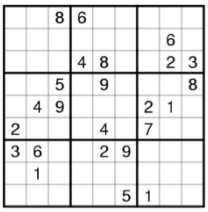
What is the computational cost of automating brilliance?

Computational complexity
and the P vs NP question

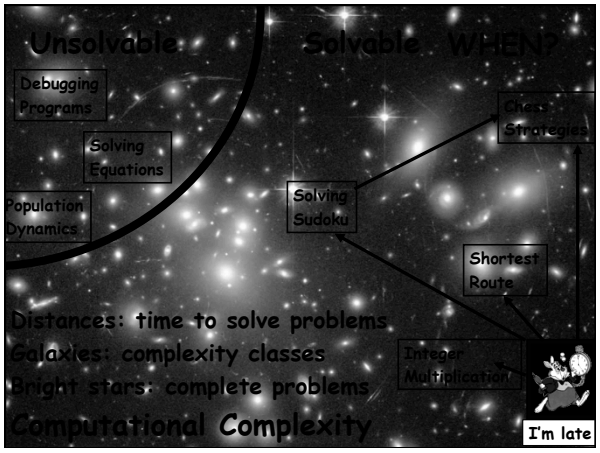
From slides by Avi Wigderson
+ few by Sanjeev Arora
+ couple by Christos Papadimitriou
++

SURVEY

Finding an efficient method to solve SuDoku puzzles is:



1: A waste of time
2: A decent spare time activity
3: A fundamental problem of science and math



Efficiency of Algorithms
Asymptotic complexity
(of an algorithm)

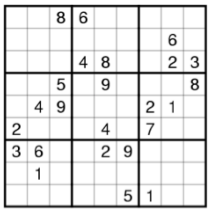
How does the number of steps of an algorithm increase with the data size (input length) ?

input

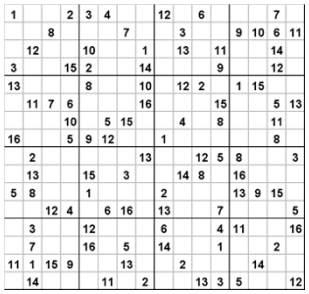
Rubik's cube

2 3 4 5 …

Sudoku

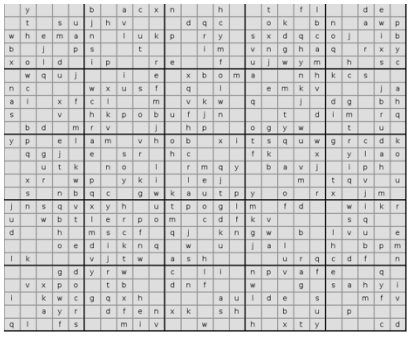


3



4

Sudoku



5 ……

Complexity of functions

comp(add) = n

~~comp(multiply) ≤ n^2 [gradeschool]~~

comp(multiply) ≤ n·(log n) [schoenhage-strassen]

Is there a better algorithm?

Is there no bette

Grade-school multiply algorithm

Main challenges of The

Only efficient algorith

Efficient: n, n·logn, n^2

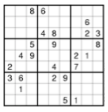
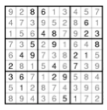
Inefficient: 2^n, $2^{\sqrt{n}}$,…

The class P

All problems having an efficient algorithm to *find* solutions
(the galaxy of problems closest to us)

Are all practically interesting problems in P?

Three problems

	Input	Output	Complexity
Factoring integers	1541 $2^{67}-1$	23 x67 ? x?	$\leq 2^{\sqrt{n}}$
Proving theorems	n+"Riemann Hypothesis"	n symbol proof	$\leq 2^n$
Solving Sudoku			$\leq n^n$

Verification

$2^{67}-1$ = 193707721 x 761838257287

n+Poincare Conjectute

n+Fermat's "Theorem"

n = 200 pages

	Input	Output	Complexity
✓ Factoring integers	1541 $2^{67}-1$	23 x67 ??	$\leq 2^{\sqrt{n}}$
✓ Proving theorems	n+"Riemann Hypothesis"	n symbol proof	$\leq 2^n$
✓ Solving Sudoku			$\leq n^n$

What is common to all 3 problems?

-All look currently intractable, even for moderate n (best algorithms exponential)

- Specific instances get solved!

- Easy verification of given solutions !!!

Cook & Levin ~1971

The class NP

All problems having efficient verification algorithms of given solutions

For every such problem, finding a solution (of length n) takes $\leq 2^n$ steps: try all possible solutions & verify each.

Includes Factoring, Theorem-proving, Sudoku, many others

Can we do better than "brute force" ?
Do all NP problems have efficient algs ?

P versus NP

P: Problems for which solutions can be efficiently *found*

NP: Problems for which solutions can be efficiently *verified*

Fact: P ⊆ NP [finding implies verification]

Conjecture: P ≠ NP [finding is much harder than verification]

"P=NP?" is a central question of math, science, technology and beyond!!!

what is in NP?

Mathematician: Given a statement, *find* a proof

Scientist: Given data on some phenomena, *find* a theory explaining it.

Engineer: Given constraints (size,weight,energy) *find* a design (bridge, medicine, phone)

In many intellectual challenges, *verifying* that we found a good solution is an easy task ! (if not, we probably wouldn't start looking)

If P=NP, these have fast, automatic *finder*

How do we tackle P vs. NP?

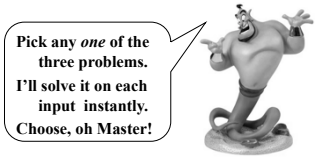
Break RSA, ruin E-commerce

Fame & glory $6M from CLAY

Take out the fun of Doing these puzzles

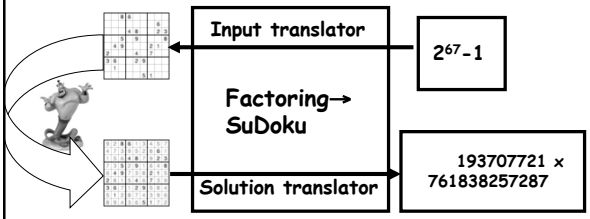
Let's choose the SuDoku solver

	Input	Output	Complexity
Factoring integers	1541 $2^{67}-1$	23 x67 ??	$\leq 2^{\sqrt{n}}$
Proving theorems	n+"Riemann Hypothesis"	n symbol proof	$\leq 2^n$
Solving SuDoku			$\leq n^n$



The power of SuDoku I

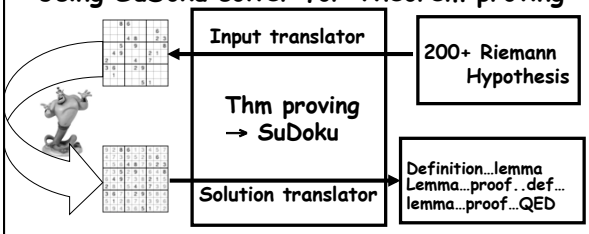
Using SuDoku solver for Integer factoring



Both translators are known, efficient algorithms!

The power of SuDoku II

Using SuDoku solver for Theorem proving



Both translators are known, efficient algorithms!

"Reduction"

"If you give me a place to stand, I will move the earth."
– Archimedes (~ 250BC)

"If you give me a polynomial-time algorithm for Sudoku, I will give you a polynomial-time algorithm for every NP problem." --- Cook, Levin (1971)

"Every NP problem is a Sudoku problem in disguise."

Universality: NP-completeness

SuDoku solver can solve any NP problem

1971: NP-complete problems exist!
SAT is NP-complete: There is a "reduction" from any NP problem to SAT

NP-complete problems abound!
1972: 21 problems in logic, optimization, algebra
Today: ~3000 problems in all sciences, *equivalent*

P=NP iff SuDoku has an efficient algorithm

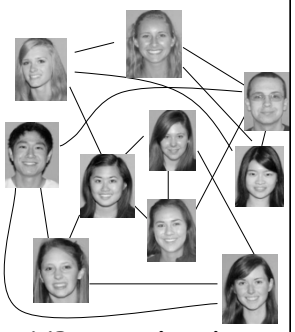
Universality: NP-completeness

NP-complete problems:
If one is easy, then all are!
If one is hard, then all are!

SuDoku,	NP-complete
Thm proving:	NP-complete
Integer factoring:	we don't know

CLIQUE Problem

- Social network
- Each node represents a student
- Two nodes connected by edge if those students are friends
- In this social network, is there a clique of k or more people?
- CLIQUE: Group of students, every pair of whom are friends
- What is a good algorithm for detecting the biggest clique?
- How does efficiency depend on network size and desired clique size?



NP-complete!

Map Coloring

Input: planar map M

4-COL: is M 4-colorable?

YES!

3-COL: is M 3-colorable?

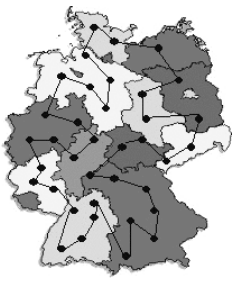
Give me an algorithm.

NP-complete!

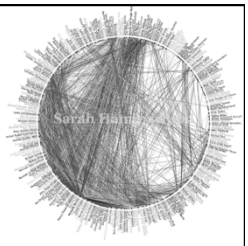
Traveling Salesman Problem (aka UPS Truck problem)

- Input: *n* points and all pairwise inter-point distances, and a distance *k*
- Decide: is there a path that visits all the points ("salesman tour") whose total length is at most *k*?

- **NP-complete!**

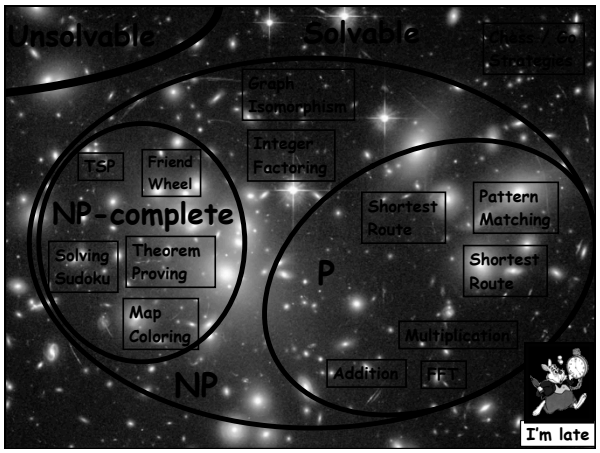


Optimal Friend Wheel



- Input: list of your friends, together with information on which pairs of them are friends
- Decide: is there a way to lay them out on a circle so that the sum of distances between friends is at most B?

- **NP-complete!**



Why is P vs NP a million-dollar open problem?

- If P = NP then brilliance becomes routine (best schedule, best route, best design, best math proof, etc…) and no more ecommerce..
- If P ≠ NP then we know something new and fundamental not just about computers but about the world (akin to "Nothing travels faster than light").