This statement is false.

CS310

The Halting Problem

Section 4.2

November 19, 2008

Some material from:

Introducing the Theory of Computation, Goddard

Will it stop?

- Goldbach's Conjecture
  - Every even integer at least 4 is the sum of two primes
    - 4 = 2 + 2
    - 6 = 3 + 3
    - 8 = 5 + 3
    - 100 =
  - A TM looking for a counterexample may never halt

Self Denial (and reference)

$\mathbf{S}_{\mathbf{elf}}$ = { <M> | M is a TM that does not accept <M>}

– Can we build a machine that accepts that language $\mathbf{S}_{\mathbf{elf}}$?

Will it ever stop?

- A_{TM} = { <M, w> | M is a TM and M accepts w }
  - undecidable
  - U is a Universal TM
    - Capable of simulating any other TM from the description of that machine
  - TM U recognizes A_{TM}:
    - 1. Simulate M on input w with U
    - 2. If M accepts then U accepts; if M rejects then U rejects; *if M never halts then U never halts*
    - If we could get U to halt, then we could get M to halt

Why do we care?

- There is some specific problem, the Halting problem, which is *algorithmically unsolvable*!

- Software verification: does the software satisfy the requirements?
  - not *algorithmically solvable*!
  - (In general, for all software)

Further limits

- Some languages are not TM recognizable
  - show that the set of all TMs ***is smaller than*** the set of all languages
  - How many TMs are there?
  - How many Languages are there?

Counting

- Diagonalization
  - how can we determine if two infinite sets are the same size? (Georg Cantor)
  - cannot just count them up
  - the two sets are the same size if the elements of one set can be paired with the elements from the other set (no counting!)
  - define a function as a ***correspondence***

Correspondence

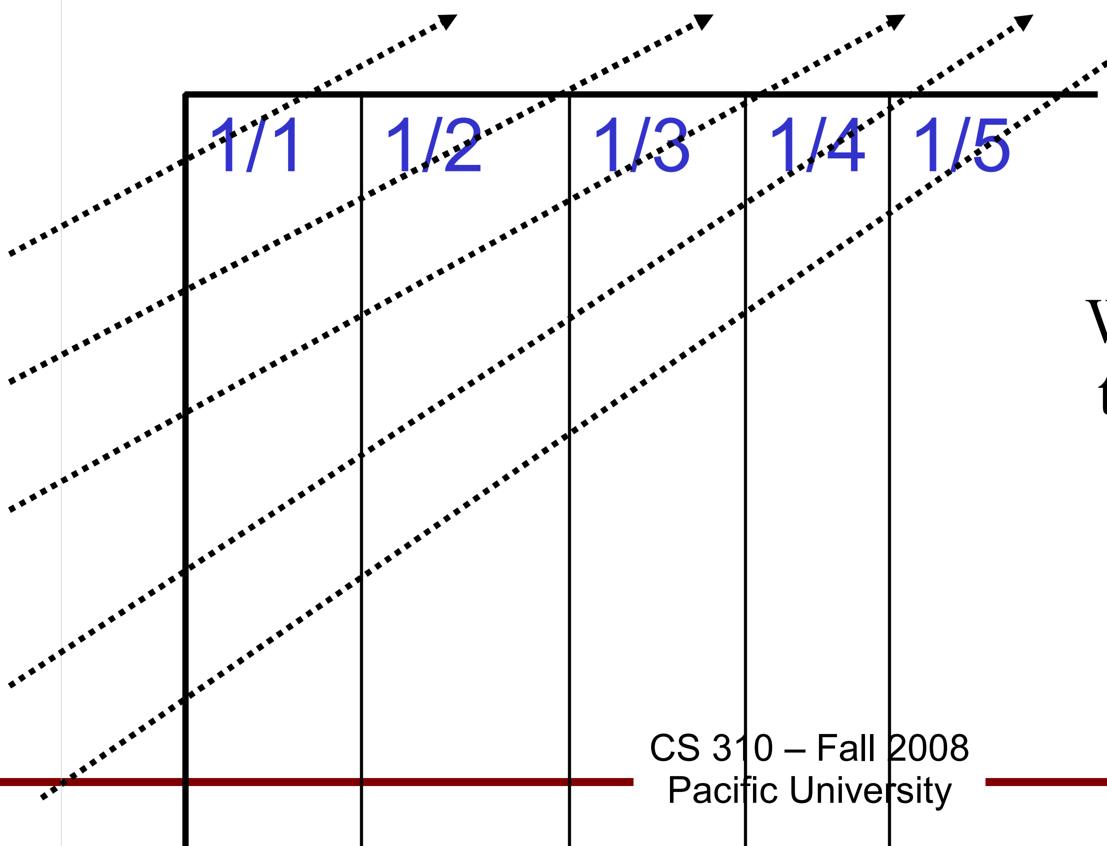
- A and B are sets, F is a function from A to B; $F: A \rightarrow B$
  - Fis *one-to-one* if it never maps two different elements to the same place, if $F(a) \neq F(c)$ whenever $a \neq c$
  - F is *onto* if it hits every element of B, for each $b \in B$ this is an $a \in A$ such that $F(a) = b$
  - A and B are the *same size* if there is a one-to-one, onto function F
  - F is a correspondence

Application

- Let N be the set of natural numbers, let E be the set of even natural numbers.
- If we can find a correspondence function between these two infinite sets, the are the same size
  - $f(n) = 2n$
- Definition: a set is *countable* if it is finite or in correspondence with the set of natural numbers

Diagonialization

- Is Q = { m/n | m,n $\in$ N } countable?
  - can we find a correspondence?
  - **We can make a list** of all the elements in Q, and match them with the elements in N



We cannot just go down the first row because…

What could ever be uncountable?

- The set of Real Numbers, R
- Proof by contradiction
  - assume R is countable
  - there must exists a correspondence function f with the set N
  - find some number $x \in$ R that is not paired with a number $p \in$ N
  - we will construct this number x

Real Numbers are uncountable

p	f(p)
1	3.14159…
2	5.55555
3	0.1234…
p	f(p)

- Assume f() exists
- Construct x such

x ≠f(p) for any p

- x is between 0 and 1
- ensure x ≠f(1), set the 10ths' place to 4
- ensure x ≠f(2), set the 100ths' place to 6
  - forever….
  - never select 0 or 9 since .1999… = .2000*
- we know x ≠f(p) for any p since x differs from f(p) in the pth decimal place

*On the final, prove this for 2 points extra credit

Some languages are not TM recog.

- show that the set of all TMs is countable
  - the set of all TM is countable because each TM, M, can be encoded into a string, <M>
  - omit all strings that are not valid TMs
- show that the set of all languages is not
  - set of all infinite binary sequences, B, is uncountable, using proof by contradiction similar to Real Numbers

Encode TM as string

- Assume $\Sigma = \{0, 1\}$; $\Gamma = \{0, 1, \triangledown\}$
- Encode elements of δ using 1s

  $\delta(q_i, x) = (q_j, y, M)$ is

  - $en(q_i)0en(x)0en(q_j)0en(y)0en(M)$

  – two 0s separate transitions,

  beginning and end marked with 000

  q_0 is start

  q_1 is accept

  q_{n-1} is reject

Z	en(Z)
0	1
1	11
$\triangledown$	111
Z	en(Z)

- We could build a TM to check to see if a string is a legal encoding of a deterministic TM
  – what does that language look like?

The Halting Problem, Proof

- A_{TM} = { <M, w> | M is a TM and M accepts w }
  - undecidable, may never halt
  - assume A_{TM} is decidable and that H is a TM decider (always halts) for A_{TM}
  - on input <M,w>:

$$H(\langle M,w\rangle) \begin{cases} \text{accept if M accepts w} \\ \text{rejects if M } \textit{does not accept } \text{w} \end{cases}$$

The Halting Problem, Proof, cont.

- Construct a TM, D, with H as subroutine.
- D calls H to determine what M does when input is its encoding. Once D determines, it does the opposite.
  - D = On input <M>, where M is a TM
    1) Run H on <M, <M>>
    2) If H accepts, reject. If H rejects, accept.

$$\textbf{D(<D>)} \begin{cases} \text{accept if D } \textit{does not accept} \text{ <D>} \\ \text{reject if D accepts <D>} \end{cases}$$

Contradiction! We can use diagonalization to explore this further