The Beauty and Joy of Computing

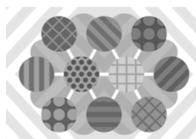
Lecture #23 Limits of Computing

UC Berkeley EECS
Lecturer
Gerald Friedland

2014-11-24

THE THREE BREAKTHROUGHS THAT HAVE FINALLY BROUGH AI ON THE WORLD

http://www.wired.com/2014/10/future-of-artificial-intelligence/

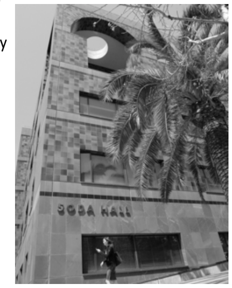


---

www.eecs.berkeley.edu/Research/Areas/

Computer Science … A UCB view

- **CS department research areas:**
  - Artificial Intelligence
  - Biosystems & Computational Biology
  - Database Management Systems
  - Graphics
  - Human-Computer Interaction
  - Networking
  - Programming Systems
  - Scientific Computing
  - Security
  - Systems
  - Theory
    - Complexity theory
  - …

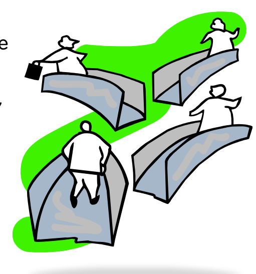


---

www.csprinciples.org/docs/APCSPrinciplesBigIdeas20110204.pdf

Let's revisit algorithm complexity

- **Problems that…**
  - are tractable with efficient solutions in reasonable time
  - are intractable
  - are solvable approximately, not optimally
  - have no known efficient solution
  - are not solvable



Tractable with efficient sols in real time

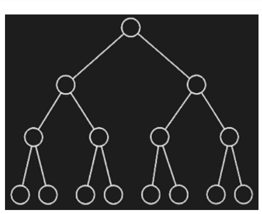
- **Recall our algorithm complexity lecture, we've got several common orders of growth**
  - Constant
  - Logarithmic
  - Linear
  - Quadratic
  - Cubic
  - Exponential
- **Order of growth is polynomial in the size of the problem**
- **E.g.,**
  - Searching for an item in a collection
  - Sorting a collection
  - Finding if two numbers in a collection are same
- **These problems are called being "in P" (for polynomial)**

---

en.wikipedia.org/wiki/Intractability_(complexity)#Intractability

Intractable problems

- **Problems that can be solved, but not solved fast enough**
- **This includes exponential problems**
  - E.g., $f(n) = 2^n$
    - as in the image to the right
- **This also includes poly-time algorithm with a huge exponent**
  - E.g, $f(n) = n^{10}$
- **Only solve for small n**

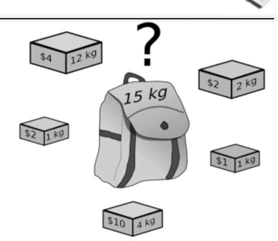


Imagine a program that calculated something important at each of the bottom circles. This tree has height n, but there are 2^n bottom circles!

Peer Instruction

What's the most you can put in your knapsack?

a) **$10**
b) **$15**
c) **$33**
d) **$36**
e) **$40**



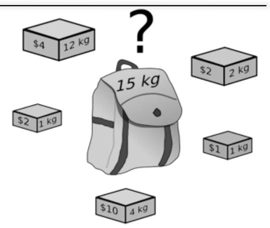
Knapsack Problem

You have a backpack with a weight limit (here **15kg**), which boxes (with weights and values) should be taken to maximize value?

(any # of each box is available)

Solvable approximately, not optimally in reas time

- A problem might have an optimal solution that cannot be solved in reasonable time
- BUT if you don't need to know the perfect solution, there might exist algorithms which could give pretty good answers in reasonable time



Knapsack Problem
You have a backpack with a weight limit (here 15kg), which boxes (with weights and values) should be taken to maximize value?

Have no known efficient solution

- Solving one of them would solve an entire class of them!
  - We can transform one to another, i.e., reduce
  - A problem P is "hard" for a class C if every element of C can be "reduced" to P
- If you're "in NP" and "NP-hard", then you're "NP-complete"

-2 -3 15
14 7 -10

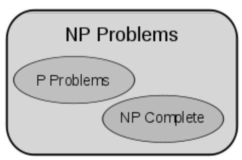
Subset Sum Problem
Are there a handful of these numbers (at least 1) that add together to get 0?

- If you guess an answer, can I verify it in polynomial time?
  - Called being "in NP"
  - Non-deterministic (the "guess" part) Polynomial

The fundamental question. Is P = NP?

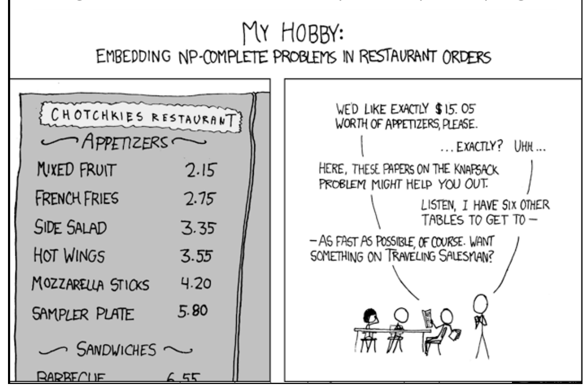
- This is THE major unsolved problem in Computer Science!
  - One of 7 "millennium prizes" w/a $1M reward
- All it would take is solving ONE problem in the NP-complete set in polynomial time!!
  - Huge ramifications for cryptography, others

If P ≠ NP, then

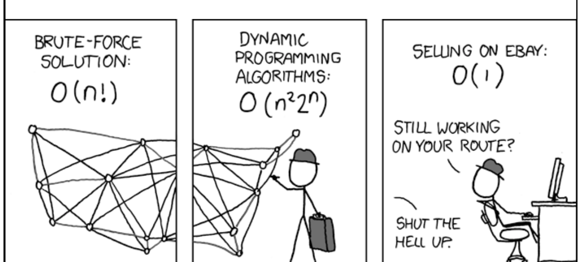


- Other NP-Complete
  - Traveling salesman who needs most efficient route to visit all cities and return home

imgs.xkcd.com/comics/np_complete.png

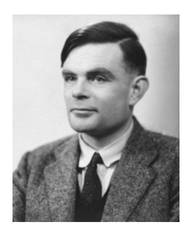


imgs.xkcd.com/comics/travelling_salesman_problem.png



Problems NOT solvable

- Decision problems answer YES or NO for an infinite # of inputs
  - E.g., is N prime?
  - E.g., is sentence S grammatically correct?
- An algorithm is a solution if it correctly answers YES/NO in a finite amount of time
- A problem is decidable if it has a solution



Alan Turing
He asked:
"Are all problems decidable?"
(people used to believe this was true)
Turing proved it wasn't for CS!

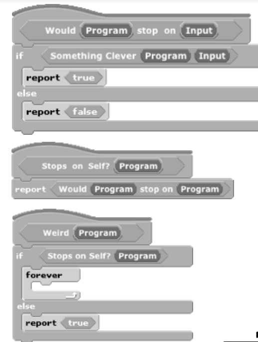
Review: Proof by Contradiction

- **Infinitely Many Primes?**
- **Assume the contrary, then prove that it's impossible**
  - Only a finite set of primes, numbered $p_1, p_2, \ldots, p_n$
  - Consider $q=(p_1 \bullet p_2 \bullet \ldots \bullet p_n)+1$
  - Dividing q by p_i has remainder 1
  - q either prime or composite
    - If prime, q is not in the set
    - If composite, since no p_i divides q, there must be another p that does that is not in the set.
  - So there's infinitely many primes

Euclid
www.hisschemoller.com/wp-content/uploads/2011/01/euclides.jpg

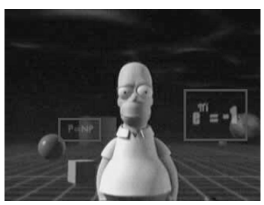
Turing's proof : The Halting Problem

- **Given a program and some input, will that program eventually stop? (or will it loop)**
- **Assume we could write it, then let's prove a contradiction**
  - 1. write Stops on Self?
  - 2. Write Weird
  - 3. Call Weird on itself



Conclusion

- **Complexity theory important part of CS**
- **If given a hard problem, rather than try to solve it yourself, see if others have tried similar problems**
- **If you don't need an exact solution, many approximation algorithms help**
- **Some not solvable!**



P=NP question even made its way into popular culture, here shown in the Simpsons 3D episode!