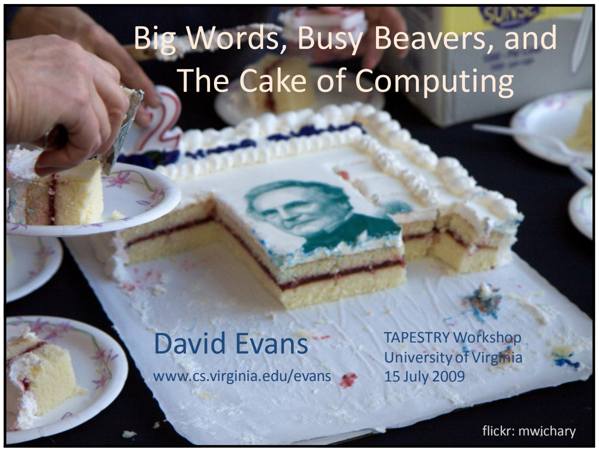
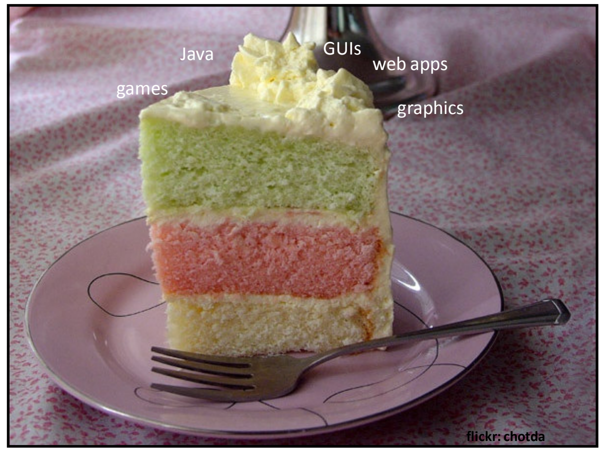
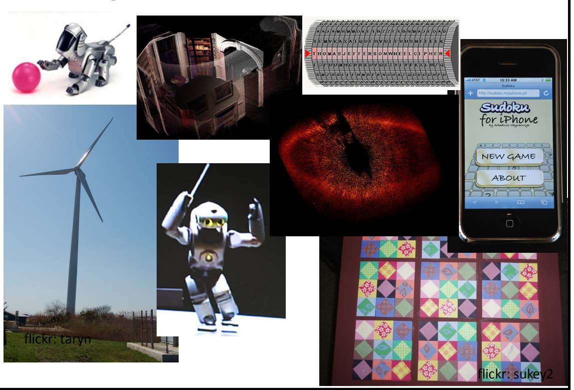




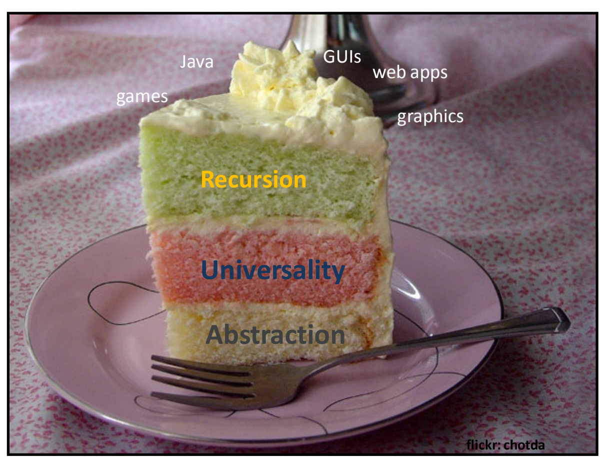
Frosting: *Doing Cool Stuff with Computers*



NCAA Definition

The current core-curriculum areas were legislated in the early 1980's. At that time, computer-science courses were programming based and academic in nature. In today's secondary education environment, the vast majority of computer courses no longer contain programming elements but teach life skills, such as the use of a desktop computer and software applications. **Although these software and keyboarding skills may be beneficial to college-bound students, they are not academic in nature.** … It should be noted that computer courses that include a significant element of programming might be encompassed in the mathematics-curriculum requirement.

Revision of NCAA Eligibility Requirements, August 2005



Recursive Definitions

What's the longest word in the English language?

Longest Words?

honorificabilitudinitatibus (27 letters, longest by Shakespeare)
  With honor.
antidisestablishmentarianism (28 letters)
  Movement against division of church and state.
hippopotomonstrosesquipedaliophobia (35 letters)
  Fear of long words.
pneumonoultramicroscopicsilicovolcanoconiosis (45 letters)
  (longest word in most dictionaries)
  Lung disease contracted from volcanic particles.

Like all words, these words are "made up".

Making Longer Words

antihippopotomonstrosesquipedaliophobia
  Against the fear of long words.

antiantihippopotomonstrosesquipedaliophobia
  Against a thing against the fear of long words.

Language is *Recursive*

No matter what word you think is the longest word, I can always make up a longer one!

word ::= **anti-***word*

By itself, this definition of *word* is circular.

Zero, One, Infinity

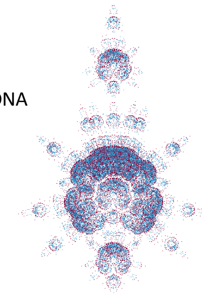
word ::= **anti-***word* This rule can make 0 words.

word ::= **hippopotomonstrosesquipedaliophobia** This rule can make 1 word.

word ::= **anti-***word*
word ::= **hippopotomonstrosesquipedaliophobia**
These two rules can make infinitely many words, enough to express all ideas in the universe!

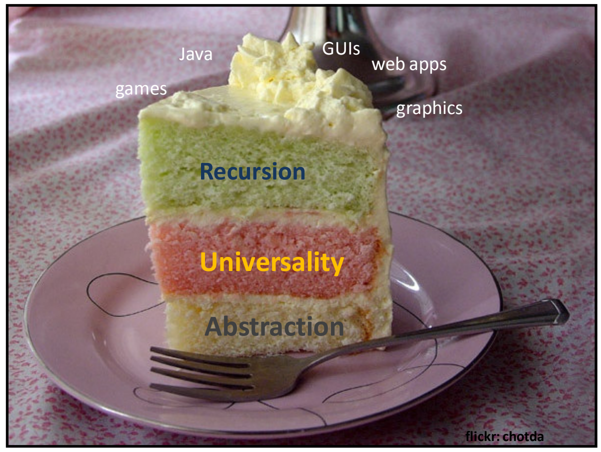
Recursive Definitions *Everywhere*

- Language
  - Words, Sentences, Structures
- Nature
  - **Plant Growth**, Quantum Physics, DNA
- Mathematics
  - Numbers, Arithmetic Algorithms
- Music
  - Harmony, structure
- Computing
  - Data, procedures



Red Hot and Blue by Paul DiOrio, Rachel Phillips

Wes Weimer may talk more about this tomorrow!

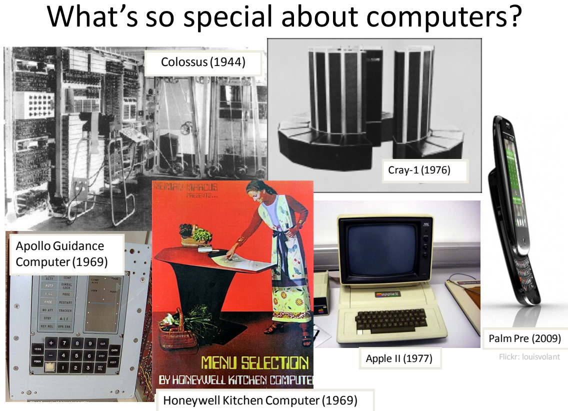


Biggest Number Game

- When I say "GO", write down the biggest number you can in 30 seconds.
- Requirement:
  - Must be an exact number
  - Must be defined mathematically
- Biggest number wins!

Countdown Clock

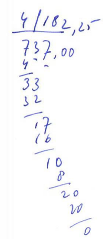
What's so special about computers?



Toaster Science?

“Computers” before WWII

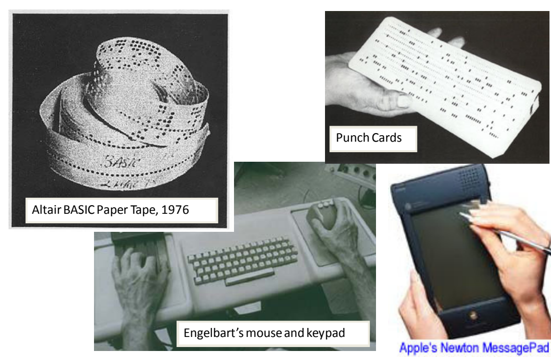
Mechanical Computing



Modeling Computers

- Input
  - Without it, we can’t describe a problem
- Output
  - Without it, we can’t get an answer
- Processing
  - Need some way of getting from the input to the output
- Memory
  - Need to keep track of what we are doing

Modeling Input

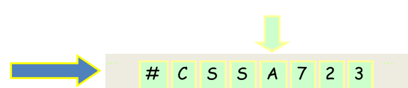


Turing’s Model

“Computing is normally done by writing certain symbols on paper. We may suppose this paper is divided into squares like a child’s arithmetic book.”

Alan Turing, *On computable numbers, with an application to the Entscheidungsproblem*, 1936

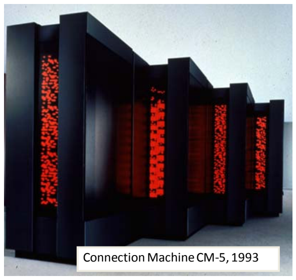
Modeling Pencil and Paper



How long should the tape be?

Modeling Output

- Blinking lights are cool, but hard to model
- Use the tape: output is what is written on the tape at the end



Connection Machine CM-5, 1993

Modeling Processing (Brains)

Look at the current state of the computation

Follow simple rules about what to do next

Modeling Processing

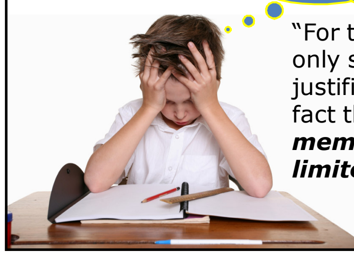
- Evaluation Rules
  - Given an input on our tape, how do we evaluate to produce the output
- What do we need:
  - Read what is on the tape at the current square
  - Move the tape one square in either direction
  - Write into the current square

				0	0	1	1	0	0	1	0	0	0					

Is that enough to model a computer?

Modeling Processing (Brains)

•Follow simple rules
•Remember what you are doing



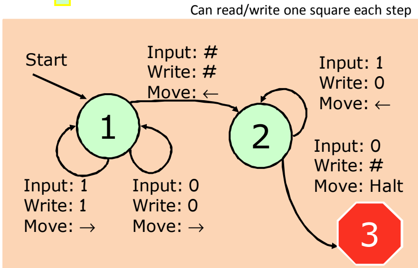
"For the present I shall only say that the justification lies in the fact that the ***human memory is necessarily limited***."

Alan Turing

Turing's Model: Turing Machine

#	1	0	1	1	0	1	1	1	0	1	1	0	1	1	1	#

Infinite Tape: Finite set of symbols, one in each square
Can read/write one square each step

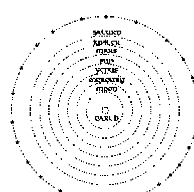


Controller:
Limited (finite) number of states

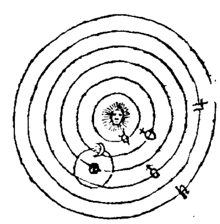
Follow rules based on current state and read symbol

Write one square each step, move left or right or halt, change state

What makes a good model?



Ptolomy



Copernicus

$F = GM_1M_2 / R^2$

Newton

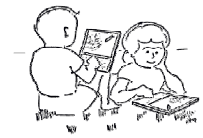
Questions about Turing's Model

- How well does it match "real" computers?
  - Can it do everything they can do?
  - Can they do everything it can do?
- Does it help us understand and reason about computing?

Church-Turing Thesis

- All mechanical computers are equally powerful*

  *Except for practical limits like memory size, time, display, energy, etc.
- There exists some Turing machine that can simulate *any* mechanical computer
- *Any* computer that is powerful enough to simulate a Turing machine, can simulate any mechanical computer



Power of Turing Machine

- Can it add?
- Can it carry out any computation?
- Can it solve any problem?

Performing Addition

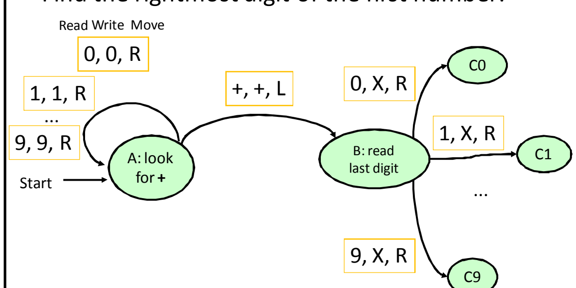
- **Input:** a two sequences of digits, separated by + with # at end.

  e.g., # 1 2 9 3 5 2 + 6 3 5 9 4 #
- **Output:** sum of the two numbers

  e.g., # 1 9 2 9 4 6 #

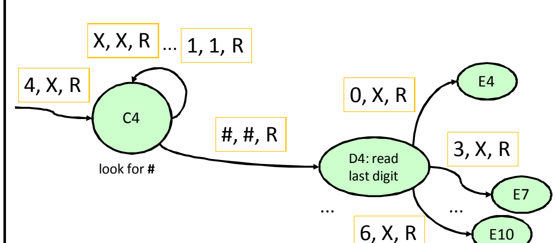
Addition Program

Find the rightmost digit of the first number:



Addition, Continued

Find the rightmost digit of the second number:



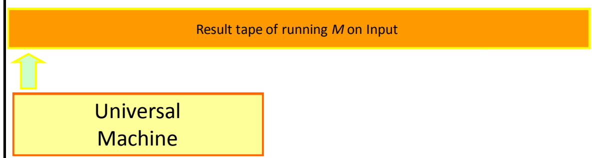
Must duplicate this for each first digit – states keep track of first digit!

Power of Turing Machine

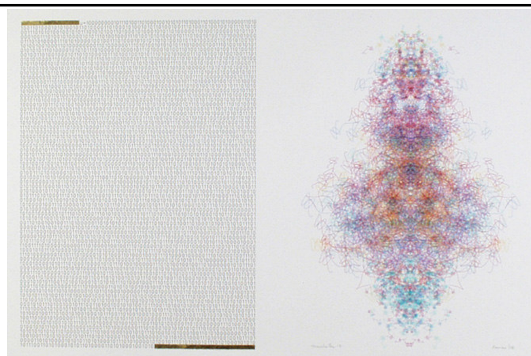
✓ Can it add?

- Can it carry out any computation?
- Can it solve any problem?

Universal Machine

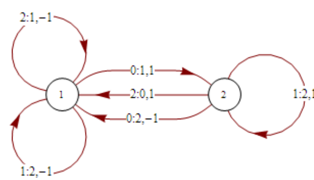


A Universal Turing Machine can simulate any Turing Machine running on any Input!



Manchester Illuminated Universal Turing Machine, #9
from http://www.verostko.com/manchester/manchester.html

Universal Computing Machine



2-state, 3-symbol Turing machine proved universal by Alex Smith in 2007

What This Means

- Your cell phone, watch, iPod, etc. has a processor powerful enough to simulate a Turing machine
- A Turing machine can simulate the world's most powerful supercomputer
- Thus, your cell phone can simulate the world's most powerful supercomputer (it'll just take a lot longer and will run out of memory)

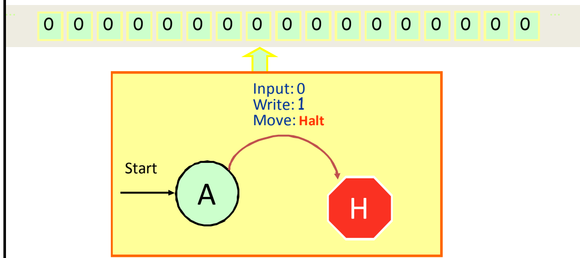
Are there problems computers can't solve?

The "Busy Beaver" Game

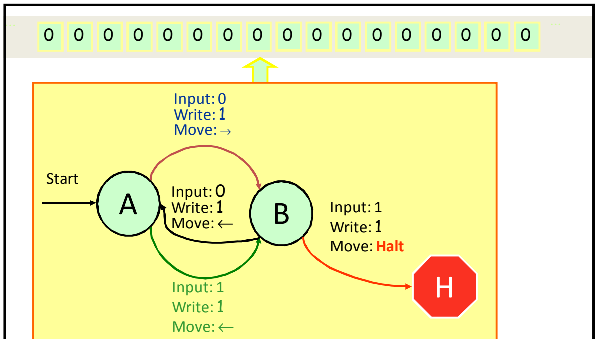
- Design a Turing Machine that:
  - Uses two symbols (e.g., "0" and "1")
  - Starts with a tape of all "0"s
  - Eventually halts (can't run forever)
  - Has *N* states
- Goal: machine runs for as many steps as possible before **eventually** halting

Tibor Radó, 1962

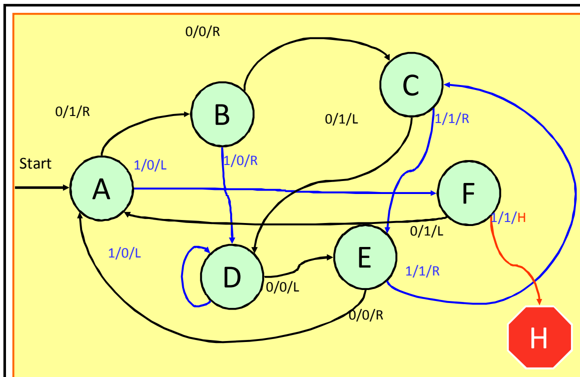
Busy Beaver: N = 1



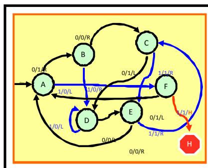
BB(1) = 1 Most steps a 1-state machine that halts can make



BB(2) = 6



6-state machine found by Buntrock and Marxen, 2001



Best found before 2001, only 925 digits!

In Dec 2007, Terry and Shawn Ligocki beat this: 2879 digits!

3002327716523562828955103018341340185147754337246752500373381801735214240760383265881912082978202876698984017860713458482804223834928227160518485855836681537972514386185617302094154876855700785386587573048574872220400307698440450988713670876150791383110343531646410779192098908371644773632893742255319551260232511722590345701550873036836546308741559908225161299384258306913786072736707081901605255340770400392265930739979231701547753586298504217125133785270862231126806779737517900329375785200176667922468399088559203629337677447608701284468834554778063164916018557844268607690279445427980061526931674528213366899174608861064865741890154011940348575777182530655416326563343142423255924867001185067165813034232717489654261604097971730737166888272814359046394456059281752540483211093060024746589681087933819123818123362279928399308330859334788531765747027760628582891565683922959635862636541393838567647280513949655544096884565781227432963199608083680945364210391495849467580065091609857013289970263017087602355002395981194105921426216696145528272444292174164654943638916971139653168926606117092900485806775661787157523545940490167192780698328665223329235413702930596679960013193766985516838488514746251520945671106154519868398944908856870822449787745514532043585886615939797639351028965232958039400236732031017449865507324968504369997537113430673286761581462692927233756620156128269241054548496584109615740312114406110889753498991567148886819523660180862466877120985530770548253674340626717567600703889221174349326334447731387837140237358987127902782883771982603800651050757929252394534506229992082975795848934488862781276290441632922518154100535222460845527615133839346231290832669493773809504666431216897465119968476812750763132O6

(1730 digits)

Busy Beaver Numbers

BB(1) = 1
BB(2) = 6
BB(3) = 21
BB(4) = 107
BB(5) = Unknown!

Best so far is 47,176,870

BB(6) > 10^{2879}

Discovered 2007

Winning the "Biggest number" game: BB(BB(BB(BB(111111111))))

Computing Busy Beaver Numbers

- Input: N (number of states)
- Output: BB(N)
  - The maximum number of steps a Turing Machine with N states can take before halting

Is it possible to design a Turing Machine that solves the Busy Beaver Problem?

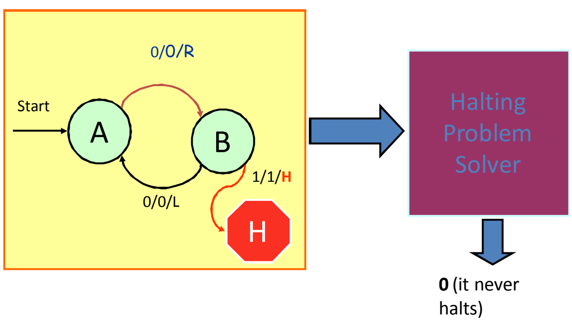
The Halting Problem

- Input: a description of a Turing Machine
- Output: “1” if it eventually halts, “0” if it never halts, starting on a tape full of “0”s.

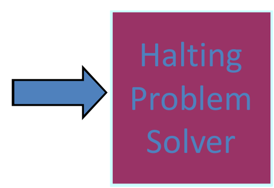
Is it possible to design a Turing Machine that **solves** the Halting Problem?

“Solves” means for all inputs, the machine finishes and produces the right answer.

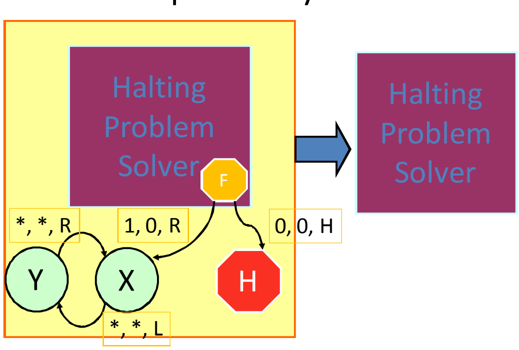
Example



Example



Impossibility Proof!



Impossible to make Halting Problem Solver

- If it outputs “0” on the input, the input machine would halt (so “0” cannot be correct)
- If it outputs “1” on the input, the input machine never halts (so “1” cannot be correct)

If it halts, it doesn’t halt!
If it doesn’t halt, it halts!

Busy Beaver is Impossible Too!

- If you could solve it, could solve Halting Problem:
  - Input machine has N states
  - Compute BB(N)
  - Simulate input machine for BB(N) steps
  - If it ever halts, it must halt by now
- ... but we know that is impossible, so it must be impossible to computer BB(N)

The BB numbers are so big you can't even compute them!

Recap

- A *computer* is something that can carry out well-defined steps:
  - Read and write on scratch paper, follow rules, keep track of state
- All computers are equally powerful
  - If a machine can simulate any step of another machine, it can simulate the other machine (except for physical limits)
  - What matters is the *program* that defines the steps

You can have your frosting and eat cake too!

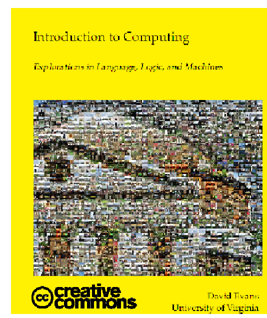
Questions

David Evans

evans@cs.virginia.edu

Some Sources:
Matthias Felleisen, Shriram Krishnamurthi , *Why Computer Science Doesn't Matter*, Communications of the ACM July 2009.

Scott Aaronson, *Who Can Name the Bigger Number?*, http://www.scottaaronson.com/writings/bignumbers.html



http://www.computingbook.org/