Curing Cancer with Your Cell Phone: Why *all* Sciences are Becoming Computing Sciences

David Evans
http://www.cs.virginia.edu/evans

Computer Science = Doing Cool Stuff with Computers?

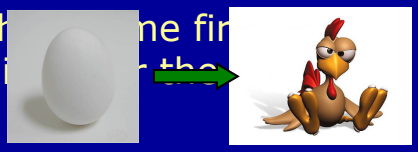
Toaster Science = Doing Cool Stuff with Toasters?

Computer Science

- Mathematics is about declarative ("what is") knowledge; Computer Science is about imperative ("how to") knowledge
- The Study of Information Processes
  - How to describe them — Language
  - How to predict their properties — Logic
  - How to implement them quickly, cheaply, and reliably — Engineering

Most Science is About Information Processes

Wh ne fir
chi r th



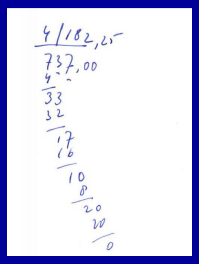
How can a (relatively) simple, single cell turn into a chicken?

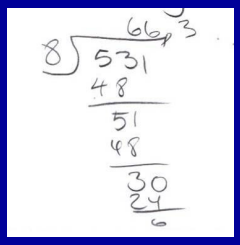
Agenda

- Three Big Ideas:
  - All Computers are Equally Powerful
  - Programs are Data, Data are Programs
  - Many Surprisingly Different Problems are Equally Difficult
- One Open Question
  - Is a machine that can always guess correctly able to solve problems a normal machine can't?

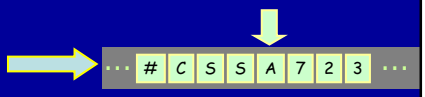
"Computers" before WWII

Mechanical Computing





Modeling Pencil and Paper



How long should the tape be?

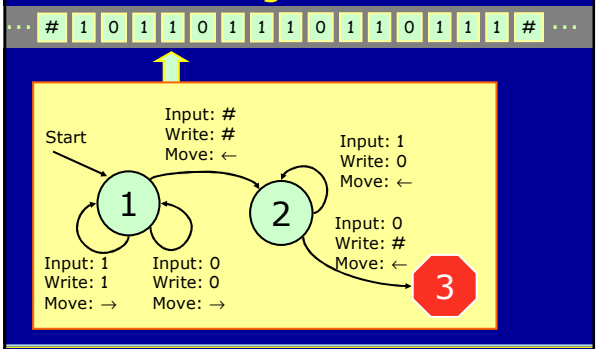
"Computing is normally done by writing certain symbols on paper. We may suppose this paper is divided into squares like a child's arithmetic book."
Alan Turing, *On computable numbers, with an application to the Entscheidungsproblem*, 1936

Modeling Brains

•Rules for steps
•Remember a little

"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



Universal Machine

Result tape of running *M* on Input

Universal Machine

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

Church-Turing Thesis

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

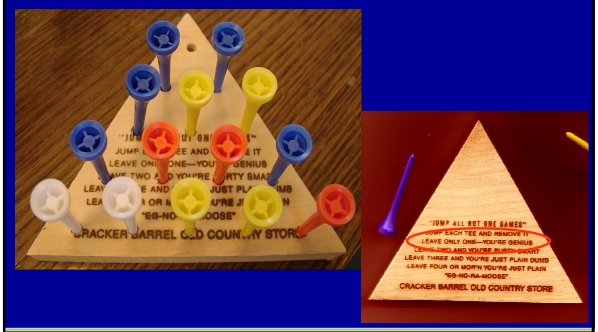
What This Means

- 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)

Recap

- All Computers are Equally Powerful
- Programs are Data, Data are Programs
- Many Problems are Equally Difficult
  - But no one knows how difficult!

A "Hard" Problem?



Generalized Pegboard Puzzle

- Input: a configuration of n pegs on a cracker barrel style pegboard (of any size)
- Output: if there is a sequence of jumps that leaves a single peg, output that sequence of jumps. Otherwise, output **false**.

Is this a "hard" problem?

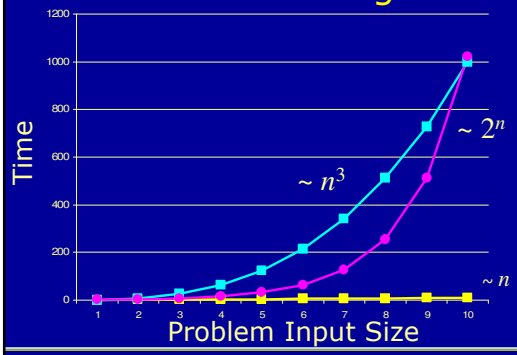
Solving Problems

- A solution to a *problem instance*: given a pegboard configuration, here's the sequence of jumps
- A solution to a *problem*: a procedure that (1) always finds the correct answer, and (2) always finishes.

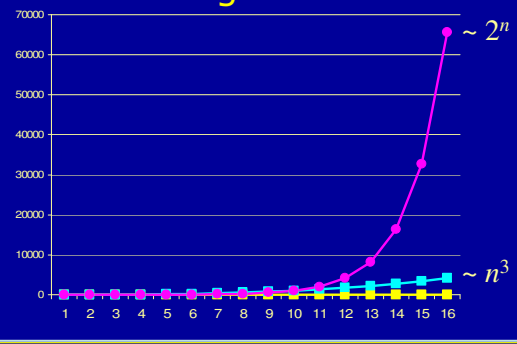
"Brute Force" Solvers

- Enumerate all possible answers
  - Every possible sequence of jumps
- Try them all until you find one that works
  - Simulate the jumps
- This works for almost all problems!
- Problem: how long does it take?

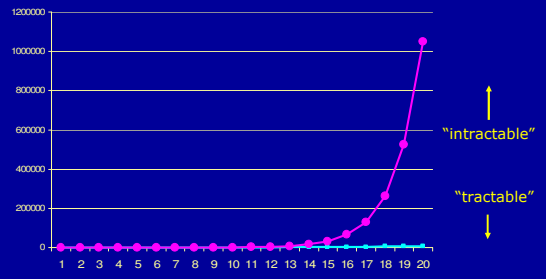
Problem Solving Time



Increasing Problem Size

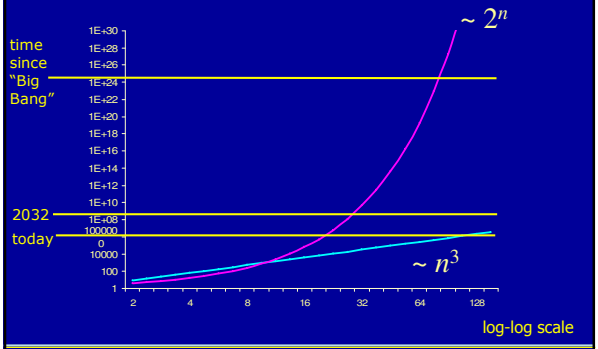


Tractable and Intractable Problems



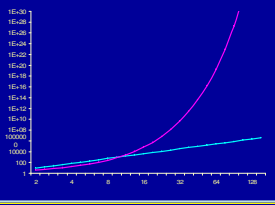
I do nothing that a man of unlimited funds, superb physical endurance, and maximum scientific knowledge could not do.
– Batman (may be able to solve intractable problems, but computer scientists can only solve tractable ones for large *n*)

This makes a huge difference!



Back to the Pegboard...

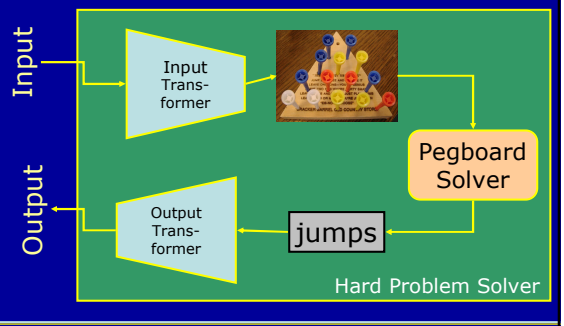
- A brute force solution is easy...but on the pink line
- Is there a *tractable* solution?



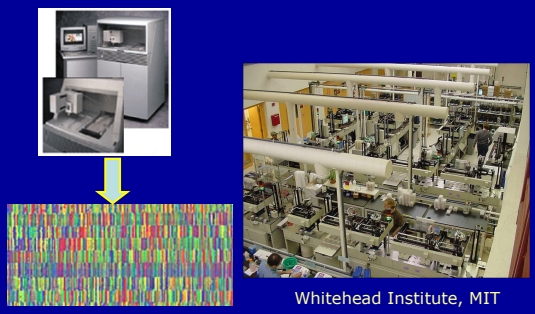
Deciding a Problem Is Hard

- "I tried really hard and still couldn't solve it."
  - Maybe the speaker isn't smart enough
  - Maybe a few days more effort will find it
- "Lots of really smart people tried really hard and no one could solve it."
- "It seems sort of like this other problem that we think is hard..."

Reduction



Reading the Genome



Whitehead Institute, MIT

Gene Reading Machines

- One read: about 700 base pairs
- But…don't know where they are on the chromosome

Read 3 TACCCGTGATCCA

Read 2 TCCAGAATAA

Read 1 ACCAGAATACC

Actual Genome ACCAGAATACCCGTGATCCAGAATAA

Genome Assembly

Read 1 ACCAGAATACC

Read 2 TCCAGAATAA

Read 3 TACCCGTGATCCA

Input: Genome fragments (but without knowing where they are from)

Ouput: The full genome

Genome Assembly

Read 1 ACCAGAATACC

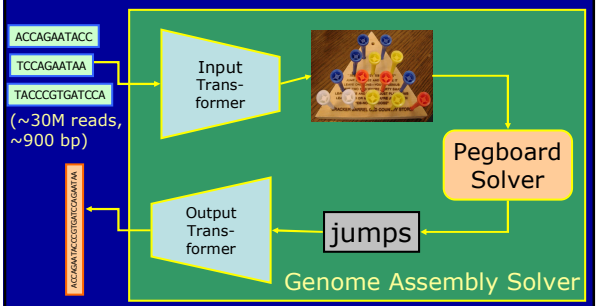
Read 2 TCCAGAATAA

Read 3 TACCCGTGATCCA

Input: Genome fragments (but without knowing where they are from)

Ouput: The *smallest* genome sequence such that all the fragments are substrings.

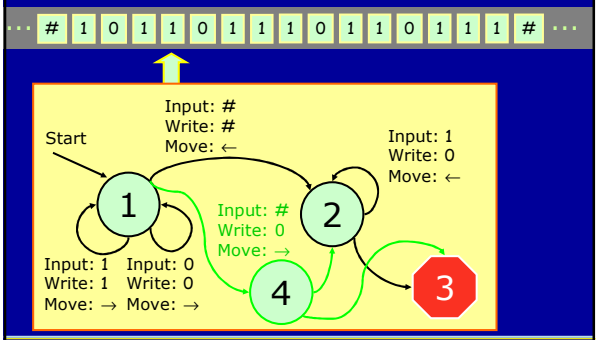
Genome Assembly Solver



What This Means

- We already know the shortest common superstring (genome assembly) problem is "hard"
- The pegboard problem must also be hard, since we could use a solver for it to solve the genome assembly problem
  - Requires: we can build fast transformers that don't increase the problem size exponentially

Non-Deterministic Machines

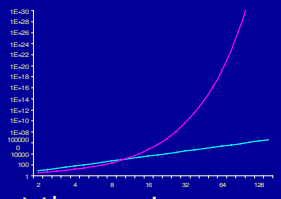


Non-Deterministic Machine

- Everytime there is a choice, it can *guess* the correct choice without looking ahead
- If we had such a machine, solving Pegboard (or Genome Assembly, etc.) problem would be easy:
  - It can guess the solution one step (alignment) at a time

Big Open Question

Is a non-deterministic machine able to solve problems that are intractable on a deterministic machine?



Seems obvious that the magic guess correctly ability should be useful...but no one knows for sure!

Recap

- "P vs NP" problem (one of the millennium prize problems)
- Solving the pegboard puzzle is equivalent to solving genome assembly
- With a non-deterministic machine, we could solve both
- With a mechanical computer, we don't know if a tractable solution exists (but can't prove it doesn't): We don't know if checking a solution is really easier than finding it

Summary

- Computer Science is the study of information processes: all about problem solving
- Many seemingly paradoxical results:
  - All Computers are Equally Powerful!
  - Many Surprisingly Different Problems are Equivalent!
- And seemingly obvious open problems:
  - Is checking a solution is really easier than finding it?

Computer Science at UVa

- New *Interdisciplinary Major in Computer Science* for A&S students (approved last year)
- Take CS150 this Spring
  - Every scientist needs to understand computing, not just as a tool but as a *way of thinking*
- Lots of opportunities to get involved in research groups

My Research Group

- Computer Security: computing in the presence of *adversaries*
- Recent student projects:
  - Proof that the Pegboard puzzle is hard (Mike Peck and Chris Frost)
  - Disk-level virus detection (Adrienne Felt)
  - Web Application Security (Sam Guarnieri)
  - N-Variant Systems: run variants of a program simultaneously (Sean Talts)

Questions

http://www.cs.virginia.edu/evans
evans@cs.virginia.edu