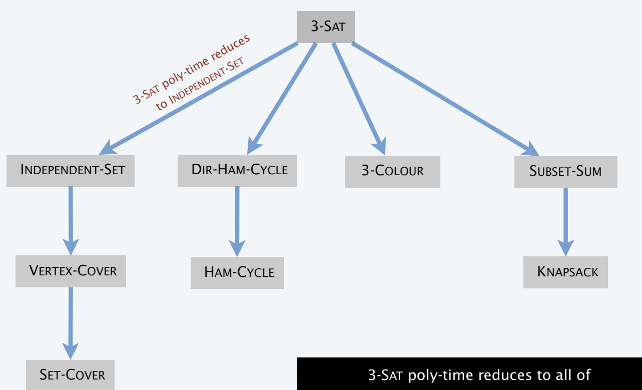
CSCI 355: Algorithm Design and Analysis
10. Intractability

- P vs. NP
- NP-completeness

Review: reducibility



3-Sat poly-time reduces to all of these problems (and many, many more)

CSCI 355: Algorithm Design and Analysis
10. Intractability

- *P vs. NP*
- NP-completeness

The class P

Decision problems.

- A problem X is a set of strings.
- An instance s of a problem is one string.
- An algorithm A solves problem X: $A(s) = \begin{cases} yes & \text{if } s \in X \\ no & \text{if } s \notin X \end{cases}$

Def. Algorithm A runs in polynomial time if, for every string s, $A(s)$ terminates in $\leq p(|s|)$ "steps," where $p(\cdot)$ is some polynomial function.

length of s

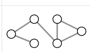
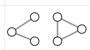
Def. **P** = set of decision problems for which there exists a poly-time algorithm.

on a deterministic Turing machine

problem PRIMES:	$\{ 2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, \ldots \}$
instance s:	592335744548702854681
algorithm:	Agrawal–Kayal–Saxena (2002)

Some problems in P

P. Set of decision problems for which there exists a poly-time algorithm.

problem	description	poly-time algorithm	yes	no
MULTIPLE	Is x a multiple of y?	grade-school division	51, 17	51, 16
REL-PRIME	Are x and y relatively prime?	Euclid's algorithm	34, 39	34, 51
PRIMES	Is x prime?	Agrawal–Kayal–Saxena	53	51
EDIT-DISTANCE	Is the edit distance between x and y less than 5?	Needleman–Wunsch	niether neither	acgggt ttttta
L-SOLVE	Is there a vector x that satisfies $Ax = b$?	Gauss–Edmonds elimination	$\begin{bmatrix} 0 & 1 & 1 \\ 2 & 4 & -2 \\ 0 & 3 & 15 \end{bmatrix}, \begin{bmatrix} 4 \\ 2 \\ 36 \end{bmatrix}$	$\begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 1 \end{bmatrix}, \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$
U-CONN	Is an undirected graph G connected?	depth-first search		

The class NP

Def. An algorithm $C(s, t)$ is a certifier for problem X if for every string s: $s \in X$ iff there exists a string t such that $C(s, t) = yes$.

"certificate" or "witness"

Def. **NP** = set of decision problems for which there exists a poly-time certifier.

- $C(s, t)$ is a poly-time algorithm.
- Certificate t is of polynomial size: $|t| \leq p(|s|)$ for some polynomial $p(\cdot)$.

problem COMPOSITES:	$\{ 4, 6, 8, 9, 10, 12, 14, 15, 16, 18, 20, \ldots. \}$
instance s:	437669
certificate t:	541 ← 437,669 = 541 × 809
certifier C(s, t):	grade-school division

Certifiers and certificates: satisfiability

SAT. Given a CNF formula Φ, does it have a satisfying truth assignment?
3-SAT. SAT where each clause contains exactly 3 literals.

Certificate. An assignment of truth values to the Boolean variables.

Certifier. Checks that each clause in Φ has at least one true literal.

instance s $\Phi = \left(\overline{x_1} \vee x_2 \vee x_3\right) \wedge \left(x_1 \vee \overline{x_2} \vee x_3\right) \wedge \left(\overline{x_1} \vee x_2 \vee x_4\right)$

certificate t $x_1 = true,\ x_2 = true,\ x_3 = false,\ x_4 = false$

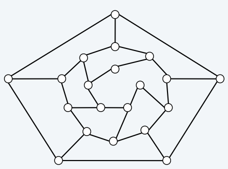
Conclusions. SAT $\in$ **NP**, 3-SAT $\in$ **NP**.

Certifiers and certificates: Hamiltonian path

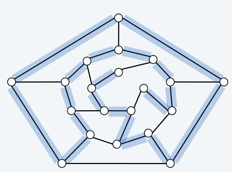
HAMILTON-PATH. Given an undirected graph $G = (V, E)$, does there exist a simple path P that visits every vertex?

Certificate. A permutation π of the n vertices.

Certifier. Checks that π contains each vertex in V exactly once, and that G contains an edge between each pair of adjacent vertices.



instance s

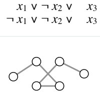
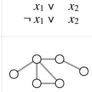


certificate t

Conclusion. HAMILTON-PATH $\in$ **NP**.

Some problems in NP

NP. Set of decision problems for which there exists a poly-time certifier.

problem	description	poly-time algorithm	yes	no
L-SOLVE	Is there a vector x that satisfies $Ax = b$?	Gauss–Edmonds elimination	$\begin{bmatrix} 0 & 1 & 1 \\ 2 & 4 & -2 \\ 0 & 3 & 15 \end{bmatrix}, \begin{bmatrix} 4 \\ 2 \\ 36 \end{bmatrix}$	$\begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 1 \end{bmatrix}, \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$
COMPOSITES	Is x composite?	Agrawal–Kayal–Saxena	51	53
FACTOR	Does x have a nontrivial factor less than y?	???	(56159, 50)	(55687, 50)
SAT	Given a CNF formula, does it have a satisfying truth assignment?	???	$\neg x_1 \vee x_2 \vee \neg x_3$ $x_1 \vee \neg x_2 \vee x_3$ $\neg x_1 \vee \neg x_2 \vee x_3$	$\neg x_2$ $x_1 \vee x_2$ $\neg x_1 \vee x_2$
HAMILTON-PATH	Is there a simple path between u and v that visits every vertex?	???		

Significance of NP

NP. Set of decision problems for which there exists a poly-time certifier.

> *" NP captures vast domains of computational, scientific, and mathematical endeavours, and seems to roughly delimit what mathematicians and scientists have been aspiring to compute feasibly. "*
>
> — *Christos Papadimitriou*

The classes P, NP, and EXP

P. Set of decision problems for which there exists a poly-time algorithm.
NP. Set of decision problems for which there exists a poly-time certifier.
EXP. Set of decision problems for which there exists an exp-time algorithm.

Proposition. **P** $\subseteq$ **NP**.
Pf. Consider any problem $X \in$ **P**.
- By definition, there exists a poly-time algorithm $A(s)$ that solves X.
- Certificate is $t = \varepsilon$, certifier is $C(s, t) = A(s)$. ▪

Proposition. **NP** $\subseteq$ **EXP**.
Pf. Consider any problem $X \in$ **NP**.
- By definition, there exists a poly-time certifier $C(s, t)$ for X where a certificate t satisfies $|t| \leq p(|s|)$ for some polynomial $p(\cdot)$.
- To solve the instance s, run $C(s, t)$ on all strings t with $|t| \leq p(|s|)$.
- Return *yes* iff $C(s, t)$ returns *yes* for any of these potential certificates. ▪

Fact. **P** $\neq$ **EXP** $\Rightarrow$ either **P** $\neq$ **NP**, or **NP** $\neq$ **EXP**, or both.

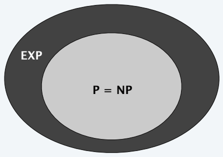
The big question: P vs. NP

Q. How do we solve an instance of 3-SAT with n variables?
A. Exhaustive search: try all 2^n truth assignments.

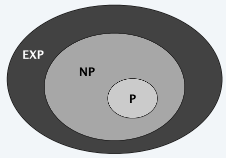
Q. Can we do anything substantially more clever?
Conjecture. There exists $\underbrace{\text{no poly-time algorithm}}_{\text{"intractable"}}$ for 3-SAT.

The big question: P vs. NP

Does P = NP? [Cook, Levin, ...]
Is the decision problem as easy as the certification problem?



If P = NP



If P ≠ NP

If yes... Efficient algorithms exist for 3-SAT, TSP, VERTEX-COVER, FACTOR, ...
If no... No efficient algorithms are possible for 3-SAT, TSP, VERTEX-COVER, ...

Consensus opinion. Probably no.

Possible outcomes

P ≠ NP

" I conjecture that there is no good algorithm for the traveling salesman problem. My reasons are the same as for any mathematical conjecture: (i) It is a legitimate mathematical possibility and (ii) I do not know."

— Jack Edmonds (1966)

" My intuitive belief is that P is unequal to NP [...] I believe that the traditional proof techniques will not suffice. Something entirely novel will be required.

My hunch is that the problem will be solved by a young researcher who is not encumbered by too much conventional wisdom about how to attack the problem. "

— Richard Karp (2002)

Possible outcomes

P ≠ NP

" When I was a graduate student in the mid 1970s, I predicted that it would be solved by the century's end. I also bet Len Adleman an ounce of gold that I would be right.

Now that I've paid off, I'm more reluctant to make a prediction once again. But I'll go out on a limb and give it another 25 years, so by around 2025. And I'll stick with my earlier prediction that the resolution will be a proof that P ≠ NP. The technique would be combinatorial, but that isn't saying much. No more bets, however. "

— Michael Sipser (2002)

Possible outcomes

P = NP

> " I think that in this respect I am on the loony fringe of the mathematical community: I think (not too strongly!) that P=NP and this will be proved within twenty years. Some years ago, Charles Read and I worked on it quite a bit, and we even had a celebratory dinner in a good restaurant before we found an absolutely fatal mistake. "
>
> — *Béla Bollobás (2002)*

Other possible outcomes

P = NP, but with a $\Omega(n^{100})$ algorithm for 3-SAT.

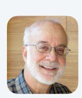
P ≠ NP, but with a $O(n^{\log^* n})$ algorithm for 3-SAT.

P = NP is independent of ZFC axiomatic set theory.

> " It will be solved by either 2048 or 4096. I am currently somewhat pessimistic. The outcome will be the truly worst case scenario: namely that someone will prove P = NP because there are only finitely many obstructions to the opposite hypothesis; hence there exists a polynomial time solution to SAT but we will never know its complexity! "
>
> — *Donald Knuth (2002)*

Other possible outcomes

> " I feel that theoretical computer scientists should devote a constant fraction of their lives to trying to resolve the P vs. NP question.
>
> I personally spend a few days each year thinking about it. I've proven (at least twice) that NP does not equal co-NP (and hence P does not equal NP). I've also proven (also at least twice) that NP equals co-NP.
>
> My most recent proof that NP does not equal co-NP occurred about a week ago as I write this, and the proof survived for about half an hour (not quite long enough for me to run it by someone else). My longest-surviving proof that NP does not equal co-NP survived for about 3 days and fooled some very smart people into believing it. "
>
> — *Ronald Fagin (2002)*

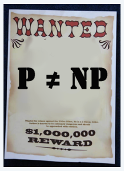


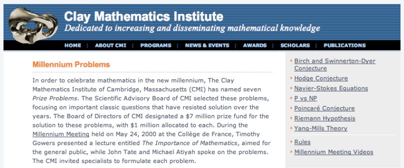
Millennium prize

Millennium Problems. $1 million for a resolution to the **P** vs. **NP** problem.

The only Millennium Problem relating to CS!

- Birch and Swinnerton-Dyer conjecture
- Hodge conjecture
- Navier-Stokes existence and smoothness
- **P vs. NP problem**
- ~~Poincaré conjecture~~ (solved)
- Riemann hypothesis
- Yang-Mills existence and mass gap

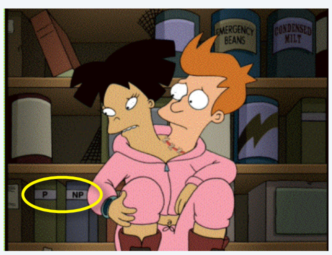




P vs. NP and pop culture

Some writers for the Simpsons and Futurama.

- J. Steward Burns. *M.S. in mathematics (Berkeley '93).*
- David X. Cohen. *M.S. in computer science (Berkeley '92).*
- Al Jean. *B.S. in mathematics. (Harvard '81).*
- Ken Keeler. *Ph.D. in applied mathematics (Harvard '90).*
- Jeff Westbrook. *Ph.D. in computer science (Princeton '89).*



CSCI 355: Algorithm Design and Analysis
10. Intractability

- P vs. NP
- *NP-completeness*

NP-completeness

NP-completeness. A problem $Y \in$ **NP** is NP-complete if it has the property that for every problem $X \in$ **NP**, $X \le_P Y$.

Proposition. Suppose $Y \in$ **NP**-complete. Then $Y \in$ **P** iff **P** = **NP**.

Pf.

[⇐] If **P** = **NP**, then $Y \in$ **P** because $Y \in$ **NP**.

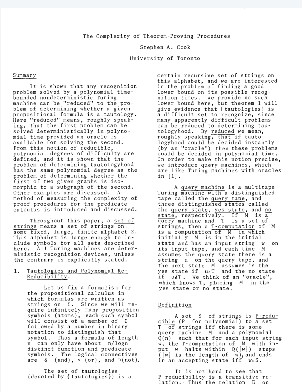
[⇒] Suppose $Y \in$ **P**.

- Consider any problem $X \in$ **NP**. Since $X \le_P Y$, we have $X \in$ **P**.
- This implies **NP** ⊆ **P**.
- We already know **P** ⊆ **NP**. Thus **P** = **NP**. ▪

Fundamental question. Are there any "natural" **NP**-complete problems?

The first NP-complete problem

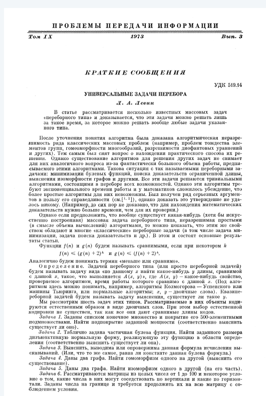
Theorem. [Cook 1971, Levin 1973] SAT ∈ **NP**-complete.



The Complexity of Theorem-Proving Procedures

Stephen A. Cook

University of Toronto



ПРОБЛЕМЫ ПЕРЕДАЧИ ИНФОРМАЦИИ

Том IX 1973 Вып. 3

КРАТКИЕ СООБЩЕНИЯ

УДК 519.14

УНИВЕРСАЛЬНЫЕ ЗАДАЧИ ПЕРЕБОРА

Л. А. Левин

Establishing NP-completeness

Remark. Once we establish the first "natural" **NP**-complete problem, the others fall like dominoes.

Recipe. To prove that $Y \in$ **NP**-complete:

- Step 1. Show that $Y \in$ **NP**.
- Step 2. Choose an **NP**-complete problem X.
- Step 3. Prove that $X \le_P Y$.

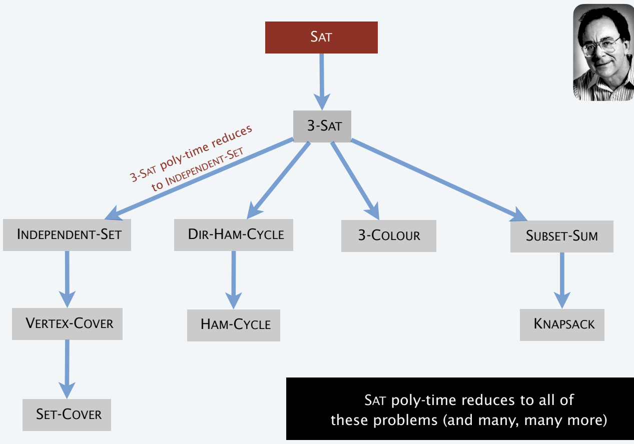
Proposition. If $Y \in$ **NP**, $X \in$ **NP**-complete, and $X \le_P Y$, then $Y \in$ **NP**-complete.

Pf. Consider any problem $W \in$ **NP**. Then, both $W \le_P X$ and $X \le_P Y$.

by definition of **NP**-completeness   by assumption

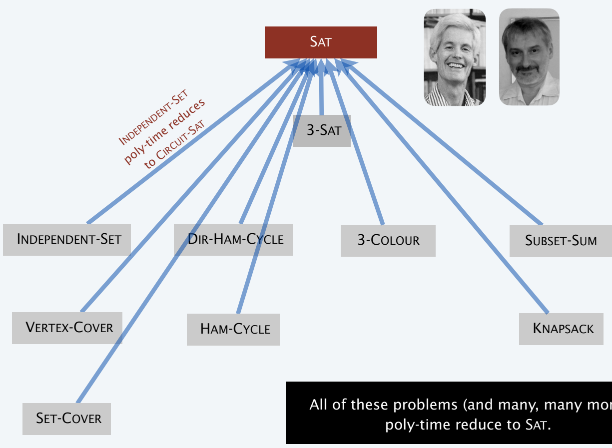
- By transitivity, $W \le_P Y$.
- Hence $Y \in$ **NP**-complete. ▪

Implications of Karp



SAT poly-time reduces to all of these problems (and many, many more)

Implications of Cook–Levin



All of these problems (and many, many more) poly-time reduce to SAT.

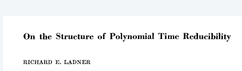
Some NP-complete problems

Basic classes of NP-complete problems and examples.
- Packing/covering problems: SET-COVER, VERTEX-COVER, INDEPENDENT-SET.
- Constraint satisfaction problems: SAT, 3-SAT, CIRCUIT-SAT.
- Sequencing problems: HAMILTON-CYCLE, TSP.
- Partitioning problems: 3-COLOUR, 3D-MATCHING.
- Numerical problems: SUBSET-SUM, KNAPSACK.

Practice. Most **NP** problems are known to be either in **P** or **NP**-complete.

"NP-intermediate" problems? FACTOR, DISCRETE-LOG, GRAPH-ISOMORPHISM, …

Theorem. [Ladner 1975] Unless **P** = **NP**, there exist problems in **NP** that are neither in **P** nor **NP**-complete.


On the Structure of Polynomial Time Reducibility

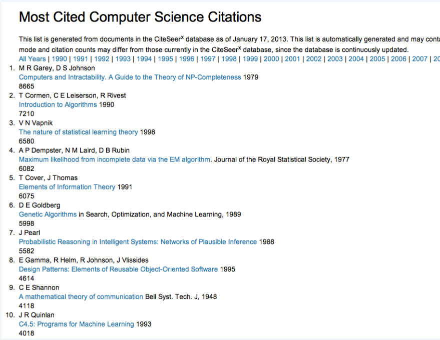
RICHARD E. LADNER

University of Washington, Seattle, Washington

More hard computational problems

M. R. Garey and D. S. Johnson. *Computers and Intractability.*

- Appendix includes over 300 **NP**-complete problems.
- Most cited reference in computer science literature.

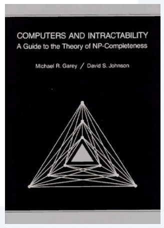


Most Cited Computer Science Citations

This list is generated from documents in the CiteSeer^x database as of January 17, 2013. This list is automatically generated and may contain errors. The list is generated in batch mode and citation counts may differ from those currently in the CiteSeer^x database, since the database is continuously updated.

All Years | 1990 | 1991 | 1992 | 1993 | 1994 | 1995 | 1996 | 1997 | 1998 | 1999 | 2000 | 2001 | 2002 | 2003 | 2004 | 2005 | 2006 | 2007 | 2008 | 2009 | 2010 | 2011 | 2012 | 2013

1. M R Garey, D S Johnson
   Computers and Intractability. A Guide to the Theory of NP-Completeness 1979
   8665
2. T Cormen, C E Leiserson, R Rivest
   Introduction to Algorithms 1990
   7210
3. V N Vapnik
   The nature of statistical learning theory 1998
   6580
4. A P Dempster, N M Laird, D B Rubin
   Maximum likelihood from incomplete data via the EM algorithm. Journal of the Royal Statistical Society, 1977
   6082
5. T Cover, J Thomas
   Elements of Information Theory 1991
   6075
6. D E Goldberg
   Genetic Algorithms in Search, Optimization, and Machine Learning, 1989
   5998
7. J Pearl
   Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference 1988
   5582
8. E Gamma, R Helm, R Johnson, J Vlissides
   Design Patterns: Elements of Reusable Object-Oriented Software 1995
   4614
9. C E Shannon
   A mathematical theory of communication Bell Syst. Tech. J, 1948
   4118
10. J R Quinlan
    C4.5: Programs for Machine Learning 1993
    4018



More hard computational problems

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 $a_1, \ldots, a_n$, compute $\int_0^{2\pi} \cos(a_1\theta) \times \cos(a_2\theta) \times \cdots \times \cos(a_n\theta)\, d\theta$
Mechanical engineering. Structure of turbulence in sheared flows.
Medicine. Reconstructing 3d shape from biplane angiocardiogram.
Operations research. Traveling salesperson problem.
Physics. Partition function of 3d Ising model.
Politics. Shapley–Shubik voting power.
Recreation. Versions of Sudoku, Checkers, Minesweeper, Tetris, Rubik's Cube.
Statistics. Optimal experimental design.

Extent and impact of NP-completeness

Extent of NP-completeness. [Papadimitriou 1995]

- Prime intellectual export of CS to other disciplines.
- 6,000 citations per year (more than "compiler", "OS", "database").
- Broad applicability and classification power.

NP-completeness can guide scientific inquiry.

- 1926: Ising introduces a simple model for phase transitions.
- 1944: Onsager finds a closed-form solution to 2D-ISING.
- 19xx: Top minds seek a solution to 3D-ISING. ← a holy grail of statistical mechanics
- 2000: Istrail proves 3D-ISING ∈ **NP**-complete.

the search for a closed formula appears doomed


Ising


Onsager


Istrail