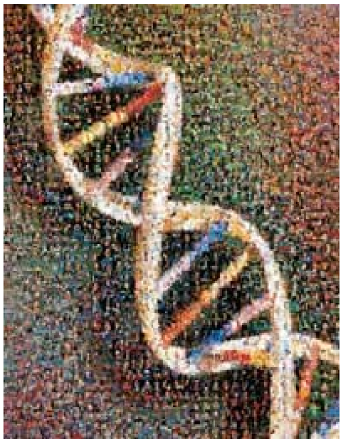
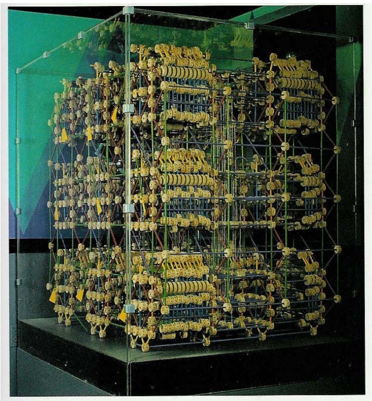
Glue, Photons, P=NP?



DNA Helix Photomosaic from cover of *Nature*, 15 Feb 2001 (made by Eric Lander)



Color Plate 1 Early model of the Tinkertoy computer, on display at The Computer Museum, Boston, also plays tic-tac-toe.

One-Slide Summary

- The **lambda calculus** is a universal, fundamental model of computation. You can view it as “the essence of Scheme”. It contains terms and rules describing variables, function abstraction, and function application.
- It is possible to **encode** programming concepts, such as true, false, if, numbers, lists, etc., in the lambda calculus. Lambda calculus can simulate Turing machines.
- **Quantum computers** and **non-determinsitic** Turing machines can try many options at once. They are **not more powerful** than normal Turing machines.
- The **Complexity Class P** contains tractable problems that can be solved in polynomial time. The **Complexity Class NP** contains problems for which solutions can be verified in polynomial time.
- **Does P = NP?** We don't know! $1+ million if you know.

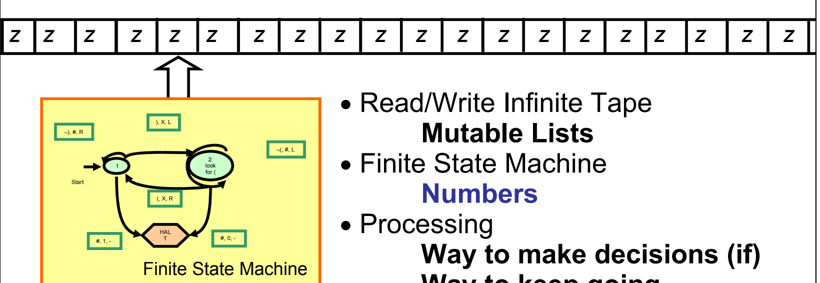
Final Exam

- Vote!
- **Option A:** All students must complete the take-home final exam. The final exam will be worth 15-50% of your grade, as per the syllabus. Individuals may still opt for the comprehensive oral exam.
- **Option B:** There will be no final exam; no student may complete the final exam. Individuals may still opt for the comprehensive oral exam.

Office Hours

- Sunday 1pm – My Apologies
- Evan Davis Today, **5pm - 9pm**, Stacks
- Vote: Yes/No
  - Evan Davis Tomorrow, 5pm-9pm, Stacks
- Pick At Most One Time and one Modality:
  - Yang He, Today, 6:00-7:00pm
  - Yang He, Tomorrow, any time **except** Noon-4pm
  - Yang He, Wednesday, any time **before** 5pm
  - Pick One: Office Hour or aim/gchat channel

Lambda Calculus is a Universal Computer?



- Read/Write Infinite Tape
  - **Mutable Lists**
- Finite State Machine
  - **Numbers**
- Processing
  - **Way to make decisions (if)**
  - **Way to keep going**

What is 42?

42

forty-two

XLII

cuarenta y dos

Meaning of Numbers

- "42-ness" is something who's **successor** is "43-ness"
- "42-ness" is something who's **predecessor** is "41-ness"
- "Zero" is special. It has a **successor** "one-ness", but no **predecessor**.

Meaning of Numbers

pred (succ N) $\rightarrow N$

succ (pred N) $\rightarrow N$

succ (pred (succ N)) $\rightarrow$ succ N

zero? **zero** $\rightarrow$ **T**

zero? (succ **zero**) $\rightarrow$ **F**

Is this enough?

Can we define **add** with **pred**, **succ**, **zero?** and **zero**?

add $\equiv \lambda xy.$**if** (**zero?** x) y
(**add** (**pred** x) (**succ** y))

Can we define lambda terms that behave like **zero, zero?, pred** and **succ**?

Hint: what if we had **cons**, **car** and **cdr**?

Numbers are Lists...

zero? $\equiv$ **null?**

pred $\equiv$ **cdr**

succ $\equiv \lambda\, x$. **cons F** x

The *length* of the list corresponds to the number value.

Liberal Arts Trivia: Religious Studies

- In Sunni Islam, the Five Pillars of Islam are five duties incumbent on Muslims. They include the Profession of Faith, Formal Prayers, and Giving Alms. Name the remaining two pillars.

Making Pairs

```
(define (make-pair x y)
  (lambda (selector) (if selector x y)))

(define (car-of-pair p) (p #t))
(define (cdr-of-pair p) (p #f))
```

A pair is just an if statement that chooses between the car (then) and the cdr (else).

#13

cons and car

cons $\equiv \lambda x.\lambda y.\lambda z.zxy$

Example: cons M N = $(\lambda x.\lambda y.\lambda z.zxy)$ M N

$\rightarrow_\beta (\lambda y.\lambda z.zMy)$ N

$\rightarrow_\beta \lambda z.z$MN

T $\equiv \lambda xy.\ x$

car $\equiv \lambda p.p$ **T**

Example: car (cons M N) $\equiv$ car $(\lambda z.z$MN$) \equiv (\lambda p.p$ **T**$)$

$(\lambda z.z$MN$) \rightarrow_\beta (\lambda z.zMN)$ **T** $\rightarrow_\beta$ **T**MN

$\rightarrow_\beta (\lambda xy.\ x)$ MN

$\rightarrow_\beta (\lambda y.$ M$)$N

$\rightarrow_\beta$ M

#14

cdr too!

cons $\equiv \lambda xyz.zxy$

car $\equiv \lambda p.p$ T

cdr $\equiv \lambda p.p$ F

Example: cdr (cons M N)

cdr $\lambda z.z$MN = $(\lambda p.p$ F$)$ $\lambda z.z$MN

$\rightarrow_\beta (\lambda z.zMN)$ F

$\rightarrow_\beta$ FMN

$\rightarrow_\beta$ N

#15

Null and null?

null $\equiv \lambda x.$**T**

null? $\equiv \lambda x.(x\ \lambda y.\lambda z.$**F**$)$

Example:

null? null $\rightarrow \lambda x.(x\ \lambda y.\lambda z.$**F**$)$ $(\lambda x.$ **T**$)$

$\rightarrow_\beta (\lambda x.$ **T**$)(\lambda y.\lambda z.$**F**$)$

$\rightarrow_\beta$ **T**

#16

Null and null?

null $\equiv \lambda x.$**T**

null? $\equiv \lambda x.(x\ \lambda y.\lambda z.$**F**$)$

Example:

null? (cons M N) $\rightarrow \lambda x.(x\ \lambda y.\lambda z.$**F**$)$ $\lambda z.z$MN

$\rightarrow_\beta (\lambda z.z$ MN$)(\lambda y.\lambda z.$**F**$)$

$\rightarrow_\beta (\lambda y.\lambda z.$**F**$)$ MN

$\rightarrow_\beta$ **F**

#17

Counting

0 $\equiv$ null

1 $\equiv$ **cons F 0**

2 $\equiv$ **cons F 1**

3 $\equiv$ **cons F 2**

…

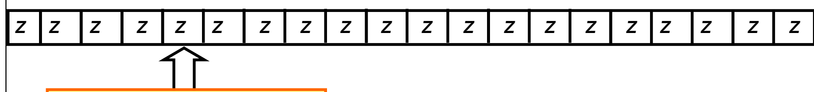
succ $\equiv \lambda x.$**cons F** x

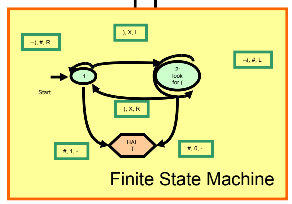
pred $\equiv \lambda x.$**cdr** x

#18

$42 = \lambda xy.(\lambda z.z\ xy)\ \lambda xy.\ y\ \lambda xy.(\lambda z.z\ xy)\ \lambda xy.\ y$
$\lambda xy.(\lambda z.z\ xy)\ \lambda xy.\ y\ \lambda xy.(\lambda z.z\ xy)\ \lambda xy.\ y\ \lambda xy.(\lambda z.z\ xy)\ \lambda xy.\ y\ \lambda xy.$
$(\lambda z.z\ xy)\ \lambda xy.\ y\ \lambda xy.(\lambda z.z\ xy)\ \lambda xy.\ y\ \lambda xy.(\lambda z.z\ xy)\ \lambda xy.\ y\ \lambda xy.(\lambda z.z\ xy)$
$\lambda xy.\ y\ \lambda xy.(\lambda z.z\ xy)\ \lambda xy.\ y\ \lambda xy.(\lambda z.z\ xy)\ \lambda xy.\ y\ \lambda xy.(\lambda z.z\ xy)\ \lambda xy.\ y$
$\lambda xy.(\lambda z.z\ xy)\ \lambda xy.\ y\ \lambda xy.(\lambda z.z\ xy)\ \lambda xy.\ y\ \lambda xy.(\lambda z.z\ xy)\ \lambda xy.\ y\ \lambda xy.$
$(\lambda z.z\ xy)\ \lambda xy.\ y\ \lambda xy.(\lambda z.z\ xy)\ \lambda xy.\ y\ \lambda xy.(\lambda z.z\ xy)\ \lambda xy.\ y\ \lambda xy.(\lambda z.z\ xy)$
$\lambda xy.\ y\ \lambda xy.(\lambda z.z\ xy)\ \lambda xy.\ y\ \lambda xy.(\lambda z.z\ xy)\ \lambda xy.\ y\ \lambda xy.(\lambda z.z\ xy)\ \lambda xy.\ y$
$\lambda xy.(\lambda z.z\ xy)\ \lambda xy.\ y\ \lambda xy.(\lambda z.z\ xy)\ \lambda xy.\ y\ \lambda xy.(\lambda z.z\ xy)\ \lambda xy.\ y\ \lambda xy.$
$(\lambda z.z\ xy)\ \lambda xy.\ y\ \lambda xy.(\lambda z.z\ xy)\ \lambda xy.\ y\ \lambda xy.(\lambda z.z\ xy)\ \lambda xy.\ y\ \lambda xy.(\lambda z.z\ xy)$
$\lambda xy.\ y\ \lambda xy.(\lambda z.z\ xy)\ \lambda xy.\ y\ \lambda xy.(\lambda z.z\ xy)\ \lambda xy.\ y\ \lambda xy.(\lambda z.z\ xy)\ \lambda xy.\ y$
$\lambda xy.(\lambda z.z\ xy)\ \lambda xy.\ y\ \lambda xy.(\lambda z.z\ xy)\ \lambda xy.\ y\ \lambda xy.(\lambda z.z\ xy)\ \lambda xy.\ y\ \lambda xy.$
$(\lambda z.z\ xy)\ \lambda xy.\ y\ \lambda xy.(\lambda z.z\ xy)\ \lambda xy.\ y\ \lambda xy.(\lambda z.z\ xy)\ \lambda xy.\ y\ \lambda xy.(\lambda z.z\ xy)$
$\lambda xy.\ y\ \lambda xy.(\lambda z.z\ xy)\ \lambda xy.\ y\ \lambda xy.(\lambda z.z\ xy)\ \lambda xy.\ y\ \lambda xy.(\lambda z.z\ xy)$
$\lambda xy.\ y\ \lambda x.\mathrm{T}$

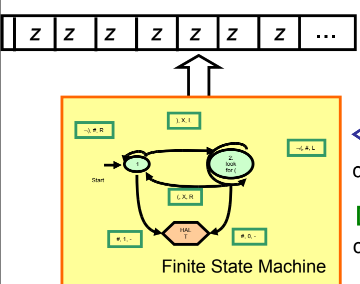
Lambda Calculus is a Universal Computer





- Read/Write Infinite Tape
  - ✓ Mutable Lists
- Finite State Machine
  - ✓ Numbers to keep track of state
- Processing
  - ✓ Way of making decisions (if)
  - ✓ Way to keep going

Equivalent Computers!



can simulate

can simulate

term = *variable*
| *term term*
| (*term*)
| λ *variable* . *term*

$\lambda y.\ M \Rightarrow_\alpha \lambda v.\ (M\,[y \to v])$
where v does not occur in M.

$(\lambda x.\ M)N \Rightarrow_\beta M\,[\,x \to N\,]$

Turing Machine

Lambda Calculus

Liberal Arts Trivia: Biology, Chemistry

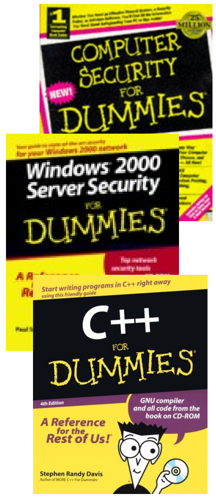
- While rendered fat obtained form pigs is known as lard, rendered beef or mutton fat is known as *this*. It is used to make animal feed and soap. Historically, it was used to make candles: it provided a cheaper alternative to wax. Before switching to vegetable oil in 1990, McDonald's cooked fries in 93% *this* and 7% cottonseed oil.

Universal Computer

- Lambda Calculus can simulate a Turing Machine
  - Everything a Turing Machine can compute, Lambda Calculus can compute also
- Turing Machine can simulate Lambda Calculus (we didn't prove this)
  - Everything Lambda Calculus can compute, a Turing Machine can compute also
- **Church-Turing Thesis**: this is true for ***any*** other mechanical computer also

What about "non-mechanical" computers?

Quantum Physics for Dummies

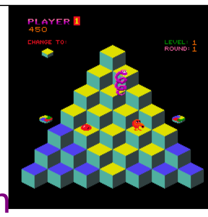


- Light behaves like both a wave and a particle at the same time
- A single photon is in many states at once
- Can't observe its state without forcing it into one state
- Schrödinger's Cat
  - Put a live cat in a box with cyanide vial that opens depending on quantum state
  - Cat is both dead and alive at the same time until you open the box

Quantum Computing

- Feynman, 1982
- Quantum particles are in all possible states
- Can try lots of possible computations at once with the same particles
- In theory, can test all possible factorizations/keys/paths/etc. and get the right one!
- In practice, very hard to keep states entangled: once disturbed, must be in just one possible state

Qubit

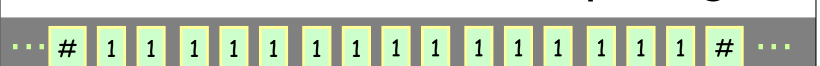


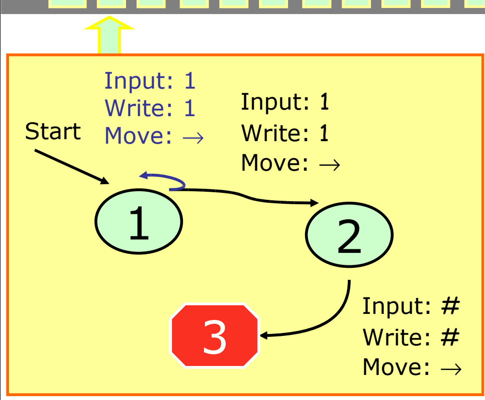
- Regular bit: either a 0 or a 1
- **Quantum bit**: 0, 1 or in between
  - p% probability it is a 1
- A single qubit is in 2 possible states at once
- If you have 7 bits, you can represent any one of 2^7 different states
- If you have 7 qubits, you have 2^7 different states (at once!)

Quantum Computers Today

- Several quantum algorithms
  - Shor's algorithm: factoring using a quantum computer
- Actual quantum computers
  - 5-qubit computer built by IBM (2001)
  - Implemented Shor's algorithm to factor:
    - "World's most complex quantum computation" **15** (= 5 * 3)
  - D-Wave 16-qubit quantum computer (2007)
    - Solves Sudoku puzzles
- To exceed practical normal computing need > 50 qubits
  - Adding another qubit is more than twice as hard

Nondeterministic Computing





There can be multiple transitions on the same input. Nondeterministic TM takes *all* of them at once. Each gets its own independent copy of the tape. If *any* path finds halting state, that is the result.

Two Ways of Thinking about Nondeterminstic Computing

- Omniscient (all-knowing): machine always guesses right (the right guess is the one that eventually leads to a halting state)
- Omnipotent (all-powerful): machine can split in two every step, all resulting machines execute on each step, if one of the machines halts its tape is the output

Computability

Is a nondeterministic TM more **powerful** than a deterministic TM?

Liberal Arts Trivia: Geography

- This second-longest river in the United States flows form Lake Itasca in Minnesota to the Gulf of Mexico. Forty percent of North America's ducks, geese, swan and wading bird species use it as a migration corridor. It serves as the shared border for ten states, contains over 29 locks and dams, and generates more than a billion dollars a year in revenue from recreational uses, including over 600 water-oriented sites.

Computability

Is a nondeterministic TM more **powerful** than a deterministic TM?

No! We can simulate a nondeterminstic TM with a regular TM.

Speed

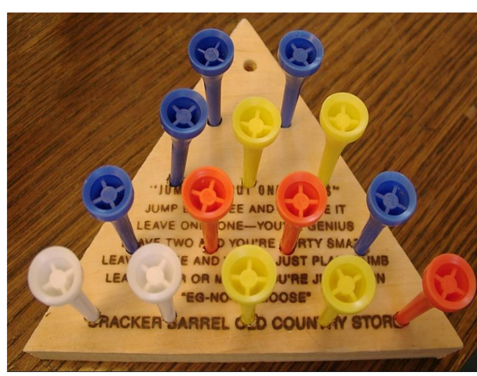
Is a nondeterministic TM **faster** than a deterministic TM?

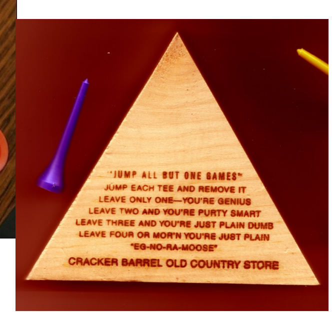
Speed

Is a nondeterministic TM **faster** than a deterministic TM?

Unknown! This is the most famous open problem in CS.

Pegboard Problem





Pegboard Problem

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

How hard is the Pegboard Problem?

Problems and Procedures

- To know a $O(f)$ **bound for a problem**, we need to find a $\Theta(f)$ *procedure* that solves it
  - The sorting **problem** is $O(n \log n)$ since we know a **procedure** that solves it in $\Theta(n \log n)$
- To know a $\Omega(f)$ bound for a **problem**, we need to prove that there is no procedure faster than $\Theta(f)$ that solves it
  - We could prove sorting is $\Omega(n \log n)$ by reasoning about the number of decisions needed

How much work is the Pegboard Problem?

- Upper bound: (O)
  $O(n!)$
  Try all possible permutations
- Lower bound: (Ω)
  $\Omega(n)$
  Must at least look at every peg
- Tight bound: (Θ)
  - What do you think?

How much work is the Pegboard Problem?

- Upper bound: (O)
  $O(n!)$
  Try all possible permutations
- Lower bound: (Ω)
  $\Omega(n)$
  Must at least look at every peg
- Tight bound: (Θ)

No one knows!

Complexity Class P "Tractable"

Class P: problems that can be solved in a polynomial ($O(n^k)$ for some constant k) number of steps by a deterministic TM.

Easy problems like sorting, making a photomosaic using duplicate tiles, and simulating the universe are all in **P**.

Liberal Arts Trivia: Astronomy

- This space telescope was launched into orbit by the Space Shuttle Discovery in 1990. After having its incorrectly-ground main mirror replaced in 1993, it has helped to make breakthroughs in astrophysics. Notable are its Ultra Deep Field image, which details the universe's most distant objects, and its help in determining the ultimate expansion of the universe (the eponymous constant).

Complexity Class NP

Class NP: Problems that can be solved in a polynomial number of steps by a ***nondeterministic*** TM.

Omnipotent: If we could try all possible solutions at once, we could identify the solution in polynomial time.

Omniscient: If we had a magic guess-correctly procedure that makes every decision correctly, we could devise a procedure that solves the problem in polynomial time.

NP Problems

- Can be solved by just trying all possible answers until we find one that is right
- Easy to quickly check if an answer is right
  - Checking an answer is in **P**
- The pegboard problem is in **NP**

  We can easily try $\sim n!$ different answers

  We can check if a guess is correct in $O(n)$ (check all n jumps are legal)

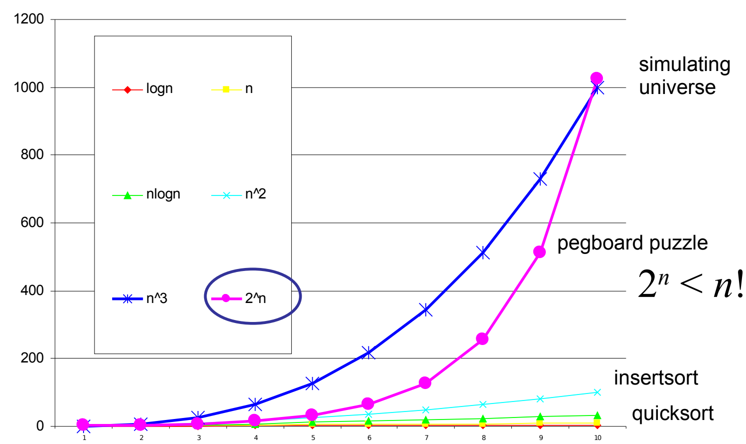
Is the Pegboard Problem in **P**?

No one knows!

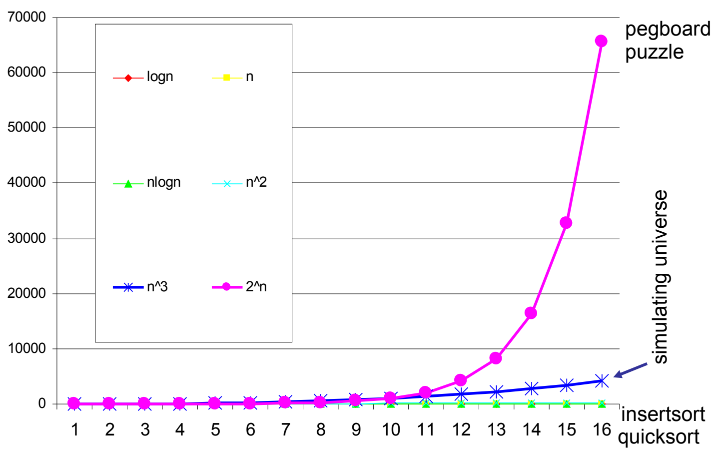
We can't find a $O(n^k)$ solution.

We also can't prove one doesn't exist.

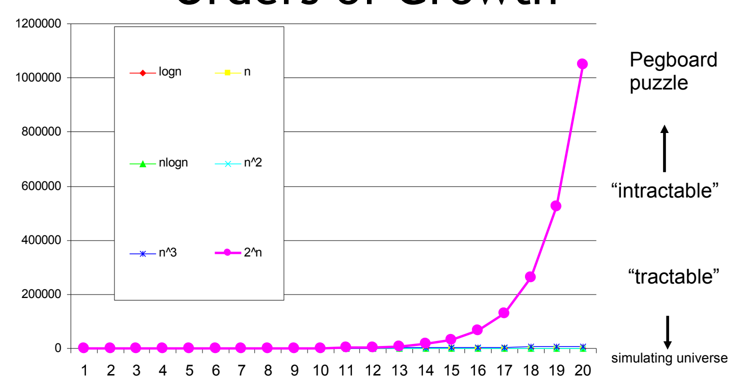
Orders of Growth



Orders of Growth



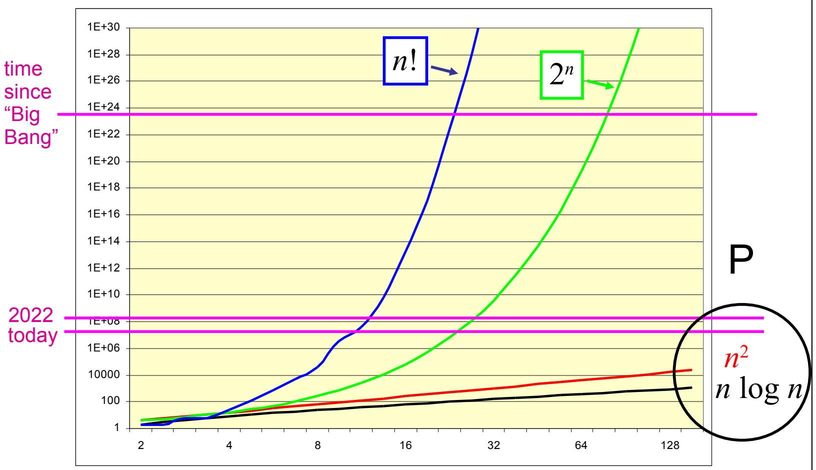
Orders of Growth



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)

Intractable Problems

log-log scale



Moore's Law Doesn't Help

- If the fastest procedure to solve a problem is $\Theta(2^n)$ or worse, Moore's Law doesn't help much.
- Every ***doubling*** in computing power only increases the solvable problem size by **1**.

Complexity Classes

Class P: problems that can be solved in polynomial time by deterministic TM

Easy problems like simulating the universe are all in **P**.

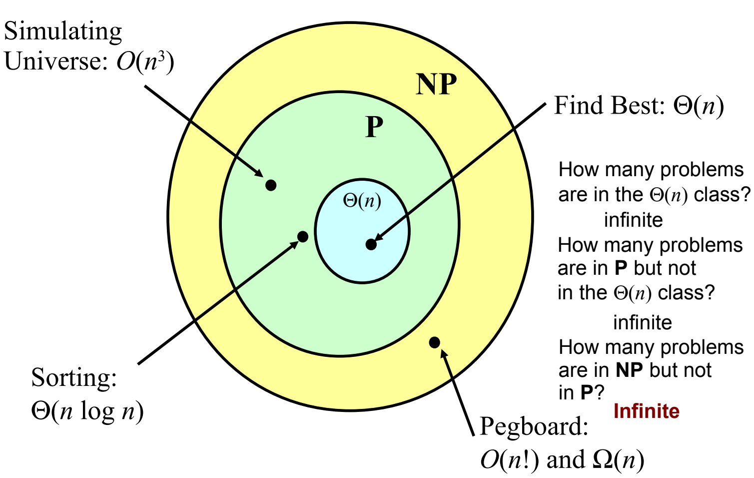
Class NP: problems that can be solved in polynomial time by a nondeterministic TM.

Includes all problems in **P** and some problems possibly outside **P** like the Pegboard puzzle.

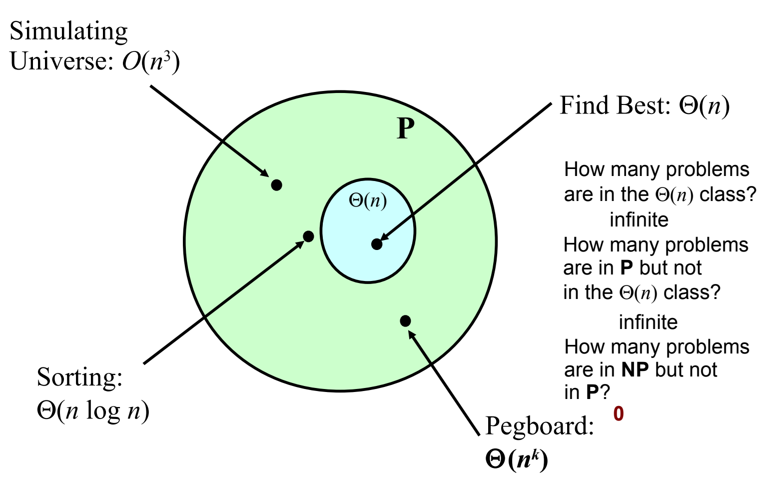
Liberal Arts Trivia: Psychology

- This Swiss philosopher and natural scientist is known as "the great pioneer of the constructivist theory of knowing." His four stages of development include infancy, preschool, childhood and adolescence. Each stage corresponds to the child's understanding of reality (e.g., conservation, abstract reasoning) during that period.

Problem Classes if P ≠ NP:



Problem Classes if P = NP:



P = NP?

- Is P different from NP: is there a problem in NP that is not also in P
  - If there is one, there are infinitely many
- Is the “hardest” problem in NP also in P
  - If it is, then every problem in NP is also in P
- **The most famous unsolved problem in computer science and math**
  - Listed first on Millennium Prize Problems
  - $1M + an automatic A+ in this course
    - (and probably all CS courses at UVA ...)

NP-Complete

- **NP-Complete**: is the class of problems that are the *hardest* problems in **NP**
- Cook and Levin proved that 3SAT was NP-Complete (1971)
  - If 3SAT can be transformed into a different problem in polynomial time, than that problem must also be **NP-complete**.
  - Pegboard ⇔ 3SAT
- **Either all NP-complete problems are tractable (in P) or none of them are!**

NP-Complete Problems

- Easy way to solve by trying all possible guesses
- If given the “yes” answer, quick (in P) way to check if it is right
- If given the “no” answer, no quick way to check if it is right
  - No solution (can’t tell there isn’t one)
  - No way (can’t tell there isn’t one)

This part is hard to prove: requires showing you could use a solution to the problem to solve a known NP-Complete problem.

Most Important Science/Technology Races

1930-40s: Decryption — Nazis vs. British
Winner: British
Reason: Bletchley Park had computers (and Alan Turing), Nazi’s didn’t

1940s: Atomic Bomb — Nazis vs. US
Winner: US
Reason: Heisenberg miscalculated, US had better physicists, computers, resources

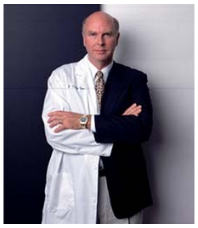
1960s: Moon Landing — Soviet Union vs. US
Winner: US
Reason: Many, better computing was a big one

1990s-2001: Sequencing Human Genome

Human Genome Race

Francis Collins
(Director of public National Center for Human Genome Research)
(Picture from UVa Graduation 2001)

VS.

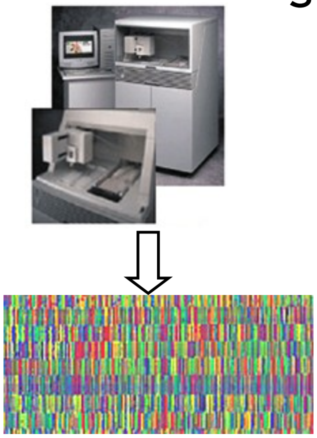


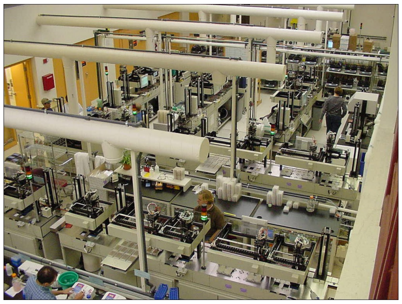
Craig Venter
(President of Celera Genomics)

- UVa CLAS 1970
- Yale PhD
- Tenured Professor at U. Michigan

- San Mateo College
- Almost court-martialed
- Denied tenure at SUNY Buffalo

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

AGGCATACCAGAATACCCGTGATCCAGAATAAGC

Actual Genome

Liberal Arts Trivia: Greek Myths

- The Sphinx is said to have guarded the entrance to the city of Thebes and to have asked a riddle of would-be travelers. The Sphinx, originally from Ethiopia, is said to have been sent by Hera or Ares. Oedipus solved her riddle, and is thus seen as a threshold figure, helping to transition between the old religious practices and the new Olympian gods.
- State the Riddle of the Sphinx and its answer.

Common Superstring

Input: A set of n substrings and a maximum length k.

Output: A string that contains all the substrings with total length $\leq k$, or no if no such string exists.

ACCAGAATACC
TCCAGAATAA
TACCCGTGATCCA

$n = 26$

⇨

ACCAGAATACC
TCCAGAATAA
TACCCGTGATCCA
ACCAGAATACCCGTGATCCAGAATAA

Common Superstring

Input: A set of n substrings and a maximum length k.

Output: A string that contains all the substrings with total length $\leq k$, or no if no such string exists.

ACCAGAATACC
TCCAGAATAA
TACCCGTGATCCA

$n = 25$

⇨ **Not possible**

Common Superstring

- In **NP:**
  - Easy to verify a "yes" solution: just check the letters match up, and count the superstring length
- In **NP-Complete:**
  - Similar to Pegboard Puzzle!
  - Could transform Common Superstring problem instance into Pegboard Puzzle instance!

Human Genome

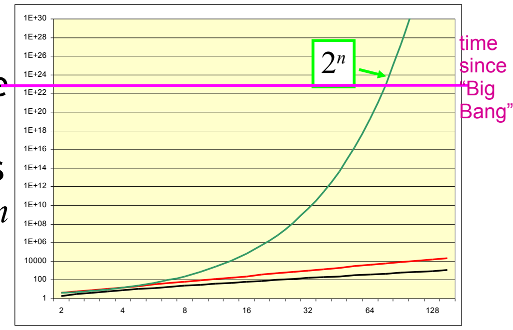
- 3 Billion base pairs
- 600-700 bases per read
- ~8X coverage required

```
> (/ (* 8 3000000000)) 650)
36923076 12/13
```

- So, $n \approx$ 37 Million sequence fragments
- Celera used 27.2 Million reads (but could get more than 700 bases per read)

Give up?

No way to solve an NP-Complete problem (best known solutions being O(2^n) for n ≈ 20 Million)



Approaches

- Human Genome Project (Collins)
  - **Change problem:** Start by producing a genome map (using biology, chemistry, etc) to have a framework for knowing where the fragments should go
- Celera Solution (Venter)
  - **Approximate:** we can’t guarantee finding the shortest possible, but we can develop clever algorithms that get close most of the time in $O(n \log n)$

Result: Draw?

President Clinton announces Human Genome Sequence essentially complete, June 26, 2000

“Biology 2.0”

- The past few weeks have seen an announcement that may, in retrospect, turn out to have been as portentous as the sequencing of the human genome: Dr Venter’s construction of an organism with a completely synthetic genome. The ability to write new genomes in this way brings true biological engineering—as opposed to the tinkering that passes for biotechnology at the moment—a step closer. *(The Economist, July 2010)*

CS 1120

- Language: Formal Systems, Rules of Eval
- Recursive Definitions
- Programming with Lists
- Programming with Mutation and Objects
- Interpreters, Lazy Eval, Type Checking
- Programming for the Internet
- Measuring Complexity
- Computability
- Models of Computation

Homework

- PS 9 Presentation Requests due Today
- This Wednesday, Here, 5pm-...
  - Presentations
  - Extra Credit on PS9 for attending
- Tuesday December 14
  - Final Project Reports Due

Liberal Arts Trivia: Bias

- Weimer recommends that you take classes on philosophy until you've covered epistemology, free will, logic, the philosophy of science, and “what it is like to be a bat”. Take cognitive psychology classes until you've covered perception and the Flynn effect. Take speech or rhetoric classes until you've covered persuasion. Take anthropology as well as gender studies classes until you've covered Mead and Freeman and you have a better feel for which behaviors are socially constructed and which may be essential. Take classes in statistics until you can avoid being fooled. Take classes in religion or ethics until you've covered the relationship between unhappiness and unrealized desires. Take classes in physics until you can explain how a microphone, radio and speaker all work. Take classes on government until you have an opinion about the feasibility of legislating morality. Take classes on history until you are not condemned to repeat the mistakes of the past.