

7.8 Intractability



A Reasonable Question about Algorithms

Q. Which **algorithms** are useful in practice?

A. [von Neumann 1953, Gödel 1956, Cobham 1964, Edmonds 1965, Rabin 1966]

- Model of computation = deterministic Turing machine.
- Measure running time as a function of input size N .
- Useful in practice ("efficient") = **polynomial time** for all inputs.

aN^b

Ex 1. Sorting N elements takes N^2 steps using insertion sort.

Ex 2. Finding best TSP tour on N elements takes $N!$ steps using exhaustive search.

Theory. Definition is broad and robust.

Practice. Poly-time algorithms scale to huge problems.



constants a and b tend to be small

Exponential Growth

Exponential growth dwarfs technological change.

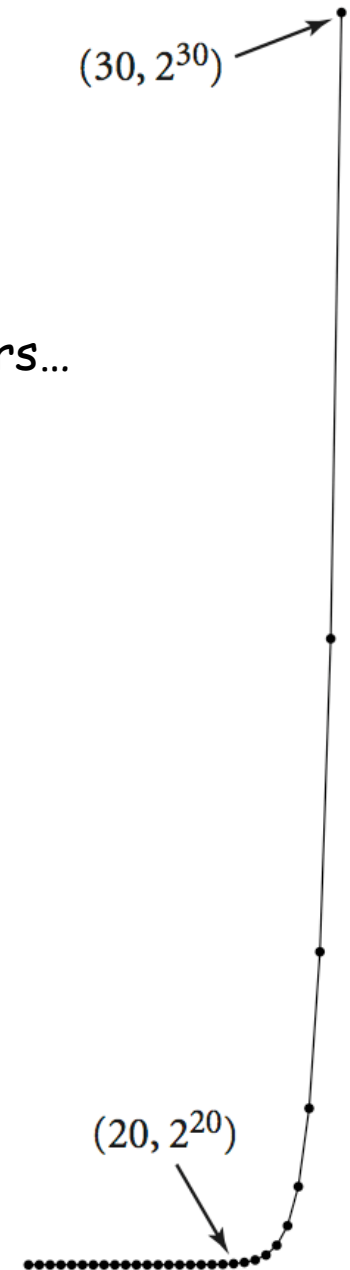
- Suppose you have a giant parallel computing device...
- With as many processors as electrons in the universe...
- And each processor has power of today's supercomputers...
- And each processor works for the life of the universe...

quantity	value
electrons in universe †	10^{79}
supercomputer instructions per second	10^{13}
age of universe in seconds †	10^{17}

† estimated

- Will not help solve 1,000 city TSP problem via brute force.

$1000! \gg 10^{1000} \gg 10^{79} \times 10^{13} \times 10^{17}$



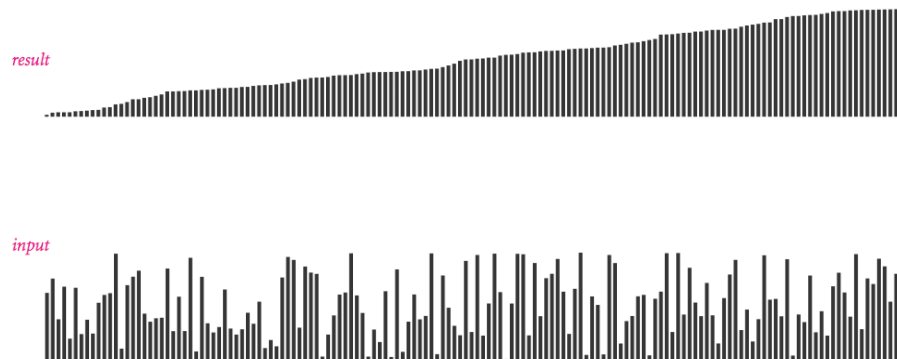
Reasonable Questions about Problems

Q. Which **problems** can we solve in practice?

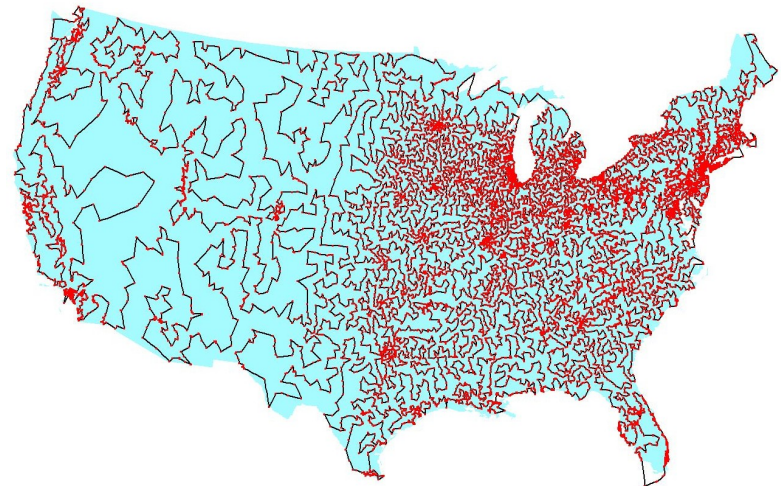
A. Those with guaranteed poly-time algorithms.

Q. Which **problems** have guaranteed poly-time algorithms?

A. Not so easy to know. Focus of today's lecture.



many known poly-time algorithms for sorting



no known poly-time algorithm for TSP

Three Fundamental Problems

LSOLVE. Given a system of **linear** equations, find a solution.

$$\begin{array}{rclcl} 0x_0 & + & 1x_1 & + & 1x_2 & = & 4 \\ 2x_0 & + & 4x_1 & - & 2x_2 & = & 2 \\ 0x_0 & + & 3x_1 & + & 15x_2 & = & 36 \end{array}$$

$$\begin{array}{rcl} x_0 & = & -1 \\ x_1 & = & 2 \\ x_2 & = & 2 \end{array}$$

LP. Given a system of linear **inequalities**, find a solution.

$$\begin{array}{rclcl} 48x_0 & + & 16x_1 & + & 119x_2 & \leq & 88 \\ 5x_0 & + & 4x_1 & + & 35x_2 & \geq & 13 \\ 15x_0 & + & 4x_1 & + & 20x_2 & \geq & 23 \\ x_0 & , & x_1 & , & x_2 & \geq & 0 \end{array}$$

$$\begin{array}{rcl} x_0 & = & 1 \\ x_1 & = & 1 \\ x_2 & = & \frac{1}{5} \end{array}$$

ILP. Given a system of linear inequalities, find a **binary** solution.

$$\begin{array}{rclcl} & x_1 & + & x_2 & \geq & 1 \\ x_0 & & & + & x_2 & \geq & 1 \\ x_0 & + & x_1 & + & x_2 & \leq & 2 \end{array}$$

$$\begin{array}{rcl} x_0 & = & 0 \\ x_1 & = & 1 \\ x_2 & = & 1 \end{array}$$

each x_i is either 0 or 1

Three Fundamental Problems

LSOLVE. Given a system of linear equations, find a solution.

LP. Given a system of linear inequalities, find a solution.

ILP. Given a system of linear inequalities, find a binary solution.

Q. Which of these problems have poly-time solutions?

A. No easy answers.

✓ **LSOLVE.** Yes. Gaussian elimination solves N -by- N system in N^3 time.

✓ **LP.** Yes. Ellipsoid algorithm is poly-time. ← open problem for decades

? **ILP.** No poly-time algorithm known or believed to exist!

Search Problems

Search problem. Given an instance I of a problem, **find** a solution S .
Requirement. Must be able to efficiently **check** that S is a solution.

or report none exists

poly-time in size of instance I



Search Problems

Search problem. Given an instance I of a problem, **find** a solution S .
Requirement. Must be able to efficiently **check** that S is a solution.

or report none exists

poly-time in size of instance I

LSOLVE. Given a system of linear equations, find a solution.

$$\begin{array}{rclcl} 0x_0 & + & 1x_1 & + & 1x_2 & = & 4 \\ 2x_0 & + & 4x_1 & - & 2x_2 & = & 2 \\ 0x_0 & + & 3x_1 & + & 15x_2 & = & 36 \end{array}$$

instance I

$$\begin{array}{rcl} x_0 & = & -1 \\ x_1 & = & 2 \\ x_2 & = & 2 \end{array}$$

solution S

- To check solution S , plug in values and verify each equation.

Search Problems

Search problem. Given an instance I of a problem, **find** a solution S .
Requirement. Must be able to efficiently **check** that S is a solution.

or report none exists

poly-time in size of instance I

LP. Given a system of linear inequalities, find a solution.

$$\begin{array}{rclcl} 48x_0 & + & 16x_1 & + & 119x_2 & \leq & 88 \\ 5x_0 & + & 4x_1 & + & 35x_2 & \geq & 13 \\ 15x_0 & + & 4x_1 & + & 20x_2 & \geq & 23 \\ x_0 & , & x_1 & , & x_2 & \geq & 0 \end{array}$$

instance I

$$\begin{array}{rcl} x_0 & = & 1 \\ x_1 & = & 1 \\ x_2 & = & \frac{1}{5} \end{array}$$

solution S

- To check solution S , plug in values and verify each inequality.

Search Problems

Search problem. Given an instance I of a problem, **find** a solution S .
Requirement. Must be able to efficiently **check** that S is a solution.

or report none exists

poly-time in size of instance I

ILP. Given a system of linear inequalities, find a binary solution.

$$\begin{array}{rclcl} & x_1 & + & x_2 & \geq & 1 \\ x_0 & & & + & x_2 & \geq & 1 \\ x_0 & + & x_1 & + & x_2 & \leq & 2 \end{array}$$

instance I

$$\begin{array}{rcl} x_0 & = & 0 \\ x_1 & = & 1 \\ x_2 & = & 1 \end{array}$$

solution S

- To check solution S , plug in values and verify each inequality (and check that solution is 0/1).

Search Problems

Search problem. Given an instance I of a problem, **find** a solution S .
Requirement. Must be able to efficiently **check** that S is a solution.

or report none exists

poly-time in size of instance I

FACTOR. Given an n -bit integer x , find a nontrivial factor.

input size = number of bits

147573952589676412927

193707721

instance I

solution S

- To check solution S , long divide 193707721 into 147573952589676412927.

NP

Def. **NP** is the class of all search problems.

↖ classic definition limits NP to yes-no problems

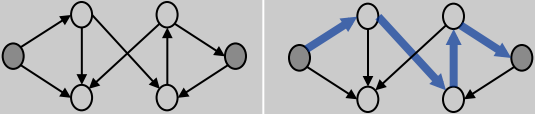
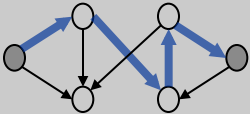
problem	description	poly-time algorithm	instance I	solution S
LSOLVE (A, b)	Find a vector x that satisfies $Ax = b$.	Gaussian elimination	$\begin{aligned} 0x_0 + 1x_1 + 1x_2 &= 4 \\ 2x_0 + 4x_1 - 2x_2 &= 2 \\ 0x_0 + 3x_1 + 15x_2 &= 36 \end{aligned}$	$\begin{aligned} x_0 &= -1 \\ x_1 &= 2 \\ x_2 &= 2 \end{aligned}$
LP (A, b)	Find a vector x that satisfies $Ax \leq b$.	ellipsoid	$\begin{aligned} 48x_0 + 16x_1 + 119x_2 &\leq 88 \\ 5x_0 + 4x_1 + 35x_2 &\geq 13 \\ 15x_0 + 4x_1 + 20x_2 &\geq 23 \\ x_0, x_1, x_2 &\geq 0 \end{aligned}$	$\begin{aligned} x_0 &= 1 \\ x_1 &= 1 \\ x_2 &= 1/5 \end{aligned}$
ILP (A, b)	Find a binary vector x that satisfies $Ax \leq b$.	???	$\begin{aligned} x_1 + x_2 &\geq 1 \\ x_0 + x_2 &\geq 1 \\ x_0 + x_1 + x_2 &\leq 2 \end{aligned}$	$\begin{aligned} x_0 &= 0 \\ x_1 &= 1 \\ x_2 &= 1 \end{aligned}$
FACTOR (x)	Find a nontrivial factor of the integer x .	???	8784561	10657

Significance. What scientists and engineers **aspire to compute** feasibly.

P

Def. **P** is the class of search problems solvable in **poly-time**.

↖ classic definition limits P to yes-no problems

problem	description	poly-time algorithm	instance I	solution S
STCONN (G, s, t)	Find a path from s to t in digraph G .	depth-first search (Theseus)		
SORT (a)	Find permutation that puts a in ascending order.	mergesort (von Neumann 1945)	2.3 8.5 1.2 9.1 2.2 0.3	5 2 4 0 1 3
LSOLVE (A, b)	Find a vector x that satisfies $Ax = b$.	Gaussian elimination (Edmonds, 1967)	$0x_0 + 1x_1 + 1x_2 = 4$ $2x_0 + 4x_1 - 2x_2 = 2$ $0x_0 + 3x_1 + 15x_2 = 36$	$x_0 = -1$ $x_1 = 2$ $x_2 = 2$
LP (A, b)	Find a vector x that satisfies $Ax \leq b$.	ellipsoid (Khachiyan, 1979)	$48x_0 + 16x_1 + 119x_2 \leq 88$ $5x_0 + 4x_1 + 35x_2 \geq 13$ $15x_0 + 4x_1 + 20x_2 \geq 23$ $x_0, x_1, x_2 \geq 0$	$x_0 = 1$ $x_1 = 1$ $x_2 = \frac{1}{5}$

Significance. What scientists and engineers **compute** feasibly.

Extended Church-Turing Thesis

Extended Church-Turing thesis.

P = search problems solvable in poly-time **in nature**.

Evidence supporting thesis. True for all physical computers.

Implication. To make future computers more efficient, suffices to focus on improving implementation of existing designs.

A new law of physics? A constraint on what is possible.

Possible counterexample? Quantum computers.



P vs. NP



Copyright © 1990, Matt Groening



Copyright © 2000, Twentieth Century Fox

Automating Creativity

Q. Being creative vs. appreciating creativity?

Ex. Mozart composes a piece of music; our neurons appreciate it.

Ex. Wiles proves a deep theorem; a colleague referees it.

Ex. Boeing designs an efficient airfoil; a simulator verifies it.

Ex. Einstein proposes a theory; an experimentalist validates it.



creative



ordinary

Computational analog. Does $P = NP$?

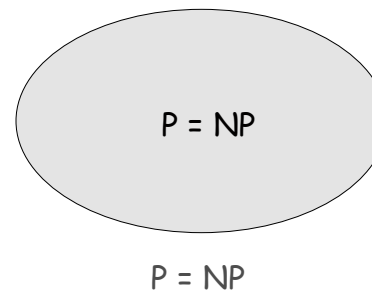
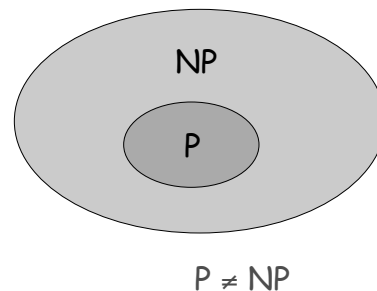
The Central Question

P. Class of search problems solvable in poly-time.

NP. Class of all search problems.

Does $P = NP$? *Can you always avoid brute-force searching and do better?*

Two worlds.



If yes... Poly-time algorithms for 3-SAT, ILP, TSP, FACTOR, ...

If no... Would learn something fundamental about our universe.

Overwhelming consensus. $P \neq NP$.


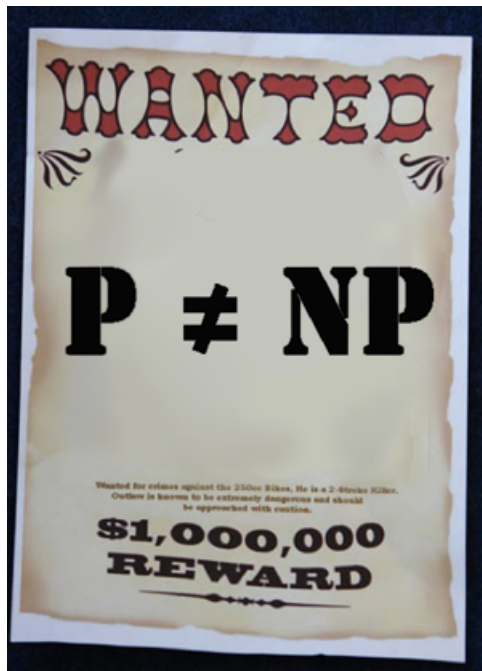
The Central Question

P. Class of search problems solvable in poly-time.

NP. Class of all search problems.

Does $P = NP$? *Can you always avoid brute-force searching and do better?*

Millennium prize. \$1 million for resolution of $P = NP$ problem.



Clay Mathematics Institute
Dedicated to increasing and disseminating mathematical knowledge

HOME | ABOUT CMI | PROGRAMS | NEWS & EVENTS | AWARDS | SCHOLARS | PUBLICATIONS

Millennium Problems

In order to celebrate mathematics in the new millennium, The Clay Mathematics Institute of Cambridge, Massachusetts (CMI) has named seven *Prize Problems*. The Scientific Advisory Board of CMI selected these problems, focusing on important classic questions that have resisted solution over the years. The Board of Directors of CMI designated a \$7 million prize fund for the solution to these problems, with \$1 million allocated to each. During the [Millennium Meeting](#) held on May 24, 2000 at the Collège de France, Timothy Gowers presented a lecture entitled *The Importance of Mathematics*, aimed for the general public, while John Tate and Michael Atiyah spoke on the problems. The CMI invited specialists to formulate each problem.

- ▶ [Birch and Swinnerton-Dyer Conjecture](#)
- ▶ [Hodge Conjecture](#)
- ▶ [Navier-Stokes Equations](#)
- ▶ [P vs NP](#)
- ▶ [Poincaré Conjecture](#)
- ▶ [Riemann Hypothesis](#)
- ▶ [Yang-Mills Theory](#)

- ▶ [Rules](#)
- ▶ [Millennium Meeting Videos](#)

Classifying Problems

Periodic Table of the Elements

1	IA	1	H	2	He	O																														
2	3	IA	Li	4	Be	IIA	5	6	7	8	9	10																								
3	11	Na	12	Mg	13	Al	14	Si	15	P	16	S	17	Cl	18	Ar																				
4	19	K	20	Ca	21	Sc	22	Ti	23	V	24	Cr	25	Mn	26	Fe	27	Co	28	Ni	29	Cu	30	Zn	31	Ga	32	Ge	33	As	34	Se	35	Br	36	Kr
5	37	Rb	38	Sr	39	Y	40	Zr	41	Nb	42	Mo	43	Tc	44	Ru	45	Rh	46	Pd	47	Ag	48	Cd	49	In	50	Sn	51	Sb	52	Te	53	I	54	Xe
6	55	Cs	56	Ba	57	*La	72	Hf	73	Ta	74	W	75	Re	76	Os	77	Ir	78	Pt	79	Au	80	Hg	81	Tl	82	Pb	83	Bi	84	Po	85	At	86	Rn
7	87	Fr	88	Ra	89	+Ac	104	Rf	105	Ha	106	Sg	107	Ns	108	Hs	109	Mt	110	111	112	113														

* Lanthanide Series	58	59	60	61	62	63	64	65	66	67	68	69	70	71
	Ce	Pr	Nd	Pm	Sm	Eu	Gd	Tb	Dy	Ho	Er	Tm	Yb	Lu
+ Actinide Series	90	91	92	93	94	95	96	97	98	99	100	101	102	103
	Th	Pa	U	Np	Pu	Am	Cm	Bk	Cf	Es	Fm	Md	No	Lr

A Hard Problem: 3-Satisfiability

Literal. A Boolean variable or its negation. x_i, x_i'

Clause. An *or* of 3 distinct literals. $C_j = x_1 \text{ or } x_2' \text{ or } x_3$

Conjunctive normal form. An *and* of clauses. $\Phi = C_1 \text{ and } C_2 \text{ and } C_3 \text{ and } C_4$

3-SAT. Given a CNF formula Φ consisting of k clauses over N variables, find a satisfying truth assignment (if one exists).

$$\Phi = (x_1' \text{ or } x_2 \text{ or } x_3) \text{ and } (x_1 \text{ or } x_2' \text{ or } x_3) \text{ and } (x_1' \text{ or } x_2' \text{ or } x_3') \text{ and } (x_1' \text{ or } x_2' \text{ or } x_4)$$

yes: $x_1 = \text{true}, x_2 = \text{true}, x_3 = \text{false}, x_4 = \text{true}$

Key application. Electronic design automation (EDA).

Exhaustive Search

Q. How to solve an instance of 3-SAT with N variables?

A. Exhaustive search: try all 2^N truth assignments.

Q. Can we do anything substantially more clever?

Conjecture. No poly-time algorithm for 3-SAT.

"intractable"



www.jollyon.co.uk

Classifying Problems

Q. Which **search problems** are in P?

A. No easy answers (we don't even know whether $P = NP$).

Goal. Formalize notion:

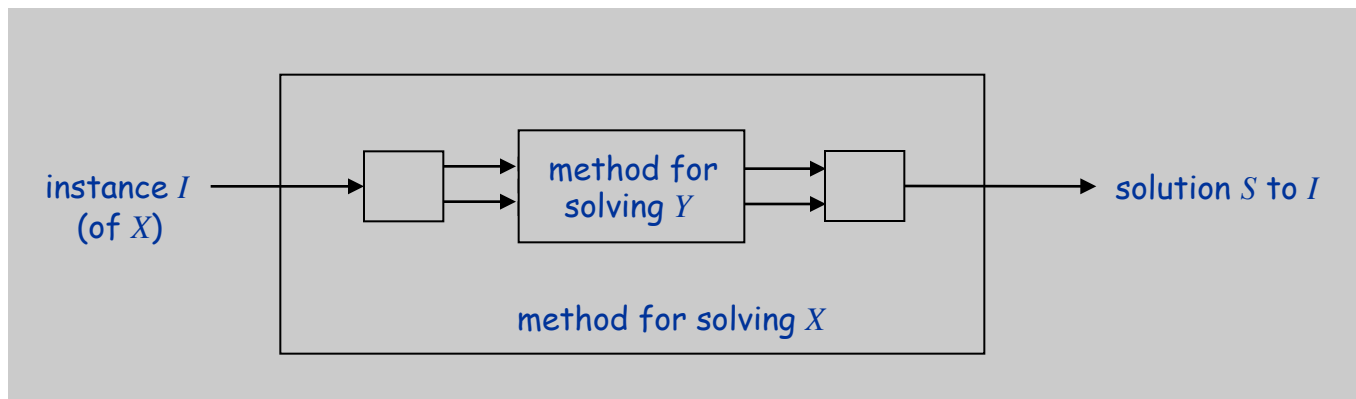
Problem X is computationally not much harder than problem Y.

Reductions

"Cook reduction"



Def. Problem X **reduces to** problem Y if you can use an efficient solution to Y to develop an efficient solution to X :



To solve X , use:

- A poly number of standard computational steps, plus
- A poly number of calls to a method that solves instances of Y .

LSOLVE Reduces to LP

LSOLVE. Given a system of linear equations, find a solution.

$$\begin{array}{rclcl} 0x_0 & + & 1x_1 & + & 1x_2 & = & 4 \\ 2x_0 & + & 4x_1 & - & 2x_2 & = & 2 \\ 0x_0 & + & 3x_1 & + & 15x_2 & = & 36 \end{array}$$

LSOLVE instance with n variables

LP. Given a system of linear inequalities, find a solution.

$$\begin{array}{rclcl} 0x_0 & + & 1x_1 & + & 1x_2 & \leq & 4 \\ 0x_0 & + & 1x_1 & + & 1x_2 & \geq & 4 \\ 2x_0 & + & 4x_1 & - & 2x_2 & \leq & 2 \\ 2x_0 & + & 4x_1 & - & 2x_2 & \geq & 2 \\ 0x_0 & + & 3x_1 & + & 15x_2 & \leq & 36 \\ 0x_0 & + & 3x_1 & + & 15x_2 & \geq & 36 \end{array} \quad \left. \vphantom{\begin{array}{rclcl} 0x_0 & + & 1x_1 & + & 1x_2 & \leq & 4 \\ 0x_0 & + & 1x_1 & + & 1x_2 & \geq & 4 \\ 2x_0 & + & 4x_1 & - & 2x_2 & \leq & 2 \\ 2x_0 & + & 4x_1 & - & 2x_2 & \geq & 2 \\ 0x_0 & + & 3x_1 & + & 15x_2 & \leq & 36 \\ 0x_0 & + & 3x_1 & + & 15x_2 & \geq & 36 \end{array}} \right\} \Rightarrow 0x_0 + 1x_1 + 1x_1 = 4$$

corresponding LP instance with n variables and $2n$ inequalities

3-SAT Reduces to ILP

3-SAT. Given a CNF formula Φ , find a satisfying truth assignment.

$$\Phi = (x'_1 \text{ or } x_2 \text{ or } x_3) \text{ and } (x_1 \text{ or } x'_2 \text{ or } x_3) \text{ and } (x'_1 \text{ or } x'_2 \text{ or } x'_3) \text{ and } (x'_1 \text{ or } x'_2 \text{ or } x_4)$$

3-SAT instance with n variables, k clauses

ILP. Given a system of linear inequalities, find a binary solution.

$$C_1 \geq 1 - x_1$$

$$C_1 \geq x_2$$

$$C_1 \geq x_3$$

$$C_1 \leq (1 - x_1) + x_2 + x_3$$

$C_1 = 1$ iff clause 1 is satisfied
(similar inequalities for $C_2, C_3,$ and C_4)

$$\Phi \leq C_1$$

$$\Phi \leq C_2$$

$$\Phi \leq C_3$$

$$\Phi \leq C_4$$

$$\Phi \geq C_1 + C_2 + C_3 + C_4 - 3$$

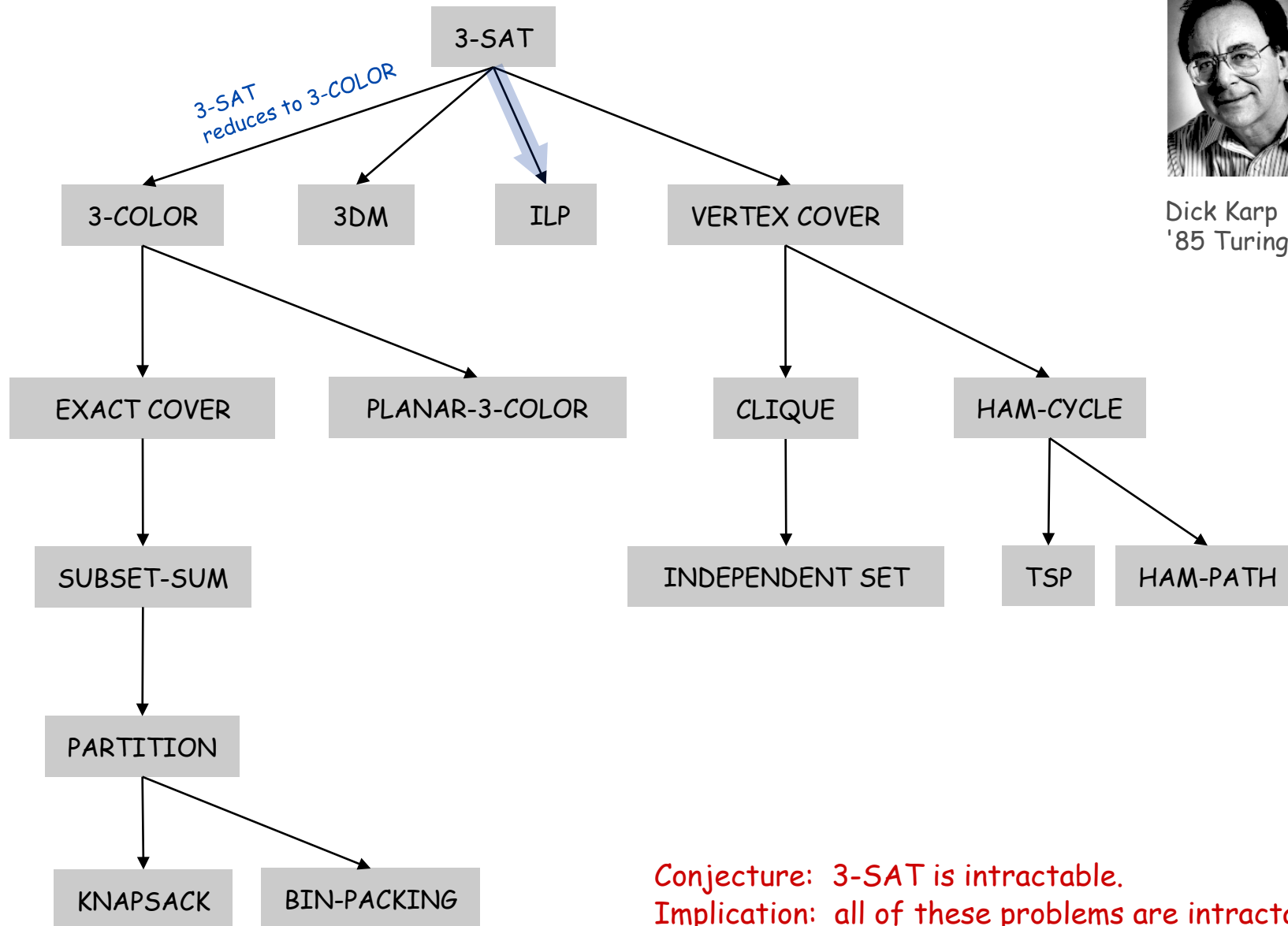
$\Phi = 1$ iff $C_1 = C_2 = C_3 = C_4 = 1$

*corresponding ILP instance with $n + k + 1$ variables and $4k + k + 1$ inequalities
(solution to this ILP instance gives solution to original 3-SAT instance)*

More Reductions From 3-SAT



Dick Karp
'85 Turing award



Still More Reductions from 3-SAT

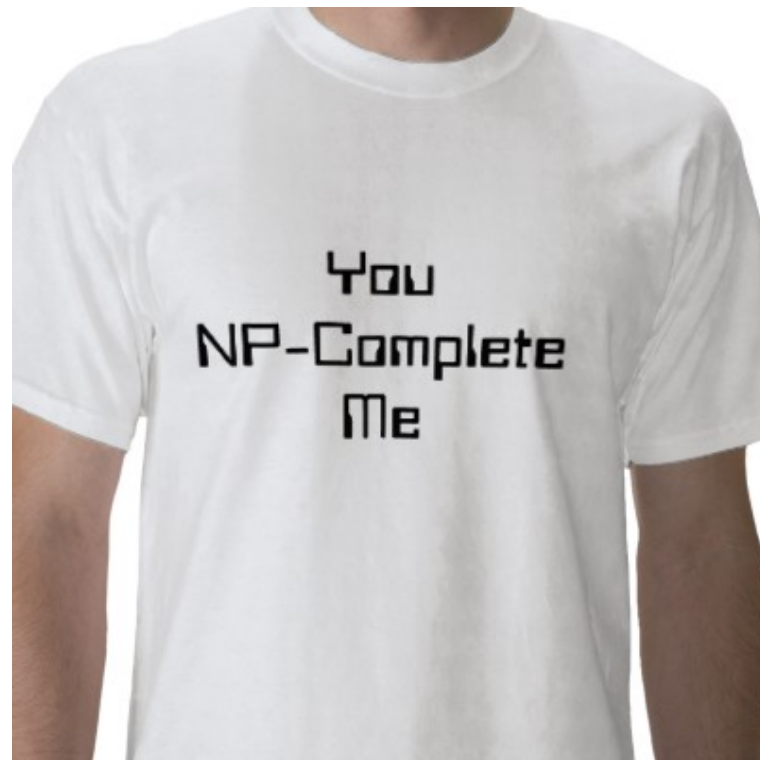
- Aerospace engineering.* Optimal mesh partitioning for finite elements.
- Biology.* Phylogeny reconstruction.
- Chemical engineering.* Heat exchanger network synthesis.
- Chemistry.* Protein folding.
- Civil engineering.* Equilibrium of urban traffic flow.
- Economics.* Computation of arbitrage in financial markets with friction.
- Electrical engineering.* VLSI layout.
- Environmental engineering.* Optimal placement of contaminant sensors.
- Financial engineering.* Minimum risk portfolio of given return.
- Game theory.* Nash equilibrium that maximizes social welfare.
- Mathematics.* Given integer a_1, \dots, a_n , compute $\int_0^{2\pi} \cos(a_1\theta) \times \cos(a_2\theta) \times \dots \times \cos(a_n\theta) d\theta$
- Mechanical engineering.* Structure of turbulence in sheared flows.
- Medicine.* Reconstructing 3d shape from biplane angiogram.
- Operations research.* Traveling salesperson problem, integer programming.
- Physics.* Partition function of 3d Ising model.
- Politics.* Shapley-Shubik voting power.
- Pop culture.* Versions of Sudoku, Checkers, Minesweeper, Tetris.
- Statistics.* Optimal experimental design.

6,000+ scientific papers per year.

Conjecture: 3-SAT is intractable.

Implication: all of these problems are intractable.

NP-completeness



NP-Completeness

Q. Why do we believe 3-SAT is intractable?

Def. An NP problem is **NP-complete** if all problems in NP reduce to it.

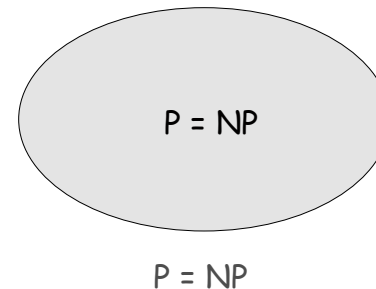
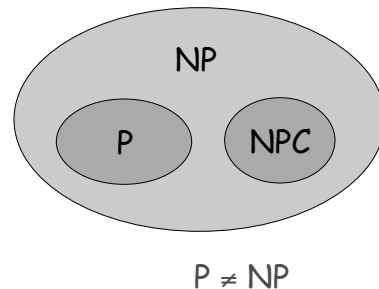
every NP problem is a 3-SAT problem in disguise



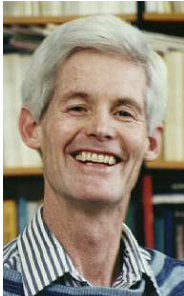
Theorem. [Cook 1971] 3-SAT is NP-complete.

Corollary. Poly-time algorithm for 3-SAT \Leftrightarrow $P = NP$.

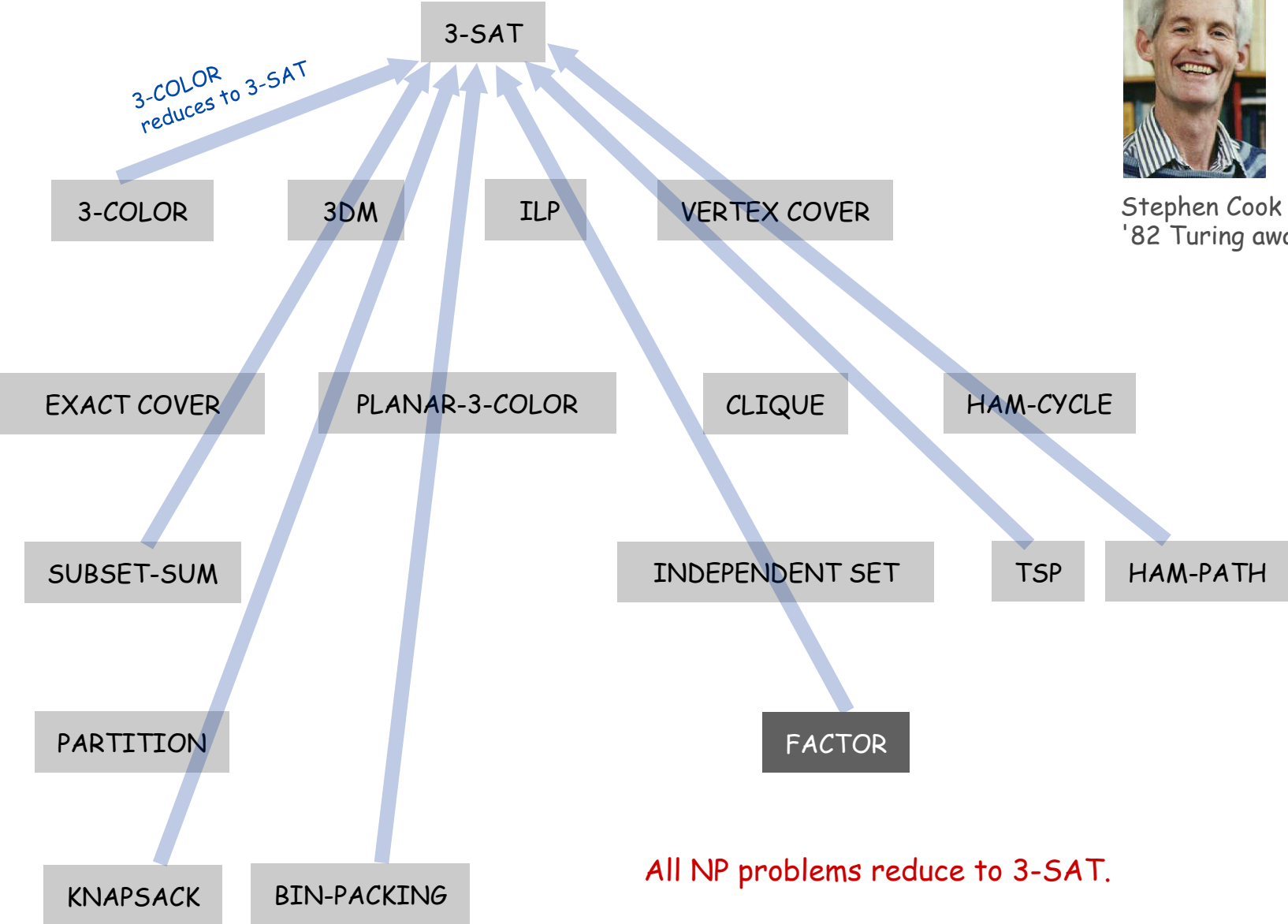
Two worlds.



Cook's Theorem

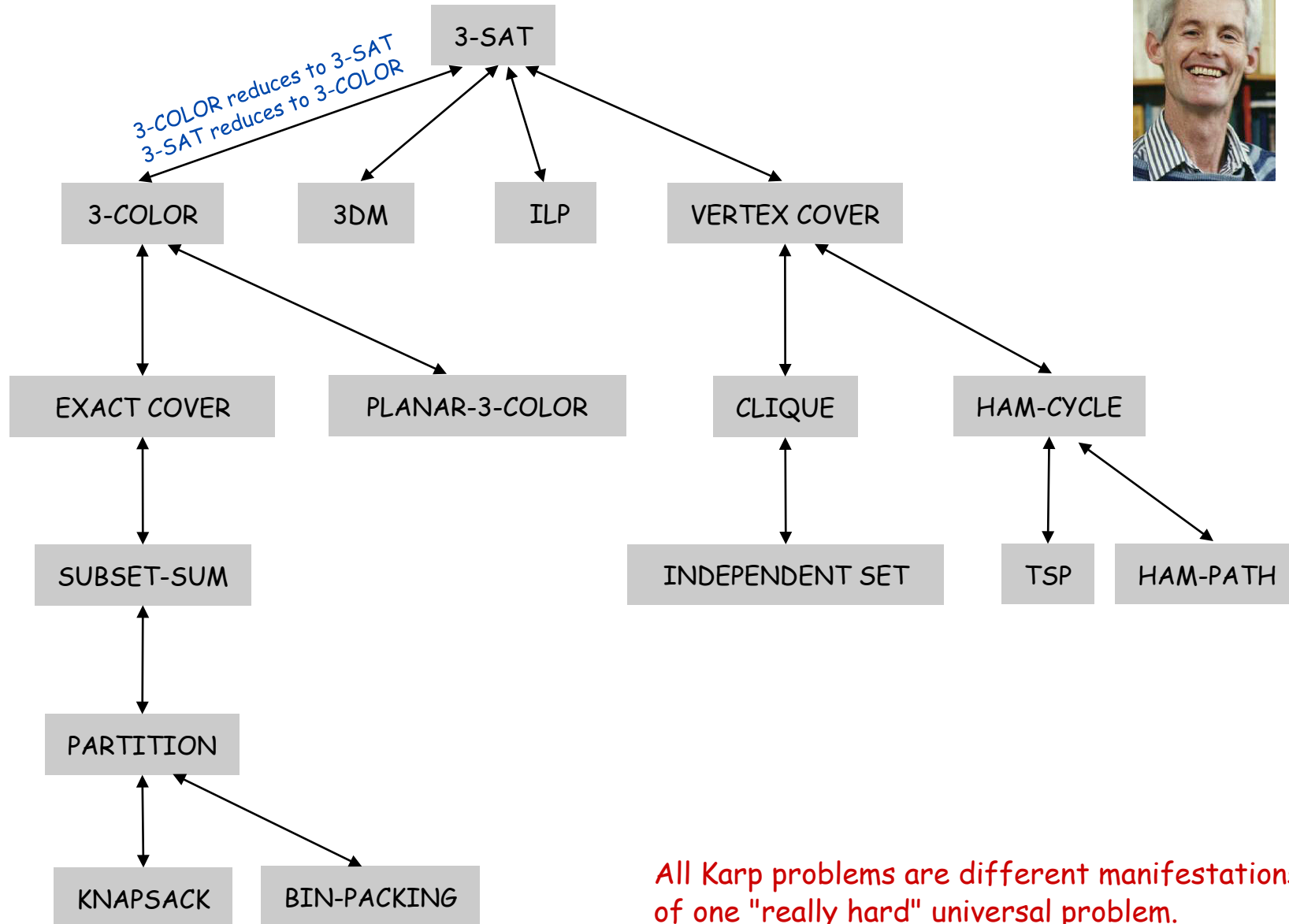
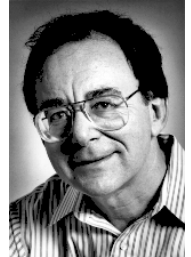


Stephen Cook
'82 Turing award



All NP problems reduce to 3-SAT.

Cook + Karp



All Karp problems are different manifestations of one "really hard" universal problem.

Implications of NP-Completeness

Implication. [3-SAT captures difficulty of whole class NP.]

- Poly-time algorithm for 3-SAT iff $P = NP$.
- If no poly-time algorithm for some NP problem, then none for 3-SAT.

Remark. Can replace 3-SAT with any of Karp's problems.

Proving a problem NP-complete guides scientific inquiry.

- 1926: Ising introduces simple model for phase transitions.
- 1944: Onsager finds closed form solution to 2D version in tour de force.
- 19xx: Feynman and other top minds seek 3D solution.
- 2000: 3D-ISING is NP-complete.

a holy grail of statistical mechanics

search for closed formula appears doomed

Summary

P. Class of search problems solvable in poly-time.

NP. Class of all search problems, some of which seem wickedly hard.

NP-complete. Hardest problems in NP.

Intractable. Problem with no poly-time algorithm.

Many fundamental problems are NP-complete.

- TSP, 3-SAT, 3-COLOR, ILP.
- 3D-ISING.

Use theory a guide:

- A poly-time algorithm for an NP-complete problem would be a stunning breakthrough (a proof that $P = NP$).
- You will confront NP-complete problems in your career.
- Safe to assume that $P \neq NP$ and that such problems are intractable.
- Identify these situations and proceed accordingly.

Coping With NP-completeness

Coping With NP-completeness

Relax one of desired features.

- Solve the problem in poly-time.
- Solve the problem to optimality.
- Solve arbitrary instances of the problem.

Complexity theory deals with worst case behavior.

- Instance(s) you want to solve may be "easy."
- Chaff solves real-world SAT instances with ~ 10k variable.
[Matthew Moskewicz '00, Conor Madigan '00, Sharad Malik]

↑
PU senior independent work (!)

Coping With NP-completeness

Relax one of desired features.

- Solve the problem in poly-time.
- Solve the problem to optimality.
- Solve arbitrary instances of the problem.

Develop a heuristic, and hope it produces a good solution.

- No guarantees on quality of solution.
- Ex. TSP assignment heuristics.
- Ex. Metropolis algorithm, simulating annealing, genetic algorithms.

Approximation algorithm. Find solution of provably good quality.

- Ex. MAX-3SAT: provably satisfy 87.5% as many clauses as possible.

but if you can guarantee to satisfy 87.51% as many clauses as possible in poly-time, then $P = NP$!

Coping With NP-completeness

Relax one of desired features.

- Solve the problem in poly-time.
- Solve the problem to optimality.
- Solve arbitrary instances of the problem.

Special cases may be tractable.

- Ex: Linear time algorithm for 2-SAT.
- Ex: Linear time algorithm for Horn-SAT.



each clause has at most one un-negated literal

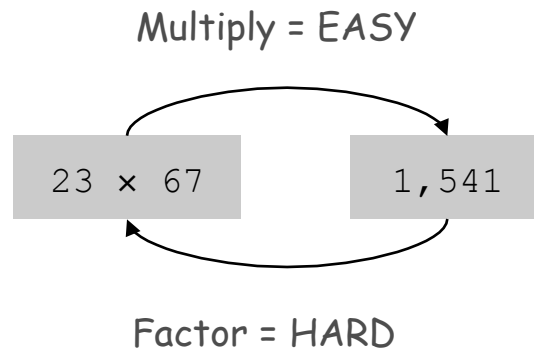
Exploiting Intractability: Cryptography

Modern cryptography.

- Ex. Send your credit card to Amazon.
- Ex. Digitally sign an e-document.
- Enables freedom of privacy, speech, press, political association.

RSA cryptosystem.

- To use: multiply two n -bit integers. [poly-time]
- To break: factor a $2n$ -bit integer. [unlikely poly-time]



Fame and Fortune through CS (revisited)

Challenge. Factor this number.

74037563479561712828046796097429573142593188889231289
08493623263897276503402826627689199641962511784399589
43305021275853701189680982867331732731089309005525051
16877063299072396380786710086096962537934650563796359

RSA-704
(\$30,000 prize if you can factor)

Can't do it? Create a company based on the difficulty of factoring.

$P \ \& \ Q \ \text{PRIME}$ $N = PQ$ $ED = 1 \ \text{MOD} \ (P-1)(Q-1)$ $C = M^E \ \text{MOD} \ N$ $M = C^D \ \text{MOD} \ N$
The RSA algorithm is the most widely used method of implementing public key cryptography and has been deployed in more than one billion applications worldwide.

RSA algorithm



RSA sold
for \$2.1 billion



or design a t-shirt

A Final Thought

FACTOR. Given an n -bit integer x , find a nontrivial factor.

Q. What is complexity of FACTOR?

A. In NP, but not known (or believed) to be in P or NP-complete.

Q. What if $P = NP$?

A. Poly-time algorithm for factoring; modern e-conomy collapses.

Quantum factoring algorithm. [Shor 1994] Can factor an n -bit integer in n^3 steps on a "quantum computer."

Q. Do we still believe the extended Church-Turing thesis???

CS Building, West Wall



Princeton CS Building, West Wall, Circa 2001

1 0 1 0 0 0 0

char	ASCII	binary
P	80	1010000
=	61	0111101
N	78	1001110
P	80	1010000
?	63	0111111