠ b

f is **on to**
if it hits every element of B -- i.e., for every b ϵ B there is an a in A such that f(a) = b

f is a **correspondence**
if it is both one-to-one and on to

Pairing: some technical definitions

- We say that sets A and B have the same size if there a correspondence f: A → B

- In a correspondence, every element of A maps to a unique element of B and each element of B has a unique element of A mapping to it

- A correspondence is simply a way of pairing the elements of A with the elements of B

Example

- Let:

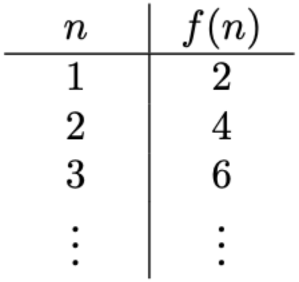
  N be the set of natural numbers {1, 2, 3, …}

  E be the set of even natural numbers {2, 4, 6,…}

- Using Cantor’s definition of size,

  N and E have the same size

- The correspondence f mapping N to E is simply f(n) = 2n



n	$f(n)$
1	2
2	4
3	6
$\vdots$	$\vdots$

- Intuitively, E seems smaller than N because E is a proper subset of N
- Yet, pairing each member of N with its own member of E is possible

Strange, but true!

Countable sets

- **Definition:**

  A set A is countable if either it is **finite** or it has the **same size** as the set of natural numbers N

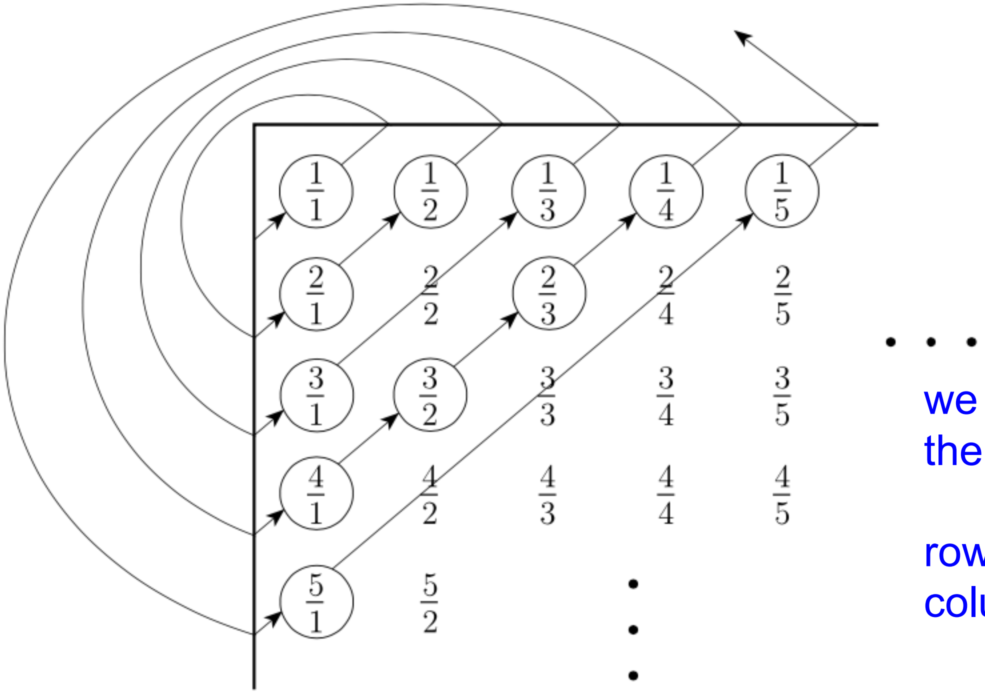
- **An even stranger example**:
  - Let Q = { m/n | m, n ϵ N}

    be the set of positive rational numbers
  - Q seems to be much larger than N,

    yet Q and N are of the *same size*!
  - Need to give a correspondence with N

    to show that Q is countable

Counting elements of Q



$\frac{1}{1}$	$\frac{1}{2}$	$\frac{1}{3}$	$\frac{1}{4}$	$\frac{1}{5}$
$\frac{2}{1}$	$\frac{2}{2}$	$\frac{2}{3}$	$\frac{2}{4}$	$\frac{2}{5}$
$\frac{3}{1}$	$\frac{3}{2}$	$\frac{3}{3}$	$\frac{3}{4}$	$\frac{3}{5}$
$\frac{4}{1}$	$\frac{4}{2}$	$\frac{4}{3}$	$\frac{4}{4}$	$\frac{4}{5}$
$\frac{5}{1}$	$\frac{5}{2}$			

we make an **infinite matrix** containing all the positive rational numbers (Q)

row i has all numbers with numerator i
column j has all numbers with denominator j

we turn this matrix into a list by going along the **diagonals** as shown in the picture

Uncountable sets

- For some infinite sets,

  no correspondence with N exists
- Such sets are called **uncountable**
- The set of real numbers R is an example of an uncountable set
- Cantor proved that R **is uncountable**
- In doing so, he introduced the

  diagonalization method

Proving R is uncountable

- To show that R is uncountable, we show that *no correspondence* exists between N and R
- The proof is by *contradiction*
- Suppose a correspondence f existed between N and R
- Our job is to show that f *fails to work* as it should
- For it to be a correspondence, f must pair all the members of N with all the members of R
- But we still find an x in R that is not paired with anything in N, which will be a contradiction

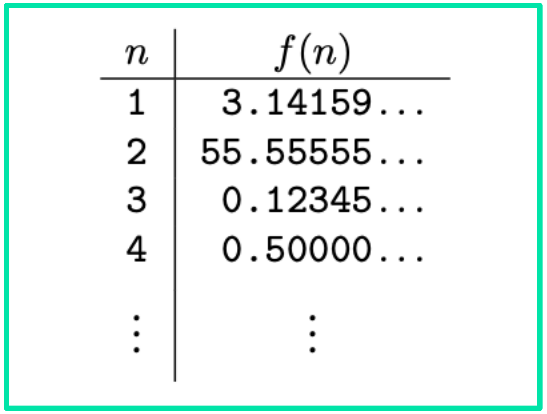
Finding x

- We find this x by actually *constructing* it

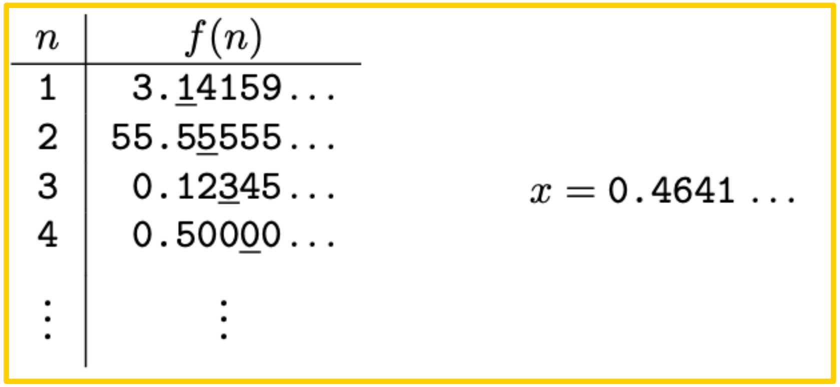
- We choose each digit of x to make x *different from one* of the real numbers that is paired with an element of N

- In the end, we are sure that x is different from *any* real number that is paired

Illustration for a construction of x



n	$f(n)$
1	3.14159...
2	55.55555...
3	0.12345...
4	0.50000...
⋮	⋮



n	$f(n)$
1	$3.\underline{1}4159\ldots$
2	$55.5\underline{5}555\ldots$
3	$0.12\underline{3}45\ldots$
4	$0.500\underline{0}0\ldots$
⋮	⋮

$x = 0.4641\ldots$

- We construct the desired x by giving its decimal representation.
- Our objective is to ensure that x ≠ f(n) for any n.
- To ensure that x ≠ f(1), we let the first digit of x be anything different from the first fractional digit of f(1) = 3.14159…Arbitrarily, let it be 4.
- To ensure that x ≠ f(2), we let the second digit of x be anything different from the second digit of f(2) = 5.5555…Arbitrarily, let it be 6.
- Continue in this fashion…

Application to theory of computation

- The theorem that R is uncountable – and its proof using the diagonalization method – has an important application to the theory of computation
- It shows that some languages are not decidable or even Turing recognizable because there are uncountably many languages yet countably many Turing machines
- Because each TM can recognize a single language and there are more languages than TMs, some languages are not recognized by any TM.
- Next lecture, we will see a proof for the statement that some languages are not Turing-recognizable